Contradictions of identification of safety and security requirements for industrial cyber-physical systems

Article

Accepted Version

Sun, Z., Ding, Y., Pei, K. and Yang, S.-H. ORCID: https://orcid.org/0000-0003-0717-5009 (2024) Contradictions of identification of safety and security requirements for industrial cyber-physical systems. IEEE Internet of Things, 11 (7). pp. 12033-12057. ISSN 2327-4662 doi: 10.1109/JIOT.2023.3332128 Available at https://centaur.reading.ac.uk/114013/

To link to this article DOI: http://dx.doi.org/10.1109/JIOT.2023.3332128

Publisher: IEEE

Contradictions Identification of Safety and Security Requirements for Industrial Cyber-Physical Systems

Zhicong Sun, Yulong Ding, Ke Pei, and Shuang-Hua Yang, *Senior Member, IEEE*,

***Abstract*—Industrial cyber-physical systems (iCPSs) are the backbone of the fourth industrial revolution, facing more safety and security (S&S) challenges compared to traditional industrial systems. One of the most critical challenges is the collaborative analysis of S&S. Considerable efforts have been made towards integrating S&S and resolving their contradictions. However, a significant research gap remains regarding the accurate definition of contradictions in S&S requirements, along with an identification methodology. This study presents a systematical methodology to address this challenge. We propose two sufficient conditions that result in contradictions and provide algorithms to help their identification. Additionally, three measures have been proposed to reduce the difficulty of contradictions identification, including a conceptual model for iCPSs with S&S objectives to constrain objects and interactions within the model, a method for unifying the elicitation of S&S requirements, and a requirements template for coordinating the representation of S&S requirements. To provide insight into the operations of the methodology, we demonstrate its application in a smart factory. The results show that this approach can effectively identify the hidden contradictions in S&S requirements.**

***Index Terms*—Contradictions identification, industrial cyber-physical systems, safety, security.**

I. Introduction

INDUSTRIAL cyber-physical systems (iCPSs) refer to the next generation of industrial systems that adeptly merge informational and physical spaces via control, communication, and computation methods. As crucial elements underpinning the fourth industrial revolution [1], iCPSs utilize networks to transpose industrial functions into the informational space, subsequently executing real-time monitoring, control, and intelligent decision-making [2]. Analogous to CPSs centered upon other physical processes, iCPSs are structured in a three-tier hierarchy [3]: the unit level (e.g., a manipulator in an automatic guided vehicle (AGV)), the system level (e.g., an AGV incorporating a sensor, manipulator, and other unit-level iCPSs), and the system-of-systems (SoS) level (e.g., a system involving multiple AGVs). However, iCPSs give greater attention to real-time control, interoperability of components, and dynamic system deployment. Such characteristics should be given careful consideration throughout the entire lifecycle of iCPSs, encompassing requirement specification, system design, operation, and maintenance.

iCPSs face more substantial challenges compared to conventional industrial systems as being connected to the expansive cyberspace. One of the most critical challenges is related to the collaborative analysis of safety and security (S&S). Safety refers to freedom of unacceptable risk related to non-intentional events such as malfunctions, component failures, explosions, and fires [4], while security is defined as the absence of unacceptable risk when restricting the concept of risk to intentional acts like cyber-attacks committed by intelligent adversaries [5]. Historically, S&S were typically analyzed independently and considered as separate disciplines [6]. However, with the emergence of iCPSs, the deep integration of industrial equipment and the cyberspace has provided a series of sophisticated cyber-attacks (such as Stuxnet [7] and BlackEnergy [8]) with opportunities to exploit security vulnerabilities and create severe safety incidents. This urges academics to recognize the necessity of scrutinizing the reciprocal impact between S&S [9], [10], with both facets necessitating simultaneous consideration during system design [6], [10]–[13].

Considerable efforts have been made towards the co-analysis of S&S within iCPSs. The majority of research within the community seeks to foster the mutual reinforcement between S&S. These research endeavors span an array of topics, including risk identification [14]–[18], risk assessment [19]–[24], risk management [15], [25], lifecycle integration [5], and requirements engineering [13], [26]–[28], all in the context of S&S co-analysis. Contrastingly, research devoted to addressing the contradictions between S&S in iCPSs is relatively sparse, notwithstanding the consistent emphasis on its criticality [6], [10], [23], [29]–[33]. The most representative studies include [33]–[37]. Specifically, in 2008, Novak and Treytl [34] presented a initial strategy designed to resolve conflicts between S&S at the requirement and function levels. In 2009, Sun *et al.* [35] provided instances of contradictions between S&S and an initial framework for their resolution. In 2015, Gu *et al.* [36] gave a rough definition of the contradiction between S&S requirements based on the lens of set theory, and proposed a initial idea to identify such conflicts through requirements decomposition. In 2021, Menon and Vidalis [33] identified areas of potential contradiction in the techniques, evaluation methods, and cultures of S&S, explaining these

(Corresponding author: Shuang-Hua Yang)

Z. Sun is with the Department of Civil and Environmental Engineering, Hong Kong Polytechnic University, Hong Kong SAR, China, and also with the Institute of Advanced Computing and Digital Engineering, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China (email: zhicong.sun@connect.polyu.hk).

Y. Ding is with the Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Internet, SUSTech, Shenzhen 518055, China, and also with the Department of Computer Science and Engineering, SUSTech, Shenzhen 518055, China (email: dingyl@sustech.edu.cn).

K. Pei is with the RAMS lab and TTE lab in Huawei's Munich Research Center, Munich, Germany (email: peike@huawei.com).

S. Yang is with the Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Internet, SUSTech, Shenzhen 518055, China, and also with the Department of Computer Science, University of Reading, UK (email: yangsh@sustech.edu.cn).

conflict zones with real-world examples. In 2022, Agbo and Mehrpouyan [37] identified contradictions in S&S boundary conditions as situations where the provision or absence of a control action results in hazards or threats, and proposed a method for analyzing and resolving these contradictions. In summary, these studies have commenced exploring the contradictions within S&S by undertaking initial measures to resolve them. Nevertheless, a significant research gap persists - the lack of a methodology to identify S&S contradictions comprehensively, which is an essential preliminary step for contradictions resolution. More specifically, it is crucial to comprehensively identify contradictions at the requirements level, given that contradictions in iCPSs frequently arise from disagreements in S&S requirements [34], [37].

Indeed, the primary motivation behind this study is to address the aforementioned research gap. Specifically, our purpose is to resolve two fundamental limitations in the prevailing literature: firstly, the definition of contradiction is inaccurate and incomplete, and secondly, the elicitation and representation of S&S requirements in previous works fail to provide a suitable context for identifying contradictions, making it difficult to detect inconspicuous contradictions (e.g., contradictory actions caused by different conditions).

To address these two critical issues, we first systematically consider the objective of identifying contradictions in S&S requirements and divide the process of achieving this goal into three interrelated steps: requirements elicitation, requirements representation, and contradictions identification. Subsequently, we refer to the related work from software requirements engineering and propose a systematic methodology. The approach provides an appropriate requirements elicitation and representation process that that creates a stronger foundation for identifying contradictions. Moreover, it redefines the definition of contradiction and supports finding deep-hidden contradictions. Finally, we illustrate the methodology via a case study of a smart factory composed of multiple AGVs. The main contributions of this work are summarized as follows:

- An iCPSs conceptual model is proposed in this work with some widely recognized S&S objectives that are adopted and redefined to constrain the objects and interactions in the model. This model simplifies the complex iCPSs according to the needs of system design, thus limiting the analysis scope of S&S requirements.
- The causes-phenomena-effects analysis (CPEA) method is proposed to unify the elicitation of S&S requirements. CPEA assists in identifying contradictions for two main reasons. First, CPEA provides identical information and a similar perspective for both S&S requirements analysis, which reduces ambiguity in the expression of requirements. Secondly, CPEA dives into the causal information of each risk and considers both S&S factors, resulting in a set of comprehensive and detailed requirements that are conducive to discovering deep-hidden contradictions.
- A requirement template with constrained natural language patterns is designed for expressing both S&S requirements. The template reduces the difficulty of contradictions identification while ensuring comprehensibility. In addition, requirements expressed using this template have the potential to be formalized and used for system verification and even contradictions elimination.
- The concept of contradictions in S&S requirements is defined, and two sufficient conditions that result in contradictions are proposed. Additionally, algorithms are provided to judge whether these conditions are satisfied or not.

The remainder of this paper is structured as follows. Section II reviews the related work and discusses their limitations. Section III introduces the five foundations we proposed for identifying contradictions in S&S requirements. Section IV provides a systematic methodology consisting of four phases for identifying contradictions. Section V presents the case study of the proposed approach to the smart factory compound of AGVs and discusses the results. Section VI discusses the advances of our work compared with related studies and limitations to be overcome. Finally, Section VII highlights the most significant contribution of our work, the impact on the community, and presents challenges for future research.

II. Related Work

As discussed in Section I, we divide the process of identifying contradictions in S&S requirements into three significant steps: elicitation of S&S requirements, representation of S&S requirements, and contradictions identification. In this section, we introduce and discuss the related work that is specifically relevant to each of these steps.

A. *Elicitation of safety/security requirements*

Eliciting S&S requirements is a crucial prerequisite for identifying contradictions, as it determines the consistency of requirements expression to a significant extent and consequently influences the level of difficulty in identifying contradictions.

Typically, S&S are considered separately, and their requirements are derived independently. One of the most common methods to derive safety/security requirements is based on the information collected by risk analysis techniques such as HAZOP [38], FMEA [39], and STPA [40] for safety and ATA [41], FMVEA [42], and SPTA-Sec [43] for security. Another type of methods is specifically designed for safety/security requirements analysis, including those based on patterns [44], [45], ontology [46], [47], and model [48]–[50]. However, separate elicitation often pays little attention to the interactions between S&S, making requirements not comprehensive. Additionally, this approach often leads to variations in the expression of S&S requirements, which in turn makes it more challenging to identify contradictions.

