Composite Hazard Analysis of System of Systems for Mixed-traffic Automation in Underground Mine

Nazakat Ali and Sasikumar Punnekkat
School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden
{nazakat.ali, sasikumar.punnekkat}@mdu.se

***Abstract*—** Hazard analysis for a single system focuses on identifying and evaluating potential hazards associated with the individual system, its components, and their interactions. There are well-established hazard analysis techniques that are widely used to identify hazards for single systems. However, unlike single systems, hazard analysis in a System of Systems (SoS) must focus on analyzing the potential hazards (including emergent ones) that can arise from the interactions between multiple individual systems. This type of analysis considers the complex interactions between systems and the interdependence between their components and the environment in which they operate. Therefore, it is necessary to understand the application scenarios of SoS and to employ a systematic approach to identify all potential hazards. This paper applies the Composite Hazard Analysis Technique (CompHAT) to an industrial case study from a mining and equipment domain. The results show that the CompHAT is useful in identifying the interaction faults and their propagation routes between components of a constituent system and between constituent systems in an SoS. We also report that, due to the tool support, CompHAT can be beneficial for safety engineers to trace the faults in the network of an SoS in a more efficient and effective manner.

***Index Terms*—**Hazard Analysis, System of Systems, Safety, Underground Mine

I. Introduction

The collaboration of intelligent systems provide functionality that is greater than the sum of the individual systems. This extended functionality offers new capabilities and solutions that would not be possible with individual systems working in isolation. Such kind of system is also called a System of Systems (SoS). An SoS refers to a set of multiple, independent, and diverse systems integrated into a larger system to achieve a common goal [1]. For instance, autonomous vehicles enable autonomous driving systems to interact with each other and their surrounding infrastructure to improve traffic safety flow, safety, robustness, and reliability [2]. However, these autonomous systems rely on receiving data from other systems timely and reliably to complete their mission safely. Therefore, the SoS´s safety does not relate to a single system anymore but stretches across the constituent systems in an SoS.

SoS presents many challenges and requires a different system design and integration approach to make it reliable and safe. The main challenge lies in making sure that multiple systems work together seamlessly and that the overall system achieves its common mission while ensuring safety. This requires a holistic approach to system design and integration, which considers the inter-dependencies between the systems and the potential impact of changes in one system on the others. For example, it is possible that a fault in a constituent system of an SoS would not have a significant impact on it and would not be considered safety critical. However, this fault or erroneous data can be communicated with other systems that may be aggregated with other inputs, leading to a failure. Therefore, it is unclear how to achieve safety when developing such an SoS because existing functional safety standards, such as IEC 61508 [3] or ISO 26262 [4], do not explicitly cover SoS. The interaction of multiple systems leads to a behavior that cannot be attributed to a single system and cannot be specified for each participating system, making it impossible to ensure that each system behaves correctly. Therefore, we need a hazard analysis technique that would consider hazards for the entire SoS, including hazards that stem from the interactions.

In our previous studies [5]–[7], we proposed a **Comp**osite **H**azard **A**nalysis **T**echnique (CompHAT) and developed a tool called $SoCPSTracer$ to analyze hazards for SoS. In this paper, we apply $SoCPSTracer$ on an SoS (mixed traffic automation in an underground mine) to see whether the $SoCPSTracer$ can be applied to identify hazards for an SoS and facilitate the identification of interaction hazards, their causes, sources, and consequences. In particular, we focus on the following research questions:

RQ1. Can a hazard analysis be performed in an SoS using CompHAT?

RQ2. Can we identify hazards related to interactions within components of a constituent system or between the constituent systems of an SoS using $SoCPSTracer$?

The remainder of this paper is organized as follows. Section II discusses the necessary background and related work. In section III, we describe the mixed traffic automation for underground mines (case study), while in Section IV, we apply the composite hazard analysis technique to analyze hazards for the above case study. Section V concludes this paper by presenting potential limitations and some future research directions.

II. Background and Related Work

A. System of Systems

In this subsection, we provide a detailed description of an SoS. There are various definitions of SoS; however, according

to ISO 21841 standard, SoS is defined as a "set of systems or system elements that interact to provide a unique capability that none of the constituent systems can accomplish on its own." [8]. The constituent systems are considered to be independent systems that form an SoS. Maier [1] categorized SoS into directed, collaborative, and virtual. Our case study falls under the category of directed SoS.