Recently, a few studies have paid attention to co-analysing S&S requirements and made some attempts to do so [6]. Brunner *et al.* [27] proposed a domain-independent method in which integrates S&S requirements into a model, so as to support certification processes during the design and run-time phases of CPS. Kavallieratos *et al.* [13] proposed an approach called SafeSec Tropos, which is based on the Secure Tropos and STPA method, for jointly eliciting S&S requirements. SafeSec Tropos elicits requirements based on S&S objectives, which facilitates the prioritization of requirements, the communication of requirements to the relevant stakeholders. Additionally, it provides similar documentation structures for both S&S requirements, making it easier to detect requirements conflicts. However, although these methods provide a more comprehensive requirements for protecting the system, they may elicit requirements with ambiguity (such as describing the same object in different words) due to the use of different models to analyse S&S, which may make it difficult to identify contradictions.

B. Representation of safety/security requirements

There are five kinds of approaches for representing requirements: formal, natural-language, semi-formal, automata/graphs, and seamless [51].

Formal methods, such as FORML-L [52], use discrete mathematics to describe the function and architecture of a system, which helps with logical reasoning and validity of requirements [53]. Nevertheless, this representation is rigorous but complex, requiring both stakeholders and requirements engineers to receive professional training and closely collaborate.

Natural language approaches, such as EARS [54] and RELAX [55]), are the most commonly used methods using human language or structured statements with format restrictions to express requirements. This form of representation is comprehensible for stakeholders, but it may lack precision and verifiability.

Graph-based approaches, such as Petri nets [56], [57], rely on automata or graph theory and usually represent the state transition and dependency of the system by graphs. Though this type of representation is effective in describing the overall system scheme, it incurs high learning costs. In addition, some graphics used in these approaches can only be comprehended by experts, and limited methods are available to support the verification process.

Seamless approaches, such as SOOR [58], use an identical programming language to combine requirements with other software tasks, including design and development. This technique can achieve the seamless integration of requirements, design, and development, but it lacks support for verification, and further research is still necessary to determine its effectiveness.

Semi-formal approaches, like real-time specification patterns [59] and semi-formal requirement modelling pattern [60], incorporate both the comprehensibility of informal methods (e.g., natural language and graph-based approaches) and the rigor of formal methods. Each of these approaches expresses requirements using rigorously defined syntax and semantics accompanied by corresponding representations typically in natural language and formal notation. However, these approaches are usually formed by combining different approaches, which may result in the lack of systematic symbolic representation and consistent symbolic semantics.

C. Contradictions identification for safety and security of iCPSs

There are relatively few existing studies aiming at identifying contradictions between S&S, let alone detecting contradictions in S&S requirements.

Sun *et al.* [35] proposed a formal method to identify contradictions between the S&S of CPS. This approach uses a formal rewriting logic language to model the S&S spaces. Each space is defined as a four-tuple compound of model, propositions, assumptions, and requirements. Based on this definition, the problem of identifying S&S contradictions is simplified as finding a model with opposite S&S requirements under the same assumption. However, this simplification does not always hold up under scrutiny because contradictions between S&S may not only occur at the level of requirements but may also exist between measures elicited by the same requirement. In addition, this work suffers from other issues. Firstly, this work does not provide a way to identify requirements contradictions. Next, this work does not supply a requirement elicitation approach, which may cause ambiguity when expressing the same requirement. Lastly, the formal method for requirements representation cannot provide good readability.

Gu *et al.* [36] proposed a goal-based method to extract interdependent S&S requirements for industrial control systems and resolve their conflicts based on severity. This study defines the conflict between S&S requirements as occurring when an action in a safety/security requirement is detrimental to the security/safety goals. It represents each S&S requirement by natural language and decomposes it into four elements, i.e., condition, subjective, action, and goal. Based on the representation, conflict can be detected by traversing decomposed requirement items and locating requirements with the same condition and subject but opposite actions. Although this method considers the elicitation of requirements, it uses the safety analysis method (i.e., HAZOP) to elicit security requirements, making them only safety-related and therefore one-sided. In addition, this work does not equip a method to decompose the requirements into its specified four elements, and the use of natural language to express the requirements also increases the difficulty of decomposition.

D. Summary of the related work

Based on the above discussion, we summarized the main characteristics, strengths and weaknesses of the presented related work in Table I. In addition, we have identified two key points for identifying contradictions. First, a unified method for analysing S&S requirements, which comprises a system model for analysing S&S, a method for eliciting S&S requirements in a unified manner, and a template for representing them, is pivotal to provide a strong premise for identifying contradictions as it helps reduce ambiguity in S&S requirements. Specifically, the system model should be able to represent iCPSs in a way that is suitable for system design and S&S analysis. The elicitation method is responsible for ensuring that S&S factors are thoroughly taken into account while reducing ambiguity in S&S during the process. The requirements representation should be comprehensible, user-friendly, able to demonstrate the characteristics of iCPSs, and have the potential to be verified if possible. Secondly, we must provide a comprehensive definition of S&S requirement contradictions, and it is critical to provide algorithms for

TABLE I
SUMMARY OF THE RELATED WORK

Phases	Category of methodologies	Characteristics	Advantages	Limitations
Elicitation of S&S requirements	Separate elicitation methods (e.g., patterns-based [44], [45], risk-analysis-based [39], [40], [42], [43], ontology-based [46], [47], and model-based [48]–[50])	Consider and elicit only safety or security requirements	Comprehensible	Obtained requirements cannot comprehensively protect the system
	Joint elicitation methods (e.g., Brunner's work [27] and SafeSec Tropos [13])	1. Consider and elicit both S&S requirements 2. Use different system models to analyse S&S 3. Use different methods to derive S&S requirements	Obtained requirements can comprehensively protect the system	Result in ambiguity of S&S requirements
Representation of S&S requirements	Formal methods (e.g., FORML-L [52])	Use discrete mathematics to describe the function and the architecture of a system	Help logical reasoning and requirement validity	Require users to undergo time-consuming training and close collaboration
	Natural language methods (e.g., EARS [54]and RELAX [55])	Use natural language or structured statements with format restrictions to express requirements	Comprehensible and user-friendly	Lack of precision and verifiability
	Graph-based methods (e.g., Petri nets [56], [57])	Use graphs to represent the state transition and dependency of a system	Describe the scheme of a system comprehensively	High learning cost
	Seamless methods (e.g., SOOR [58])	Use a programming language to represent requirements	Connect requirements, design, and development seamlessly	Hard to verify, and its effectiveness needs to be further investigated
	Semi-formal methods (e.g., Konrad's work [59] and Jue's work) [60]	Combine a formal method and a natural language method	Comprehensible and rigorous	Lack of systematic symbolic representation and consistent symbolic semantics
Contradictions identification	Sun's work [35]	Identify contradictions between S&S by a formal method	Convert the S&S contradictions to the S&S requirement contradictions	Lack a method to identify requirements contradiction
	Gu's work [36]	1. Identify requirement contradictions between S&S by a goal-based method 2. Use HAZOP to elicit S&S requirements	Provide an initial idea for identifying S&S requirement contradictions	1. Incomprehensive and inaccurate definition of contradiction 2. Requirements are safety-related and therefore one-sided 3. Lack a method to preprocess requirements before contradictions identification

identifying contradictions based on the proposed unified S&S requirements analysis framework.

III. Proposed Foundations

iCPSs act as an interactive medium between the physical and information worlds through a workflow that comprises industrial processes, networking, actuation, and computing. This workflow processes information and physical objects, occurring within the system and between iCPSs and the external physical and cyber environments. In this context, this work proposes an iCPSs conceptual model that defines the types of objects and their interactions. Furthermore, S&S objectives are introduced to regulate each object and interaction. And we argue that the fundamental reason for the insecurity and unsafeness of iCPSs is their failure to comply with these basic objectives. Building on this argument, we propose CPEA as a means to unify the elicitation of requirements for S&S, which reduces the ambiguity between these requirements and supports the sharing of risk information. Complementary, a template with constrained natural language patterns is provided for expressing S&S requirements. Lastly, the contradictions in S&S requirements are discussed and redefined. This section presents the five theoretical foundations, which we have proposed in this work.

A. Conceptual Model of iCPSs

The conceptual model of iCPSs presented in this study is based on established research in the field regarding the architecture of iCPSs [61]–[68]. In essence, iCPSs are recognized as the fusion of the physical and cyber worlds [15]–[18], [65], [68]–[71], consisting predominantly of computational and physical processes [61]–[66], [70], where interactions between components are facilitated through the flow of information and energy [15]–[18], [68], [71]. As such, in the presented conceptual model, iCPSs mainly includes five categories of objects: *system objects, physical objects, cyber objects, information objects, and energy objects*, along with two types of interactions, namely *physical interactions and information interactions*. As a support, we provide an Object-energy-information (OEI) diagram to describe them. The ontology of OEI diagram is shown in Fig. 1, and an example of OEI diagram is shown in Fig. 2. A detailed introduction to the objects, interactions and the example of OEI diagram is provided as follows:

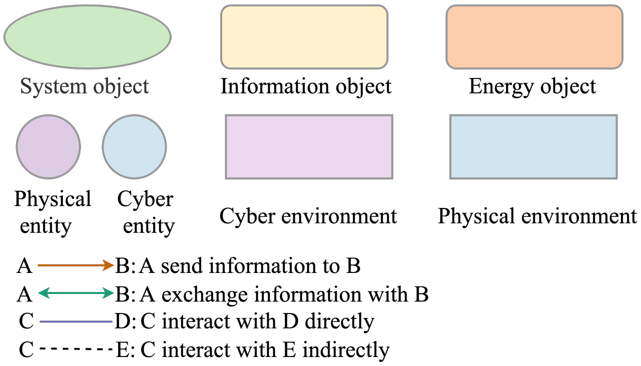
System object
Information object
Energy object
Physical entity
Cyber entity
Cyber environment
Physical environment
A → B: A send information to B
A ↔ B: A exchange information with B
C —— D: C interact with D directly
C - - - - - - - E: C interact with E indirectly

Fig. 1. Ontology of the object-energy-information diagram.

System objects are components at a certain level of the iCPSs to be developed (iCPSs in a narrow sense). As shown

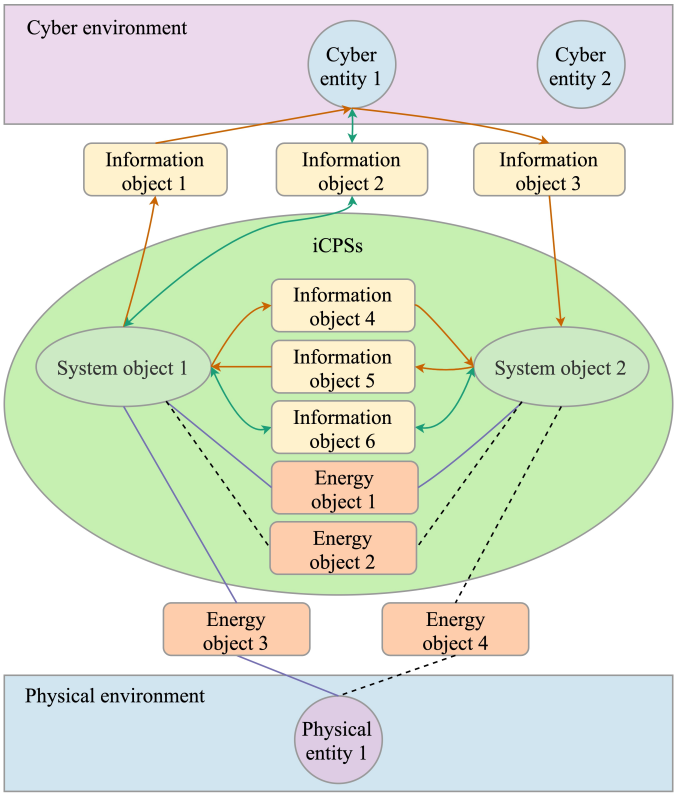

Fig. 2. An example of objects and interactions represented by the OEI diagram.

in Fig. 2, the iCPSs to be developed is represented by a large green oval, while system objects, namely system object 1 and system object 2, are represented by small green ovals. A system object can be abstract, like a smart factory system, or concrete, like an AGV. It can be either a software, such as a path planning module, or a hardware, such as a laser radar. System objects play a crucial role in defining the scope of a project, as minimum system objects determine the level at which modelling analysis ceases. A minimum system object is defined as a system object that cannot be divided or whose operational process falls outside the scope of research. It is also known as an non-decomposable system object, which does not need to be modelled, while other decomposable system objects must be modelled during analysis. For example, an acceleration sensor may be considered a minimum system object if its method of obtaining acceleration is not of interest to system designers. It is important to note that multiple minimum objects can exist at different levels within a system, depending on the designer's focus. For instance, in a smart factory with a commercial and multi-AGV system at the same level, the designer may focus on the industrial processes of the multi-AGV system and not the commercial system components. In this case, the commercial system is a minimum system object, while the multi-AGV system is divided into several AGVs and a control center, which are also minimum system objects. Thus, minimum system objects can exist at different levels within a system.

Physical objects include the physical environment where iCPSs are situated, such as storehouses, or the physical entities coexisting in the environment, such as human beings and goods in the storehouses. As shown in Fig. 2, the physical environment is represented by a light blue rectangle, and the physical entity (i.e., physical entity 1) is represented by a pink circle.

Cyber objects are either the cyber environment to which iCPSs are connected, such as the Internet, or cyber entities within the cyber environment, such as hackers. As shown in Fig. 2, the cyber environment is represented by a pink rectangle, and the cyber entities (i.e., cyber entity 1 and cyber entity 2) are represented by light blue circles.

Information objects refer to the data that is exchanged between two objects, such as sensor data. As shown in Fig. 2, information objects, i.e., information object 1-6, are denoted by yellow rectangles. They can be either data transmitted between system objects, such as information object 4-6 in Fig. 2, or data involved in the interaction between system objects and cyber entities, like information object 1-3 in Fig. 2. Yet, it should be noted that the OEI diagram represents only those information objects known to the system designer as the iCPSs to be developed may expose to an open cyber environment, where potential cyber entities like hackers may access the system objects and the communicated data is unknown. For instance, cyber entity 2 in Fig. 2 may exchange data with system objects inside the iCPSs, but the system designer cannot determine the content, thus there is no information object connected between cyber entity 2 and any system objects in the diagram.

Energy objects exist in the conversion of energy between system objects and physical objects, such as kinetic energy, potential energy, thermal energy, electrical energy, and more. However, we typically do not specify what these energy objects are and do not represent them in the OEI diagram, except in special cases such as those involving important physical or chemical processes, because we are more concerned with the information flow in the system. For example, when AGV picks up goods, it engages with specific energy objects. However, we are more concerned with the potential hazards and S&S requirements to deal with them, rather than what the involved energy objects are. As shown in Fig. 2, energy objects, i.e., information object 1-4, are denoted by orange rectangles. Similar to information objects, energy objects exist between the system objects, such as energy object 1 and energy object 2 in Fig. 2, as well as between system objects and physical entities, such as energy object 3 and energy object 4.

Information interactions refer to the processes in which objects exchange data (i.e., information objects) with one another. For instance, the temperature sensor sending temperature data to the control unit depicts an information interaction. Information interactions can be categorised into three types based on information flows: receiving, transmitting, and both. AGVs' receipt of specific task instructions involves information receiving. The SCADA system's transmission of collected data to a remote server for analysis involves information transmission. AGVs sharing data to negotiate tasks involves both information transmission and reception. In the example of OEI diagram shown in Fig. 2, information interactions involving either receiving (e.g., system object 2 receiving information object 4) or transmitting (e.g., system object 1 transmitting information object 4) are represented by lines with single arrows pointing toward the receiver of the data, while information interactions that involve both receiving and transmitting (e.g., system object 1 exchange information 6 with system object 2) are represented by lines with arrows pointing in both directions.

Physical interactions refer to the processes in which objects interact physically with each other, along with some energy conversion. Physical interactions can occur through direct contact between objects, for instance when a robotic arm grasps goods or when unwanted collisions happen between AGVs. Indirect physical interactions can also take place, as seen with laser radar scanning the environment with laser beams: while the laser radar doesn't physically touch the environment it scans, the laser beams it emits make direct contact with the environment. Another instance of indirect physical interactions is when an AGV rests grasped goods on a shelf: the AGV and the shelf don't experience direct physical contact, but rather, they engage in indirect physical contact through the goods placed on the shelf. Fig. 2 shows dashed lines representing indirect physical interactions, such as the interaction between system object 1 and physical entity 1, and solid lines representing direct physical interactions, like the interaction between system object 2 and physical entity 1.

B. Safety and Security Objectives

S&S requires that objects and interactions within the iCPSs conceptual model must abide by specific requirements set according to the S&S objectives, otherwise iCPS may be exposed to unacceptable risks. In this context, this study incorporates several widely acknowledged S&S objectives and adapts them to align with the unique characteristics of the proposed iCPS model, aiming to facilitate the elicitation of S&S requirements.

The safety objectives for iCPSs and their definitions in this study are presented in Table II. These objectives presented in this study were initially proposed by Kavallieratos et al. through a comprehensive review of prior literature [13]. Controllability, Integrity, Availability, and Resilience establish safety objectives that require the normal operation of system objects in iCPS, as well as the integrity and accessibility of data. Operability, Observability, and Quality of Service (QoS) ensure that communication and data in iCPSs remain observable and within predefined ranges, enabling the timely detection of risk sources. In the event of incidents, Fault Tolerance, Graceful Degradation, and Survivability outline safety response objectives at different levels of functional degradation within the system.

The security objectives and their corresponding definitions for iCPSs in this study are outlined in Table III. These objectives have been widely recognized by scholars in the community. They include the Parkerian Hexad [72], along with the addition of Non-Repudiation proposed by Kavallieratos et al. [13]. The Parkerian Hexad comprises six essential security objectives, namely Confidentiality, Possession, Integrity, Authenticity, Availability, and Utility. These objectives are considered more comprehensive than the CIA triad (Confidentiality, Integrity, and Availability) [73]–[75]. In this study, the security objectives of Confidentiality, Integrity, Availability, and Authenticity govern the handling of data and interactions within iCPSs, ensuring that only authorized objects can access, modify, and control them while preventing unauthorized access and modifications. Utility ensures that information objects and interactions in iCPSs serve their intended purpose. Possession and Control mandates that information objects and interactions in iCPSs possess the capability to resist unauthorized intrusions. Non-Repudiation is included as a security objective due to its reflection of the systemic nature of the analysis and its contribution to a more comprehensive study of network security and data security [13]. This objective ensures that objects in iCPSs cannot deny their responsibilities, such as in transmitting data.

S&S objectives define the desired state and functionality that system objects and information objects in iCPSs should attain, aiding in requirement elicitation. By identifying safety-critical objects and ensuring their adherence to these objectives, we can derive the corresponding S&S requirements. The detail process of eliciting requirements will be introduced in Section IV-C. However, it is important to clarify that we do not assert the absolute comprehensiveness of these S&S objectives, as the field of security and safety science is a dynamic discipline where achieving comprehensive coverage is often considered challenging and complex. Additionally, investigating comprehensiveness falls outside the scope of our research. Nevertheless, we remain open to the integration of more comprehensive objectives that may be proposed in future research in order to enhance the overall comprehensiveness of the S&S requirements.