B. Composite Hazard Analysis

The hazard analysis for SoS should be performed as part of the system safety process. However, safety analysts need to consider new aspects to complete the steps within the safety analysis process. The aspect can be: 1) identification of emergent hazards, 2) system interdependencies, and 3) traceability of hazards and residual mishap risk. Patrick et al. [9] have divided SoS hazards into two categories: Single-system hazards and emergent hazards. A single-system hazard can be attributed to a single-system alone. In contrast, an SoS hazard can be any hazard that may occur due to interactions between constituent systems of an SoS. This type of hazard is called an emergent hazard. The authors further divided emergent hazards into reconfiguration, integration, and interoperability hazard.

In order to address the above hazards, we have proposed a Composite Hazarad Analysis Technique (CompHAT) and developed a tool i.e., $SoCPSTracer$ [7], which considers interactions between components of a constituent system and interactions between constituent systems of an SoS. Fig. 1 shows the framework for $SoCPSTracer$. CompHAT has three steps:

- List all the constituent systems that make an SoS
- Perfrom hazard analysis of each constituent systems using FTA,ETA and FMEA
- Apply SoCPSTracer
- Analyze the Results

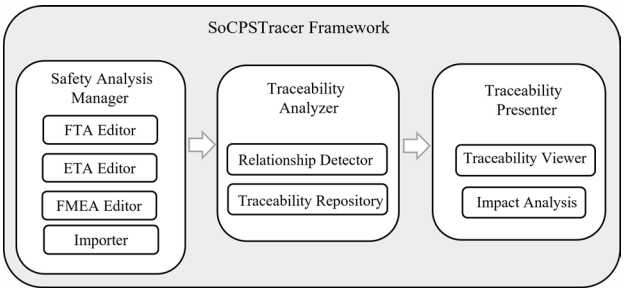

Figure 1. SoCPSTrcer Framework [7]

The major components of $SoCPSTracer$ are as follows:

Safety Analysis Manager:
: It is comprised of four components, including the Fault Tree Analysis (FTA) editor, Event Tree Analysis (ETA) editor, Failure Mode and Event Analysis (FMEA) editor, and importer, as depicted in Fig.1 on the left-hand side. The constituent systems within the network of SoS can be evaluated using these hazard analysis techniques and hazard analysis artifacts are generated for the constituent systems. These artifacts are then stored in the hazard analysis artifact repository. In addition, the importer can bring in previously created hazard analysis artifacts for the constituent systems within an SoS.

Traceability Analyzer:
: It comprises a relation detector and a traceability repository, as shown in Fig.1 at the top-right. The relation detector is a component that recognizes and connects the trace links (relationships) among hazard artifacts. For this purpose, we defined the following three relationships among the content of hazard analysis artifacts.
  - Influence Relationship: A relationship in which a fault of a constituent system affects another constituent system(s) in an SoS.
  - Overlap Relationship: If two faults in the network of SoS result in the same consequences, then their consequences will have an overlapping relationship.
  - Countermeasure Relationship: A countermeasure relationship exists when the safety guard for a particular fault in one constituent system is used to counter a fault (s) in another constituent system in an SoS.

Traceability Presenter:
: It is a component of $SoCPSTracer$ comprising a traceability viewer and an impact analysis subcomponent. The traceability viewer displays visual trace information among hazard analysis artifacts, referred to as Fault Propagation Graph (FPG) in $SoCPSTracer$, as seen in Fig.1 at the top-right. The FPG represents the relationships between hazard analysis artifacts. The impact analysis in the FPG helps to assess a fault's impact on other constituent systems.

C. Related Work

Apart from the advantages we receive, safety in SoS remains a thorny challenge, and researchers are investigating it in different directions. For instance, Michael et al. [1] proposed a new model and metrics framework for validating the adequacy of software safety requirements in safety-critical software-intensive systems and SoS. The framework is applied to a representative component system of a missile defense SoS to demonstrate its effectiveness in identifying potential issues with software safety requirements early in the development process. The authors mentioned that the framework could not entirely replace the process of validating software safety requirements; it encourages a proactive approach to ensure that correct software behaviors are implemented for safety systems. The proposed technique classifies emergent hazards into three categories and proposes a new process for analyzing interface hazards. However, the authors suggested that additional techniques must be developed and employed in conjunction with the interface hazard analysis technique to effectively address system hazards in an SoS.

Hazard analysis is the first step toward the safety analysis of a system. It is essential to identify potential hazards, assess their risks, and determine appropriate safety measures to mitigate or eliminate them. Therefore, Baumgart et al. [10] introduced an approach to identify potential hazards in SoS known as the HISoS approach. The HISoS approach systematically analyzes complex interactions between multiple systems during the early design phase, thereby allowing for the detection of previously unknown risks associated with these interactions. Furthermore, the HISoS methodology provides a useful approach to manage the complexity of designing interactions between multiple systems while ensuring safety. Finally, the authors mentioned that HISoS enables systematic analysis of the risks associated with emergent behavior that may arise in an SoS, which may not be visible through traditional single-system analysis.

Jung et al. [11] presented an approach for hazard analysis considering dynamic configuration uncertainty for SoS. The proposed approach involves creating a model that unfolds variability information using multiple system specifications and traceability analysis results. This model is then used to conduct a hazard analysis for an SoS. The authors used a case study from platooning system to demonstrate the practicality. However, this approach did not provide any tool support to automate the hazard analysis process in an SoS.

Alexander et al. [12] introduced a hazard analysis technique called SimHAZAN which employs multi-agent modeling and simulation to analyze hazards and investigates the impact of abnormal behavior of nodes in an SoS, providing the potential to uncover hazards that may be challenging or impossible to identify using traditional techniques. The proposed technique also outlines a structured approach to construct multi-agent models of SoS that begins with utilizing existing models in a reference architecture framework and concludes with developing simulation models. Furthermore, the authors provided a well-defined technique for constructing multi-agent models and conducting simulations bounded by estimated probabilities to produce comprehensive logs of simulated events. Moreover, they introduced a tool-supported analysis technique featuring machine learning and agent behavior tracking capabilities, which facilitates the identification of accident causes from the generated logs. However, this study has a scalability problem because it was applied to a small example.

III. Case Study - Mixed Traffic Automation for Underground Mine

This section investigates Mixed-traffic Automation in an Underground Mine (MAUM) from the mining and equipment domain. The MAUM consists of a tunnel connecting an underground mine with a dump area on the surface, as shown in Fig.2.

The operations in a mining tunnel typically involve a fleet of Autonomous and Semi-Autonomous Machines (ASAMs) as well as Manned Machines (MM). The goal of ASAMs is to efficiently transport ores from an underground mine to the surface. In contrast, MM (pickups and trucks) are supposed

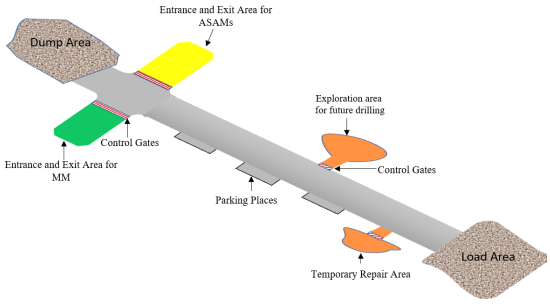

Figure 2. Abstract Representation of MAUM

to transport human operators and materials, e.g., concrete, up and down from the mine. Since the mixed mode traffic could lead to many hazardous scenarios, operations of ASAMS and MMs are performed in a highly controlled and mutually exclusive manner. These are typically done using gated (or restricted) autonomous operating zones, where humans or human-operated machined cannot enter while ASAMS are operating. However, such solutions are restrictive and do not allow the full automation potential and productivity gains.

The MAUM aims to permit mixed-traffic operations inside an autonomous operating zone to improve productivity. At the same time, we need to ensure such an opening up does not lead to safety concerns. For this, non-autonomous machines have specific access points (entrance/exit areas, depicted in green). The access point for ASAMs (entrance/exit areas) is shown in yellow. In addition, such access points have check-in/out gates (displayed in red) for maintaining access control of such machines in the operating zone. There are parking areas (depicted in grey rectangles) along the tunnel, a dedicated temporary repair area for machines, and a potential exploration area for future drilling areas.