C. Causes-phenomena-effects analysis

The occurrence process of each risk source, primarily referring to hazards and threats in this paper, comprises three event types: causes, phenomena, and effects. In this study, the proposed CPEA examines these events and is considered as the causal theoretical model for unifying the elicitation of S&S requirements. CPEA relies on the iCPSs conceptual model and S&S objects previously presented. The detailed analysis process for CPEA is as follows.

Initially, it is essential to identify *safety-critical objects*. CPEA considers system and information objects with high criticality or highly associated with risk sources as safety-critical objects and recommends two methods to determine them. The first approach involves using criticality analysis techniques, such as the one suggested in NISTIR 8179 [76], to ascertain the criticality of an object. Alternatively, by using risk assessment methods, such as FMVEA [77], one can determine the object's assessment values that relate to risk. Objects that exceed acceptable levels of criticality or risk-related assessment values are considered safety-critical. However, this research does not propose a particular approach for criticality assessment as it falls beyond the scope of this study. The second method relies on industry standards, historical data, expert experience, and the results of risk identification methods

TABLE II
SAFETY OBJECTIVES.

Safety objectives	**Definition**
Controllability	The interaction or object shall be in the desired state and able to handle hazardous events during the operations.
Integrity	The information object shall be complete.
Availability	The information object shall be available when the recipient needs it.
Resilience	The system object shall be able to absorb any disturbance caused by faults.
Operability	The information object shall be within the constraints.
Observability	The system object shall be able to determine the state of interaction to enhance the situational awareness.
Quality of Service (QoS)	The information object should arrive in time and serve its purpose to perform the necessary functions and produce the messages that are needed.
Fault Tolerance	The object and interaction should continue to be operational in a failure.
Graceful Degradation	The object and interaction should be able to maintain possibly limited but still safe functionality.
Survivability	The object and interaction shall be able to maintain the operations at some predefined acceptable level.

TABLE III
SECURITY OBJECTIVES.

Security objectives	**Definition**
Confidentiality	The information object and interaction shall be protected against unauthorized access.
Integrity	The information object and interaction shall be protected against unauthorized modifications or manipulations.
Availability	The information object and interaction shall be available to authorized objects when requested by such objects.
Authenticity	The management, configuration, and operation of the information object and interaction shall be performed by authorized objects.
Utility	The information object and interaction shall be useful.
Possession and Control	The information object and interaction shall be protected against the possibility that confidential data be possessed or controlled by unauthorized objects.
Non-Repudiation	The system object and information object shall not refute responsibility.

to determine which objects are associated with hazards and threats and thus considered safety-critical. While the first method offers greater accuracy, the second approach offers more convenience. Ultimately, the choice of the best method depends on the specific analysis requirements and practical scenario. Selecting the appropriate method ensures accurate conclusions and meaningful recommendations.

Once the safety-critical objects have been identified, more comprehensive risk information can be obtained. First, we need to determine the *system level of the risk*, namely the system level of safety-critical objects involved for every risk source. Afterward, the ecosystem model containing the safety-critical object must be determined and considered as the *risk-related model*. In this model, objects that directly interact with the safety-critical object are tightly associated with the risk source and are appropriately considered as the *risk-related objects*. Interactions between the risk-related system objects and the safety-critical object are named as the *risk-related interactions*, accordingly.

Following the acquisition of the risk information, CPEA identifies the objects and interactions associated with the causes, phenomena, and effects of each risk source.

1) ***Causes analysis****:* CPEA considers each risk source caused by the failure of risk-related interactions. If the safety-critical object is a system object, the risk results from the failed information and energy interactions, i.e., the uncontrolled flow of information and energy [78]. If the safety-critical object is an information object, the risk is caused by the uncontrolled flow of information involved in the risk-related interactions.

2) ***Phenomena analysis****:* CPEA believes that every risk source produces an abnormal system state that is manifested in the corresponding risk-related information objects. When the safety-critical object is a system object, the transmitted information objects can reflect the associated phenomena. Conversely, if the safety-critical object is an information object, the information receiver can observe corresponding phenomena.

3) ***Effects analysis****:* The effect analysis in CPEA aims to examine the impact of risk sources on both safety-critical objects and risk-related objects, as well as their interactions. If a risk source involving a safety-critical system object occurs, it may cause harm to risk-related objects that have physical interactions with the safety-critical object, and it may also disrupt the information interaction between the risk-related objects and the safety-critical system object. If the safety-critical object is an information object, the occurrence of a risk source can cause abnormal behavior in the information receiver.

Based on the causes, phenomena, and effects of risk sources, the S&S requirements can be accordingly categorized into three types: *precaution*, *detection*, and *response*. Precaution requirements deal with the causes of risk sources, detection requirements tackle the phenomena, whereas response requirements address the effects. Correspondingly, each type of S&S requirement must have its objectives to be satisfied; otherwise, risks may arise. The allocation of S&S objectives to the three categories of precaution, detection, and response helps to constrain the involved objects and their interactions. The allocation is shown in Table IV.

TABLE IV
ALLOCATION OF SAFETY AND SECURITY OBJECTIVES ACCORDING TO CAUSES, PHENOMENA, AND EFFECTS OF RISKS.

Types of objectives	**Objectives**
Safety precaution objectives	Controllability, Integrity, Availability, Resilience
Safety detection objectives	Operability, Observability, QoS
Safety response objectives	Fault tolerance, Graceful Degradation, Survivability
Security precaution objectives	Confidentiality, Integrity, Availability, Authenticity, Possession and Control, Non-Repudiation
Security detection objectives	Utility, Non-Repudiation
Security response objectives	Utility, Possession and Control, Non-Repudiation

Then, the elicitation of S&S requirements is performed. Precaution and detection requirements constrain the causes-related and phenomena-related objects and interactions to adhere to the S&S precaution and detection objectives. These constraints are expressed utilizing the requirements template that is elaborated in the following subsection. Response requirements are formulated for each *(phenomenon, effect)* link, with each requirement utilizing a phenomenon as a trigger condition and devise the appropriate countermeasures for each effect.

D. Template for Safety and Security Requirements

The real-time specification patterns (RTSP) proposed by Konrad *et al.* [59] are widely used in embedded systems. These patterns enable the representation of specifications in structured natural languages and allow for expressing quantitative and real-time properties of embedded systems. As iCPSs are essentially composed of networked and complex industrial embedded systems, these patterns are also suitable for representing their characteristics. In this context, we have borrowed the structured English grammar and pattern classification from RTSP to establish a template for representing both S&S requirements. Each pattern in this template has a generic syntax that provides a format for expressing specific components in a requirement in constrained natural language. Table V displays the generic syntax and corresponding examples. These examples were extracted from the requirements acquired during the case study. The pattern hierarchy of this template is shown in Fig. 3, which mainly consists of three levels of patterns: *Basic Patterns*, *Task Patterns*, and *Requirement Patterns*. The following provides an introduction to all patterns.

The Basic Patterns provide an expression structure for describing the actions and states of objects and their interactions in iCPSs. Specifically, *Action Patterns* describe the actions performed by system or information objects, while *State Patterns* delineate the state of either objects or interactions. Given that the Action Patterns and State Patterns share similar syntax, the following focus solely on introducing the Action Patterns. The Action Patterns encompass different types that can offer insights into various aspects of actions. *GeneralAction Pattern* is best suitable for describing actions that have no qualitative or quantitative time-related constraints. It primarily employs the syntax structure of *subject* + *verb phrase* + *object* to express the action made by a subject to an object. It utilizes prepositional and verb phrases, as well as adverbs that occur after the object to constrain the verb. The prepositional phrase can include a noun indicating how or by what means the action is achieved. For example, "by the specific-task data 2" in the first example in Table V limits the "continue" action. A verb phrase that appears after the object can also contain a noun indicating the secondary action of the object. For instance, in the second example in Table V, the verb phrase "avoid the obstacles" indicates the secondary action of "AGV1" under the "help" action by "AGV2". In addition, verb phrases can be in either the active voice (the first and second example in Table V) or passive voice (the third example in Table V). Building on the GeneralAction Pattern, *OccurrenceAction Patterns* employ descriptive terms such as "always", "never", and "eventually" to qualitatively describe the occurrence of actions, whereas *RealTimeAction Patterns* are used to quantitatively specify actions with constraints relevant to duration or period.

The Task Patterns use logic and sequence keywords to indicate the order and logic of actions and states that must be executed. Specifically, *LogicalTask Patterns* use keyword "and" or "or" to describe simultaneous or optional actions and states of one object, while *OrderTask Patterns* define the sequence of various objects' actions and states by employing keywords such as "before", "after", "between-and", and "after-until". It is important to note that "between-and" and "after-until" are distinct terms. "between-and" denotes action or state occurring between two specified times or events, while "after-until" signifies action or state commencing at one moment and extending until another. Therefore, in the example of *BetweenTask Pattern* in Table V, the action "AGV1 shall send data to the control system" only needs to occur between the two events "AGV1-state data is complete" and "control center requests for the AGV1-state data", with no requirement on the number of occurrences or duration. In contrast, in the example of *AfterUntilTask Pattern*, the action must continue from "AGV1-state data is complete" until "control center requests for the AGV1-state data" before it ends. Additionally, a crucial distinction among various types of Task Patterns exists: the LogicalTask Patterns permit only one subject; *BeforeTask*, *AfterTask*, and *PrecedenceTask Patterns* allow a maximum of two subjects; *BetweenTask* and *AfterUntilTask Patterns* allow up to three subjects.

Lastly, by consolidating each task with its corresponding triggering conditions (if any), the Requirement Patterns facilitate the obtaining of S&S requirements. The requirements in our template are divided into two categories: general requirements and conditional requirements. General requirements specify tasks throughout the operating process of the system and must be satisfied at all times. These requirements follow the syntax of *GeneralRequirement Pattern*. Conditional Requirements, on the other hand, are activated upon a condition described by either the OrderTask Pattern or LogicalTask Pattern, and define the immediate task to be executed in response to a risk source. Such requirements follow the syntax of *ConditionalRequirement Pattern*. Both types specify a corresponding safety or security objective within their requirements, thereby providing additional information to resolve contradictions.