The full potential of ASAMs in an underground mine can be achieved when they would be able to operate in autonomous mode in a particular operating zone. This automation can reduce the involvement of human operators and thus increase safety and productivity. The Traffic Management System (TMS) is used to manage these zones, where the mission for both autonomous and semi-autonomous fleets is set. The mission is planned based on factors such as environmental conditions, ongoing construction work, and worksite settings. Access control systems are also put in place to restrict human access to autonomous operating zones.

The MAUM is a good representative example of an SoS where ASAMs, and MM interact to archive productivity, performance, and safety. The work presented here is equally applicable in many similar contexts, such as warehouses with AGVs or construction sites or open pit mines, etc.

IV. Hazard Analysis using CompHAT

In this section, we apply CompHAT for hazard analysis by considering two scenarios from the MAUM case study.

Scenario 1: In MAUM, the ASAMs operate in a dedicated operating zone, and MM are restricted to that zone when ASAMs operate. However, when any MM violates this

requirement and enters the dedicated operating zone where ASAMs operate, the safety controller in MAUM sends an Emergency Stop (ES) command for all machines to avoid collision hazards. In summary, the collision hazard can arise when: 1) if ASAMs loss communication with TMS, the safety controller would not be able to send the ES command in case of an emergency, 2) if any ASAMs fail to localize the dedicated operating zone, it can go to the zone where MM operate or collide with other ASAMs.

Scenario 2: MM operate in a non-autonomous zone and are supposed to transport human operators and material. The mammed machines can be emergency vehicles that can enter the mine if necessary. We consider that a manned vehicle poses a collision hazard when: it operates in autonomous zones; enters a non-autonomous zone where an ASAM is operating in autonomous mode; ASAMs fail to communicate with the TMS or cannot be localized in the autonomous zone; MM fail to communicate with the TMS or cannot be localized in their operating zone.

When some safety issue occurs while the machines operate, an ES command is issued to stop ASAMs within the dedicated zone (or a specific set of- ASAMs in the zone). The safety controller also notifies operators in MM about such actions so they have sufficient information regarding the issue and will be able to avoid any potential collision.

In order to analyze hazards for MAUM using CompHAT, we first identified the constituent systems, i.e., ASAMS, MM, TMS, and safety controller, that make the MAUM (an SoS). In the next subsection, we apply $SoCPSTracer$ to MAUM and discuss the results.

A. Results and Discussion

This subsection discusses some representative examples of hazard identification and their propagation in the MAUM case study. In order to apply $SoCPSTracer$ to MAUM, we first analyzed hazards for each constituent system, i.e., ASAMs, MM, TMS, and safety controller using FTA, ETA, and FMEA. The generated hazard artifacts for each system are saved in the artifacts repository and used as input to the SoCPSTracer to analyze hazards for MAUM (an SoS). An excerpt of FPG for MAUM is shown in Fig. 3, where the red edges represent the countermeasure relationship, the yellow edges show an overlapping relationship, and the pink edges between nodes represent the influence relationship. The generated FPG for MAUM offers fault propagation within the components of constituent systems and between the constituent systems of an SoS (MAUM). For instance, $Broken\ of\ proximity\ sensor.[ASAMs.FTA\_2]$ is a fault that belongs to an ASAM that can lead to the $Proximity\ Sensor\ Failure.[ASAMs.FTA\_0]$ fault within ASAMs that can finally cause $Collision\ Hazard.[ASAMs.FTA\_0]$ if not adequately handled. This shows fault propagation within a constituent system.

Similarly, $Software\ not\ Updated.[TMS.FMEA\_2]$ fault from TMS can trigger $Software\ Failure.[TIMS.FMEA\_2]$ fault that can cause $ASAMs\ loss\ Communication\ with\ TMS.[MM.FTA\_1]$ fault in ASAMs, and finally, it can lead to $Collision\ Hazard.[ASAMs.FTA\_0]$. However, a safety action, i.e., $Update\ Software\ Periodically.[TMS.FMEA\_2]$ is suggested to take place in TMS to avoid the collision hazard. This shows fault propagation between the constituent systems of an SoS. The FGP can be a huge graph with hundreds of nodes, making hazard analysis difficult. Therefore $SoCPSTracer$ provides functionality where we can generate subgraphs from the main FPG. For instance, we want to see how $Incorrect\ Command.[TMS.FMEA\_2]$ fault impacts on other systems or components within the system, we double click on it, and it generates a subgraph for that particular fault, as shown in Fig. 4. The $Incorrect\ Command.[TMS.FMEA\_2]$ can lead to $Enter\ a\ Zone\ Where\ ASAMs\ Operate.\ [MM.FTA\_1]$ and $Misbehaviour\ of\ ASAMs.[ASAMs.FMEA\_0]$ fault, which can trigger $Collision\ Hazard.[ASAMs.FTA\_0]$. This is a typical example of fault propagation from one system to another system in an SoS. The incorrect command sent from the TMS can propagate to MM and ASAMs and pose a collision hazard.