E. Contradictions in Safety and Security Requirements

In this study, we define a contradiction in S&S requirements as the presence of a pair of S&S requirements that necessitate simultaneous fulfillment, and these requirements either impose conflicting actions or states on the same object, or demand inconsistent sequences for identical actions and states.

Contradictions in S&S requirements can arise between any pair of S&S requirements. These potential contradictions can occur between two safety requirements, two security requirements, or between a safety requirement and a security requirement. The listed real-world requirements below exemplify three categories of S&S requirement contradictions. In an iCPS

TABLE V
TEMPLATE FOR SAFETY AND SECURITY REQUIREMENTS.

Pattern				Generic syntax	Example
Basic Patterns	Action	GeneralAction		[subject] [shall] <verb phrase>[object] [prepositional/verb phrase] [adverb].	1. AGV2 shall continue tasks by the specific-task data 2. 2. AGV2 shall help AGV2 avoid the obstacles. 3. The AGV1-control command shall be obtained by AGV1 merely.
		OccurrenceAction	AlwaysAction	[subject] [shall] always *GeneralAction*.	4. The AGV1-control command shall always be obtained by AGV1 merely.
			AbsenceAction	[subject] [shall] never *GeneralAction*.	5. The AGV1-control command shall never be obtained by AGV2.
			ExistenceAction	[subject] [shall] eventually *GeneralAction*.	6. The transmission state of the AGV1-state data shall eventually be perceived by the AGV1.
		RealTimeAction	DurationAction	[subject] [shall] *Action* for <at least/less than> <numeral num><time unit(s)>.	7. The control center shall transmit the AGV1-control command for at least two seconds.
			PeriodAction	[subject] [shall] *Action* at least every <numeral num><time unit(s)>.	8. The control center shall transmit the AGV1-control command at least every two seconds.
	State	GeneralState		[subject] [shall] <be><adjective/prepositional phrase>.	9. The AGV1-control command shall be complete.
		OccurrenceState	AlwaysState	[subject] [shall] always *GeneralState*.	10. The AGV1-control command shall always be complete.
			AbsenceState	[subject] [shall] never *GeneralState*.	11. The AGV1-control command shall never be incomplete.
			ExistenceState	[subject] [shall] eventually *GeneralState*.	12. The AGV1-control command shall eventually be complete.
		RealTimeState	DurationState	[subject] [shall] *State* for <at least/less than> <numeral num><time unit(s)>.	13. The AGV1-state data shall be complete for at least two seconds.
			PeriodState	[subject] [shall] *State* at least every <numeral num><time unit(s)>.	14. The AGV1-state data shall be complete for at least every two seconds.
Task Patterns	LogicalTask	AndTask		[subject] [shall] *Basic Patterns* and *Basic Patterns*.	15. The AGV1-state data shall be complete and sent to the control system.
		OrTask		[subject] [shall] *Basic Patterns* or *Basic Patterns*.	16. AGV1 shall receive only the safety-critical data or security-critical data.
	OrderTask	BeforeTask		[subject] [shall] *Basic Patterns* before [subject] *Basic Patterns*.	17. The AGV1-state data shall be complete before being sent to the control system.
		AfterTask		[subject] [shall] *Basic Patterns* after [subject] *Basic Patterns*.	18. The AGV1-state data shall send data to the control system after the sensor data is complete.
		PrecedenceTask		The task that <subject>*Basic Patterns* shall be preceded by the task that <subject>*Basic Patterns*.	19. The task that the AGV1-state data is complete shall be preceded by the task that AGV1 sends data to the control system in time.
		BetweenTask		[subject] [shall] *Basic Patterns* between [subject] *Basic Patterns* and [subject] *Basic Patterns*.	20. AGV1 shall send data to the control system between the AGV1-state data is complete and the control center requests for the AGV1-state data.
		AfterUntilTask		[subject] [shall] *Basic Patterns* after [subject] *Basic Patterns* until [subject] *Basic Patterns*.	21. AGV1 shall send data to the control system after the AGV1-state data is complete until the control center receives the AGV1-state data.
Requirement Patterns	GeneralRequirement			For <safety/security><precaution/detection/response>, <subject>shall *Task*.	22. For safety detection, the AGV1-state data shall be received by the control center within the specified time.
	ConditionalRequirement			For <safety/security><precaution/detection/response>, <if/when/while/once>*OrderTask/LogicalTask*, <subject>shall *Task* [after at most/for at least] [numeral num] [time unit(s)].	23. For security response, if the specific-task data 2 indicates that AGV2 was maliciously attacked by the hackers, AGV1 shall receive only the safety-critical or security-critical specific-task data 2.

1. The content in [] and <>is a type of concept or several optional items. <be>can be "is" or "are"; [subject] is a specific system/information object or its action/state, e.g., the reception action of AGV1, the transmission state of AGV1-state data; <safety/security>is one of "safety" and "security"; [prepositional/verb phrase] is one or all of the prepositional and verb phrases.
2. In each pattern, the contents of [] are not necessary. In particular, when a pattern is used in another pattern (e.g., General Action Pattern is used in other Action Patterns), [subject] and [shall] are often omitted.

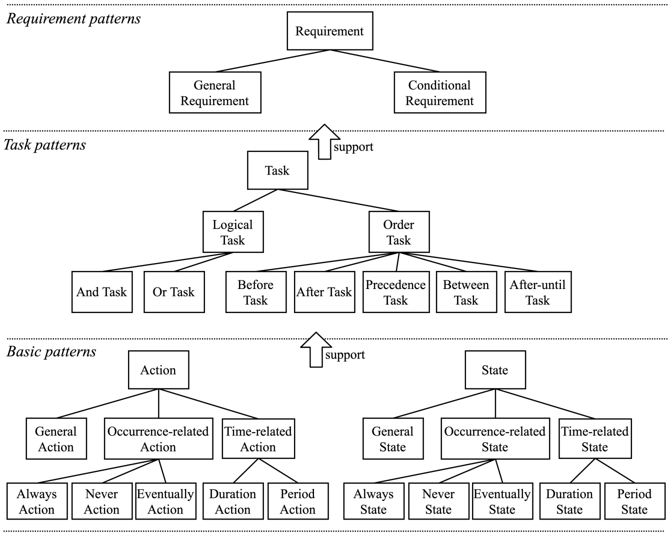

Fig. 3. Pattern hierarchy in requirements template.

that manages chemical processes, an attack by unscrupulous hackers may cause devices like reactors and a data center to fail. In order to maintain system safety, maintenance personnel are required to enter the damaged reactors to repair it (i.e., R1). However, another safety requirement mandates maintenance personnel to maintain a safe distance from the damaged reactors (i.e., R2), which contradicts the previous one. Additionally, maintenance personnel are required to access operational data to meet both S&S response requirements (i.e., R3 and R4). This data is solely accessible at the data center. Nonetheless, the data center may decline access requests according to a security requirement as the requestor's identity can not be verified (i.e., R5), which contradicts R3 and R4. In this context, contradictions arise among the security requirements (R1 and R2), among the security requirements (R4 and R5) and between S&S requirements (R3 and R5).

- **R1:** For safety response, maintenance personnel are required to enter the reactors to repair it.
- **R2:** For safety precaution, maintenance personnel should get away from the damaged reactors.
- **R3:** For safety response, if reactors fail, maintenance personnel should access operational as quickly as possible.
- **R4:** For security response, if the data center fails, maintenance personnel should access operational data as quickly as possible.
- **R5:** For security precaution, maintenance personnel cannot access operational data without authorization.

Furthermore, this study identifies two sufficient conditions that could trigger contradictions in S&S requirements, listed as follows:

- ***Condition 1:*** The actions or states for the same object in the two requirements are contradictory, or the sequences of actions and states in the two requirements are different, as illustrated in Fig. 4.
- ***Condition 2:*** The tasks to be performed in the two requirements need to be satisfied simultaneously.

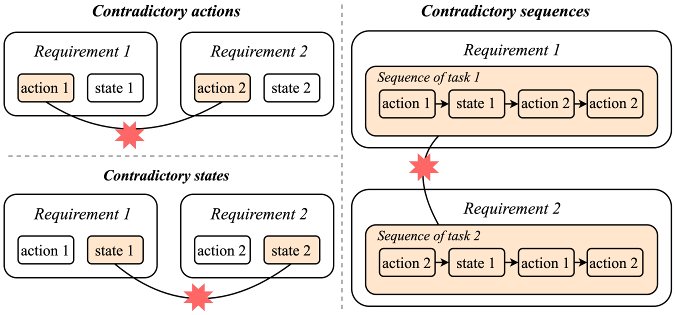

Fig. 4. Diagram of Condition 1 for contradiction. This diagram shows three types of contradiction in Condition 1: 1) the actions in two requirements are contradictory; 2) the states in two requirements are contradictory; and 3) the sequences of identical actions and states in two requirements are different

The requirement template provided in this work enables us to verify if Condition 1 is fulfilled. On the one hand, contradictory actions and states can be identified with the help of the Action/State Patterns employed by two requirements. Such contradictions are often manifested in the conflicting verb/adjective/adverb phrases, or in different types of occurrences of the same subject. For instance, R1 instructs maintenance personnel to "enter" damaged reactors, while R2 demand them to "get away from" from these devices, creating conflicting actions. On the other hand, contradictory sequences can be detected if the two requirements use different OrderTask Patterns for the same group of actions and states. Consider, for instance, the mentioned situation wherein a cyberattack compromises an iCPS controlling a chemical process, and consequently precipitates the failure of the data center and reactor. The security requirement, aiming for rapid information protection, might give higher priority to the action of "disconnecting the reactor from the internet" rather than "halting the reactor". Contrastingly, the safety requirement, with the objective of reducing physical harm, might place "halting the reactor" above "disconnecting the reactor from the internet". In such circumstances, the S&S requirements, when delineating "disconnecting the reactor from the internet" and "halting the reactor", would utilise the BeforeTask and AfterTask templates respectively, culminating in conflicting sequences.