From the subgraph (Fig.4), we can see the backward traceability of any fault. For instance, we again double-clicked on the $Collision\ Hazard.[ASAMs.FTA\_0]$ to see what faults contributed to it. This generates another subgraph, as shown in Fig. 5, where we see a number of potential faults that can potentially trigger the collision hazard in ASAMs.

Hazard Analysis for SoS using CompHAT (RQ1): From our experience, while analyzing MAUM for potential hazards, we see that $SoCPSTracer$ supports effectively in identifying potential faults within the constituent systems and between the constituent systems of an SoS as shown in Fig. 3. Furthermore, it also shows what kind of countermeasures are available in the system and how we can apply them to counter similar faults in other constituent systems. For Instance, $TriggerES.$[SC.FMEA_3] can be used to counter $Collision\ Hazard.[ASAMs.FTA\_0]$ in ASAMs and $Collision\ Hazard.[MM.FTA\_1]$ in MM. Fig.3 also shows the overlapping relationship between $Collision\ Hazard.[ASAMs.FTA\_0]$ in ASAMs and $Collision\ Hazard.[MM.FTA\_1]$ in MM. Meaning that both are the same consequences of different faults. This relationship helps us to apply a safety guard to a fault with the same consequence but no countermeasure. Therefore, we can argue that CompHAT can be used to enlist all the constituent systems (ASAMs, MM, Safety Controller, and TMS) and analyze them to identify potential faults in a network of SoS using SoCPSTracer.

SoCPSTracer and Interaction Hazards (RQ2): As discussed above, in Fig. 3, we can see the identification of interaction-related faults in the MAUM. Particularly, from Fig. 4, we see how incorrect commands sent from TMS lead to other faults in MM and ASAMs.From Fig. 4 and Fig. 5, we also see how we can trace faults forward and backward in the FPG. Therefore, it is strong evidence that $SoCPSTracer$ can

Figure 3. An excerpt of FPG for scenarios 1 and 2 from MAUM

be used effectively in identifying faults within the components of a constituent system and between constituent systems of an SoS. The aim of this SoS network hazard analysis is to connect these links with interaction-related hazards and ensure safety within a network of an SoS.

V. Conclusion

This paper presents a case study from the mining and equipment domain where ASAMs and MM operate to extract ores and transport them to the surface. The ASAMs are operated in a fleet and are integrated into the production processes, including operators on the site and other human-operated machines. Such a system can be viewed as an SoS with complex interactions between the constituent systems involved. In this paper, we apply the hazard analysis technique to a representative SoS, i.e., MAUM, to see whether it is applicable in identifying hazards, especially interaction-related ones, in an efficient manner, and we share our experiences and insights gained from this effort. Our findings from the hazard analysis and results clearly demonstrate the applicability of $SoCPSTracer$ in the context of SoS and provide