Based on the types of Requirement Patterns used for the two requirements, Condition 2 can be further divided into three situations, as shown in Fig. 5.

- ***Situation 1:*** Both requirements are represented by the GeneralRequirement Pattern, meaning they must be satisfied at all times. In this case, Condition 2 is fulfilled because the tasks specified in both requirements will be performed simultaneously.
- ***Situation 2:*** One requirement is represented by the GeneralRequirement Pattern while the other is represented by the ConditionalRequirement Pattern. In this case, the task specified in the conditional requirement will be executed after its trigger condition occurs, while the task specified in the general requirement will also be executed at this time. As a result, Condition 2 is fulfilled.
- ***Situation 3:*** Both requirements are represented by the ConditionalRequirement Pattern. Since both conditional requirements specify tasks to be executed after their respective triggering conditions occur, the fulfillment of Condition 2 depends on whether the time periods during which the triggering conditions exist overlap. However, due to the complexity of the risk sources' occurrence process, it is difficult to estimate these time periods and human expertise is required for further judgment. In this work, we consider the triggering conditions of requirements arising from the same risk source to potentially overlap in time, resulting in potential contradictions. Determining whether the triggering conditions of requirements arising from different risk sources overlap in time requires human judgment.

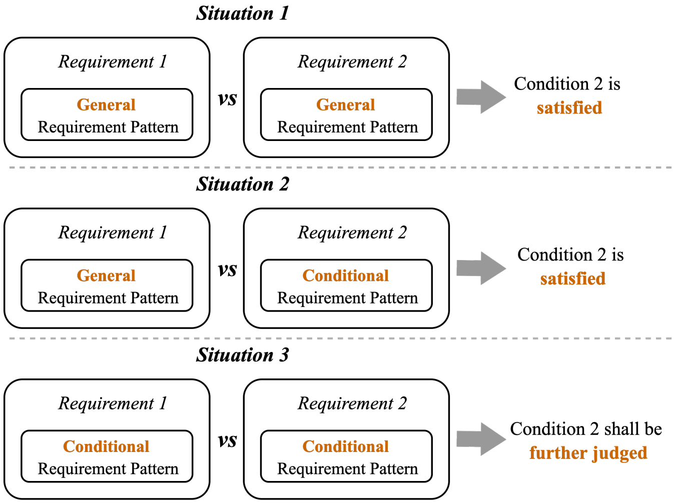

Fig. 5. Diagram of Condition 2 for contradiction.

It is crucial to emphasize that, according to our definition, the objects involved in contradictory requirements must be at the same system level. Nevertheless, various risk sources may have varying system levels, and the corresponding requirements may also involve different system levels. To identify more contradictory requirements, it is necessary to refine different risks to the same system level using a refinement mechanism that is based on the organizational view and applied only to system objects. Each risk is assigned to the subsystem object of the current safety-critical system object until it reaches the required system level. This mechanism can help identify events that may occur simultaneously and discover potential contradictions. The more detailed the information about the risk sources, the higher the likelihood of uncovering contradictions. However, refinement comes with a high cost and should be applied based on the actual needs of the analysis.

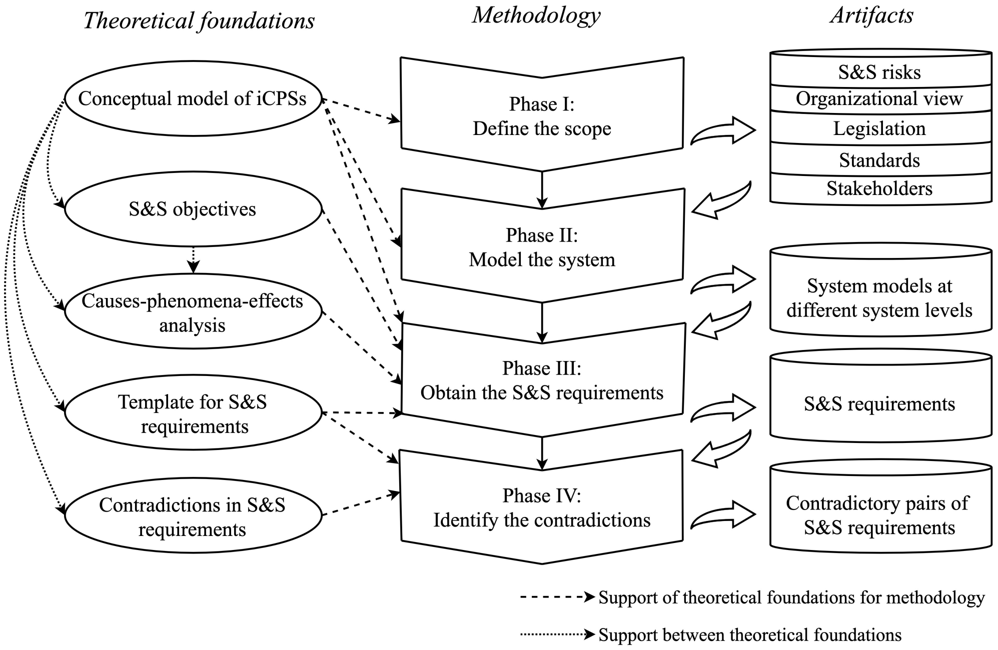
Theoretical foundations
Methodology
Artifacts
Conceptual model of iCPSs
S&S objectives
Causes-phenomena-effects analysis
Template for S&S requirements
Contradictions in S&S requirements
Phase I: Define the scope
Phase II: Model the system
Phase III: Obtain the S&S requirements
Phase IV: Identify the contradictions
S&S risks
Organizational view
Legislation
Standards
Stakeholders
System models at different system levels
S&S requirements
Contradictory pairs of S&S requirements
Support of theoretical foundations for methodology
Support between theoretical foundations

Fig. 6. Supporting relationships among theoretical foundations and methodology.

IV. Four-Phase Methodology

Based on the theoretical foundations outlined above, we propose a four-phase methodology for identifying contradictions in S&S requirements. Fig. 6 shows the artifacts produced in each phase and the relationship between the theoretical foundations and these four phases. In this section, we will introduce each phase in detail.

A. Phase I: Define the Scope

The primary objective of this phase is to determine the analysis scope of the targeted iCPSs and obtain an organizational view. Fig. 7 illustrates the process of this phase. First, stakeholders and global goals or tasks must be identified to determine the system object at the highest level of abstraction (i.e., system level 0). Then, the system object is continuously subdivided into subsystem objects with lower abstract levels according to further goals or tasks until all minimum system objects are identified. The system level of each system object is assigned in ascending order based on the decomposition order, indicating the abstract level from high to low. During each decomposition, stakeholders, legislation, and standards are identified to assist with further division and inform potential risk sources. Based on this analysis, an organizational view of the targeted iCPSs can be obtained as an artifact of this phase. Fig. 8 shows an example of an organizational view that describes system objects in the targeted iCPSs. It is important to note that this work does not provide methods for identifying sources of risks such as hazards and threats, nor does it offer approaches for analyzing adversary models, as they are beyond the scope of this study. These can be derived from the standards involved in the system or other risk identification and adversary modeling methods [14]–[18], [79]–[81]. We also encourage users to integrate additional risk identification, risk analysis, and adversary modeling methods into our approach, as it facilitates a comprehensive understanding of risks and aids in the identification of safety-critical objects.

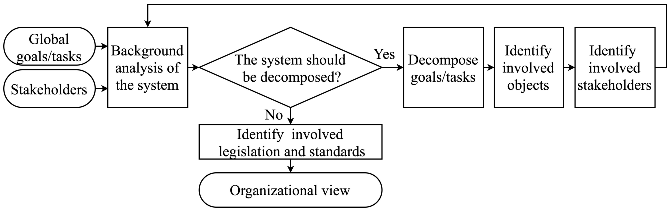
Global goals/tasks
Stakeholders
Background analysis of the system
The system should be decomposed?
Yes
No
Decompose goals/tasks
Identify involved objects
Identify involved stakeholders
Identify involved legislation and standards
Organizational view

Fig. 7. Process of scope definition (Phase I)

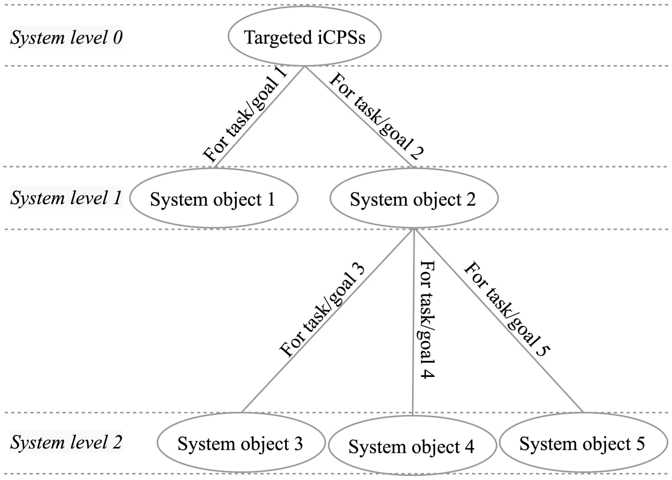

Fig. 8. Organizational view example (Phase I).

B. Phase II: Model the System

The objective of this phase is to thoroughly examine the targeted iCPSs and construct a model of the system. The three stages of this phase are depicted in Fig. 9. Only the ecosystems of decomposable system objects (excluding minimum system objects, which are not decomposable) need to be modelled. For such a system object, its ecosystem includes its subsystem objects and the information, physical, cyber, and energy objects that interact with them. System objects at the same level as the decomposable system object and that interact with objects in its ecosystem should also be considered part of the ecosystem. Next, two types of interactions between objects in the ecosystem must be defined and modelled using an OEI diagram. The resulting model helps identify risk information in the subsequent phase. As an example, consider the organizational view in Fig. 8. This example assumes that only the targeted iCPSs and system object 2 need to be modelled. Analysis of system object 2 reveals that system object 1 is at the same system level as system object 2 and that system object 2 has several subsystem objects, namely system objects 3, 4, and 5. Further identification of other potential objects and interactions in the ecosystem of system object 2 results in an ecosystem model as shown in Fig. 10.