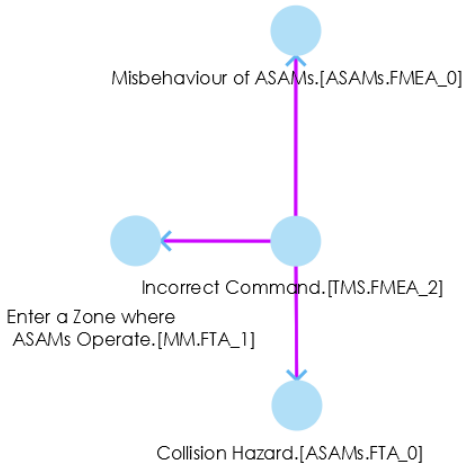

Figure 4. Fault propagation for $Incorrect\ Command.[TMS.FMEA\_2]$

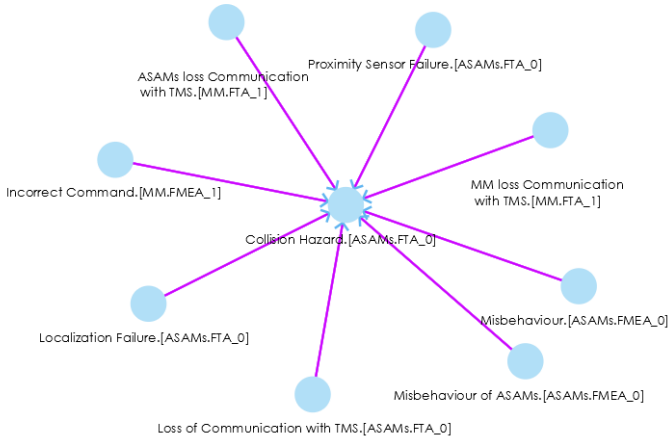

Figure 5. Back traceability of $Collision\ Hazard.[ASAMs.FTA\_0]$ in impact analysis.

the tool support to identify faults between the components of constituent systems and between the constituent system of an SoS.

The $SoCPSTracer$ tool is limited to FTA, FMEA, and ETA. This is because these three hazard analysis techniques are the only ones that can be used as inputs for the relationship detection algorithm. Therefore, it should be noted that $SoCPSTracer$ cannot be used for hazard analysis artifacts created with other hazard analysis techniques. However, extensions to the $SoCPSTracer$ can be made to support other hazard analysis techniques. The $SoCPSTracer$ can be used in any application domain aiming to track faults in collaborative systems. The severity of faults may vary across different application domains.

In the future, we plan to use the results of this hazard analysis to derive/revise safety requirements.

ACKNOWLEDGMENT

This research has been partially supported by the Vinnova funded ESCAPE-CD (Efficient Safety for Complex Autonomous Production Environments - Concept Design) Project and the SSF funded DAISY (Dependable System of Systems) Project.

REFERENCES

[1] 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.

[2] M. Hussain, N. Ali, Y. Kim, and J.-E. Hong, "Analyzing safety in collaborative cyber-physical systems: A platooning case study," in *SOFTENG 2021 The Seventh International Conference on Advances and Trends in Software Engineering*, 2021, pp. 16–22.

[3] R. Bell, "Introduction to iec 61508," in *Acm international conference proceeding series*, vol. 162. Citeseer, 2006, pp. 3–12.

[4] ISO26262-1, "Road vehicles — functional safety — part 1: Vocabulary," https://www.iso.org/standard/68383.html, 2018, [Online; accessed 20-February-2023].

[5] N. Ali, M. Hussain, and J.-E. Hong, "Analyzing safety of collaborative cyber-physical systems considering variability," *IEEE Access*, vol. 8, pp. 162 701–162 713, 2020.

[6] D. Horn, N. Ali, and J. E. Hong, "Towards enhancement of fault traceability among multiple hazard analyses in cyber-physical systems," in *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2. IEEE, 2019, pp. 458–464.

[7] N. Ali, M. Hussain, and J.-E. Hong, "Safesocps: a composite safety analysis approach for system of cyber-physical systems," *Sensors*, vol. 22, no. 12, p. 4474, 2022.

[8] ISO, "Systems and software engineering — taxonomy of systems of systems," https://www.iso.org/standard/71957.html, 2019, [Online; accessed 07-February-2023].

[9] P. J. Redmond, J. B. Michael, and P. V. Shebalin, "Interface hazard analysis for system of systems," in *2008 IEEE International Conference on System of Systems Engineering*. IEEE, 2008, pp. 1–8.

[10] S. Baumgart, J. Fröberg, and S. Punnekkat, "Analyzing hazards in system-of-systems: Described in a quarry site automation context," in *2017 Annual IEEE International Systems Conference (SysCon)*. IEEE, 2017, pp. 1–8.

[11] S. Jung and J. Yoo, "An approach for hazard analysis of multiple-cooperative systems considering dynamic configuration uncertainty," in *2022 29th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2022, pp. 279–288.

[12] R. Alexander and T. Kelly, "Supporting systems of systems hazard analysis using multi-agent simulation," *Safety science*, vol. 51, no. 1, pp. 302–318, 2013.