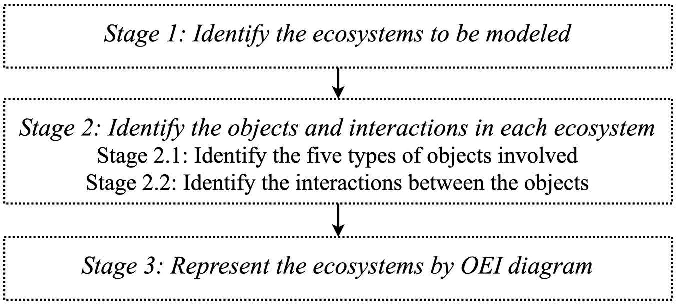
Stage 1: Identify the ecosystems to be modeled
Stage 2: Identify the objects and interactions in each ecosystem
Stage 2.1: Identify the five types of objects involved
Stage 2.2: Identify the interactions between the objects
Stage 3: Represent the ecosystems by OEI diagram

Fig. 9. Steps of modelling the system (Phase II).

C. Phase III: Obtain the Safety and Security Requirements

This phase aims to conduct CPEA on each S&S risk source and derive S&S requirements, based on the risks identified in Phase I and the system models obtained in Phase II. The first step is to apply the refinement mechanism to risks at different system levels as needed, in order to transform the risk sources to the same level. Next, a three-stage process depicted in Fig. 11 is used for each risk to elicit S&S requirements. In this process, five types of risk information must first be identified. CPEA then helps identify cause-related, phenomena-related, and effects-related objects and interactions that must be constrained by S&S precaution, detection, and response objectives. Finally, the S&S precaution, detection, and response requirements express these constraints using the requirements template. To facilitate identification of contradictions, we formally define the ith S&S requirement req_i using (1) and (2) when representing it with the template.

$$req_i = (r_i, t_i, s_i^{A\&S}) \tag{1}$$

$$s_i^{A\&S} = \{(as_j, i)\}_{j=1,2,\cdots,n_{s_i^{A\&S}}} \tag{2}$$

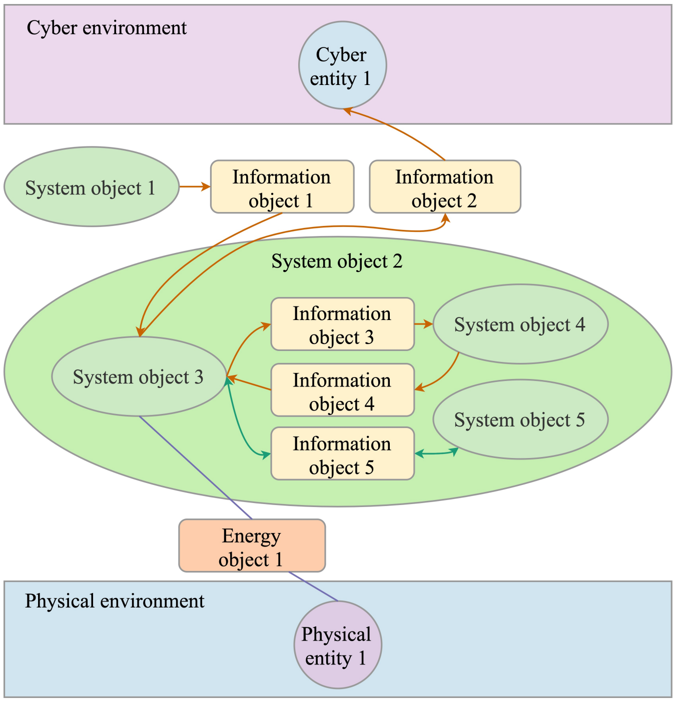

Fig. 10. Ecosystem model of system object 2 in Fig. 8 by the OEI diagram (Phase II).

In the above equations, r_i denotes the type of Requirement Pattern used in req_i, which is a binary value of 0 (GeneralRequirement) or 1 (ConditionalRequirement). t_i denotes the type of Task Pattern used in req_i and can be one of the following: 0 (only the Basic Patterns used), 1 (AndTask), 2 (OrTask), 3 (BeforeTask), 4 (AfterTask), 5 (PrecedenceTask), 6 (BetweenTask), or 7 (AfterUntilTask). $s_i^{A\&S}$ is the set of action and states that appear in sequence in the statement of the task in req_i. as_j is jth action or state sentence in $s_i^{A\&S}$. i in the equation of as_j denotes the corresponding index of requirement (i.e., req_i) that requires such action/state, and $n_{s_i^{A\&S}}$ is the number of action and states. As a result, the S&S requirements obtained in this phase can form a set $R_{S\&S}$, as shown in (3).

$$R_{S\&S} = \{req_i\}_{i=1,2,\cdots,n_{R_{S\&S}}} \tag{3}$$

where $n_{R_{S\&S}}$ is the number of S&S requirements.

D. Phase IV: Identify the contradictions

According to the definition of contradictions in Section III-E, the process of identifying contradictions has three main stages, as shown as Fig. 12 and formalized in the process of Algorithm 1.

*1) **Stage 1: Preprocessing the S&S requirements**:* The goal of this step is to identify pairs of requirements with overlapping components, providing a basis for determining whether sufficient conditions are met. The basic actions and states in the S&S requirements are extracted and then grouped respectively according to the overlap of subject and object. There are two types of overlap between a pair of actions (or states): either they have the same subject and object or

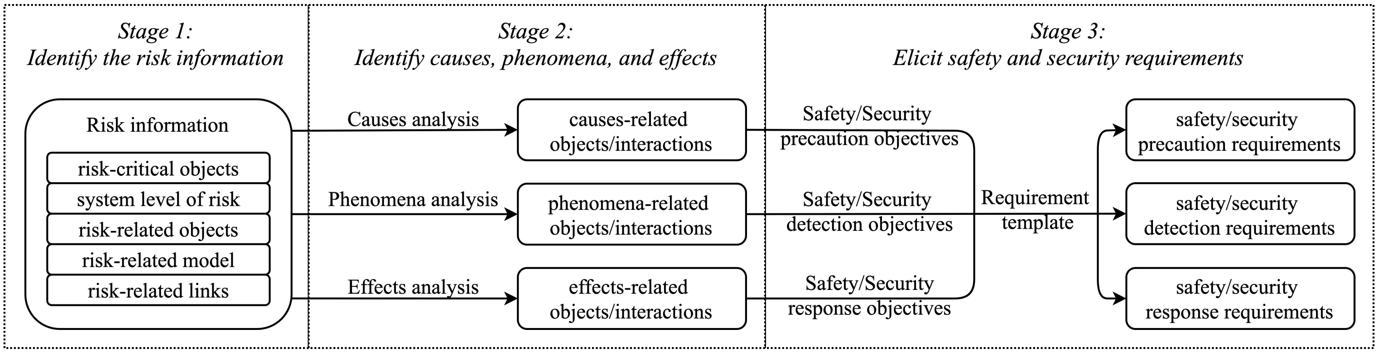
Stage 1:
Identify the risk information
Stage 2:
Identify causes, phenomena, and effects
Stage 3:
Elicit safety and security requirements
Risk information
risk-critical objects
system level of risk
risk-related objects
risk-related model
risk-related links
Causes analysis
Phenomena analysis
Effects analysis
causes-related objects/interactions
phenomena-related objects/interactions
effects-related objects/interactions
Safety/Security precaution objectives
Safety/Security detection objectives
Safety/Security response objectives
Requirement template
safety/security precaution requirements
safety/security detection requirements
safety/security response requirements

Fig. 11. Eliciting safety and security requirements for each risk source by CPEA (Phase III).

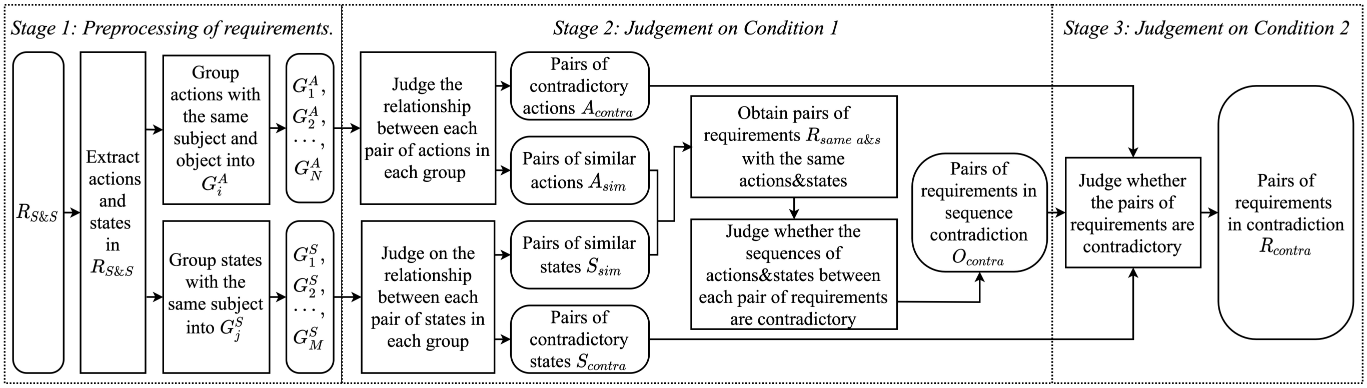
Stage 1: Preprocessing of requirements.
Stage 2: Judgement on Condition 1
Stage 3: Judgement on Condition 2
$R_{S\&S}$
Extract actions and states in $R_{S\&S}$
Group actions with the same subject and object into G_i^A
Group states with the same subject into G_j^S
G_1^A, G_2^A, ..., G_N^A
G_1^S, G_2^S, ..., G_M^S
Judge the relationship between each pair of actions in each group
Judge on the relationship between each pair of states in each group
Pairs of contradictory actions A_{contra}
Pairs of similar actions A_{sim}
Pairs of similar states S_{sim}
Pairs of contradictory states S_{contra}
Obtain pairs of requirements $R_{same\ a\&s}$ with the same actions&states
Judge whether the sequences of actions&states between each pair of requirements are contradictory
Pairs of requirements in sequence contradiction O_{contra}
Judge whether the pairs of requirements are contradictory
Pairs of requirements in contradiction R_{contra}

Fig. 12. Process of contradiction identification (Phase IV).

they have the same subject but no object. The groups of actions/states with overlapping components and their corresponding requirements are then identified. Specifically, the nth group of actions G_n^A is defined as shown in (4).

$$G_n^A = \{(a_j, ida_j^{req})\}_{j=1,2,\cdots,n_{G_n^A}} \tag{4}$$

where a_j, ida_j^{req} and $n_{G_n^A}$ are the action sentence, the index of requirement to which a_j belongs, and the number of actions in G_n^A, respectively. The mth group of states G_m^S is defined as (5).

$$G_m^S = \{(s_j, ids_j^{req})\}_{j=1,2,\cdots,n_{G_m^S}} \tag{5}$$

where s_j, ids_j^{req} and $n_{G_m^S}$ are the state sentence, the index of requirement to which s_j belongs, and the number of actions in G_m^S, respectively.

*2) **Stage 2: Judgement of Condition 1**:* In this stage, it must be determined whether each pair of requirements being evaluated meets Condition 1, i.e., whether there are contradictory actions or states or sequences of actions and states in the two requirements. This judgment is further divided into two steps.

In the first step, it must be determined whether there are contradictory actions/states or the same actions and states in each pair of requirements. Specifically, for each pair of actions/states in each group, the relationship between the two is assessed to determine whether it is similar or contradictory. The generic syntax of an action is summarized as follows: [subject] [shall] [occurrence-related constraint] <verb phrase> [object] [prepositional/verb phrase] [adv] [time-related constraint].

Algorithm 2 provides a formal process for determining the relationship between two actions. Similarly, the generic syntax of a state is summarized as: [subject] [shall] [occurrence-related constraint] <be> [adjective/prepositional phrase] [time-related constraint]. Algorithm 3 supports the determination of the relationship between a pair of states. Based on these two algorithms, pairs of contradictory or similar actions/states and their corresponding requirements are identified and formed into four sets. The indexes of the corresponding requirements are used to label each element of these sets. Specifically, $n_{A_{contra}}$ pairs of contradictory actions form a set A_{contra}, as defined in (6).

$$A_{contra} = \{(ida1_j^{contra}, ida2_j^{contra})\}_{j=1,2,\cdots,n_{A_{contra}}} \tag{6}$$

where $ida1_j^{contra}$ represents the index of the requirement corresponding to the first action in the jth pair of contradictory actions, and $ida2_j^{contra}$ represents the index of the requirement corresponding to the second action in the jth pair of contradictory actions. Similarly, $n_{S_{contra}}$ pairs of contradictory states comprise a set S_{contra}, as defined in (7). $n_{A_{sim}}$ pairs of similar actions form a set A_{sim}, as defined in (8). $n_{S_{sim}}$ pairs of similar states comprise a set S_{sim}, as defined in (9).

$$S_{contra} = \{(ids1_j^{contra}, ids2_j^{contra})\}_{j=1,2,\cdots,n_{S_{contra}}} \tag{7}$$

$$A_{sim} = \{(ida1_j^{sim}, ida2_j^{sim})\}_{j=1,2,\cdots,n_{A_{sim}}} \tag{8}$$

$$S_{sim} = \{(ids1_j^{sim}, ids2_j^{sim})\}_{j=1,2,\cdots,n_{S_{sim}}} \tag{9}$$

Algorithm 1: Contradictions Identification of Safety and Security Requirements.[1]

```
Input: A set of K S&S requirements $R_{S\&S}$
Output: A set of pairs of indexes corresponding to two
        contradictory requirements $R_{contra}$
Initialization: $R_{contra} \leftarrow \varnothing$ (empty); $S_{A\&S} \leftarrow \varnothing$;
  $A_{contra} \leftarrow \varnothing$; $A_{sim} \leftarrow \varnothing$; $S_{contra} \leftarrow \varnothing$; $S_{sim} \leftarrow \varnothing$;
for $i$=1:$n_{R_{S\&S}}$ do
    Append $s_i^{A\&S}$ to $S_{A\&S}$;
end
$G_1^A, ..., G_N^A, G_1^S, ..., G_M^S \leftarrow$ GroupA&S($S_{A\&S}$);
for $n$=1:$n_{G_n^A}$ do
    for each pair of actions $A_{pair}$ in $G_n^A$ do
        $a_{contra}, a_{sim} \leftarrow$
          IsActionsContradictoryOrSimilar($A_{pair}$);
        // This function is used to determine
           whether the input pair of actions is
           similar or contradictory.
        Append $a_{contra}$ to $A_{contra}$;
        Append $a_{sim}$ to $A_{sim}$;
    end
end
for $m$=1:$n_{G_M^S}$ do
    for each pair of actions $S_{pair}$ in $G_m^S$ do
        $s_{contra}, s_{sim} \leftarrow$
          IsStatesContradictoryOrSimilar($S_{pair}$);
        // This function is used to determine
           whether the input pair of states is
           similar or contradictory.
        Append $s_{contra}$ to $S_{contra}$;
        Append $s_{sim}$ to $S_{sim}$;
    end
end
$A\&S_{sim} \leftarrow A_{sim} \cup S_{sim}$;
Obtain matrix $M_{req \times sim\, a\&s}$ from $A\&S_{sim}$;
$R_{same\, a\&s} \leftarrow$ FindSameA&SReqPairs($M_{req \times sim\, a\&s}$);
// This function is used to determine which
   pairs of requirements have the same actions
   and states.
$O_{contra} \leftarrow$ IsOrderContradictory($R_{same\, a\&s}$,$R_{S\&S}$);
// This function is used to determine whether
   the sequence of actions and states is
   contradictory for pairs of requirements
   that have the same actions and states.
$R_{tmpcontra} \leftarrow A_{contra} \cup S_{contra} \cup O_{contra}$;
$R_{contra} \leftarrow$ IsTrueContradctort($R_{tmpcontra}$);
// This function is used to determine whether
   pairs of requirements, which have already
   met the first sufficient condition of
   contradiction, are required to be fulfilled
   simultaneously.
return $R_{conflict}$
```

Algorithm 2: Judgment on the relationship between actions.[2]

(Corresponding to the function *IsActionsContradictoryOrSimilar* in Algorithm 1).

```
Input: A pair of actions $A_{pair}$
Output: A set of index pairs corresponding to two
        conflicting requirements $R_{conflict}$
if RelpOfVerb($A_{pair}$) is Similar then
    if RelpOfPreVerbPhrase($A_{pair}$) is Similar then
        if RelpOfAdv($A_{pair}$) is Similar then
            if RelpOfTime($A_{pair}$) is Contradictory then
                return $a_{contra}$;
            else
                return $a_{sim}$;
            end
        else if RelpOfAdv($A_{pair}$) is Contradictory then
            return $a_{contra}$;
        else
            return $a_{sim}$;
        end
    else if RelpOfPreVerbPhrase($A_{pair}$) is Contradictory
      then
        return $a_{contra}$;
    else
        return $a_{sim}$;
    end
else if RelpOfVerb($A_{pair}$) is Contradictory then
    return $a_{contra}$;
else
    return Not Contradict;
end
```

Algorithm 3: Judgment on the relationship between states.[3]

(Corresponding to the function *IsStatesContradictoryOrSimilar* in Algorithm 1).

```
Input: A pair of actions $S_{pair}$
Output: A set of index pairs corresponding to two
        conflicting requirements $R_{conflict}$
if RelpOfAdjPre($S_{pair}$) is Similar then
    if RelpOfTime($S_{pair}$) is Contradictory then
        return $s_{contra}$;
    else
        return $s_{sim}$;
    end
else if RelpOfAdjPre($S_{pair}$) is Contradictory then
    return $s_{contra}$;
else
    return Not Contradict;
end
```

Specifically, pairs of requirements corresponding to contradictory actions and states (i.e., A_{contra} and S_{contra}) are considered to have potential contradictions and are ultimately evaluated in stage 3. In contrast, pairs of requirements corresponding to similar actions and states are evaluated in the second step of this stage.

In the second step, pairs of requirements with the same actions and states are determined based on the sets of similar actions and states (i.e., A_{sim} and S_{sim}). For each such pair of requirements, the consistency of the sequences of actions and states in the two requirements must be evaluated. Algorithm 4 provides a method for such judgment. As an output, $n_{O_{contra}}$ pairs of requirements with contradictory orders of actions and states are identified and form a set $O_{contradict}$, as defined in (10).

$$O_{contra} = \{(ido1_j^{contra}, ido2_j^{contra}\}_{j=1,2,\cdots,n_{O_{contra}}} \quad (10)$$

[1] Time complexity: $\Theta(n^2)$, space complexity: $\Theta(n)$, where n is the number of requirements in the input set $R_{S\&S}$. Source code can be found at https://github.com/zhicongsun/Identifying-Contradictions-in-Safety-and-Security-Requirements-for-iCPSs

[2] Time complexity: $\Theta(1)$, space complexity: $\Theta(1)$.

[3] Time complexity: $\Theta(1)$, space complexity: $\Theta(1)$.

where $ido1_j^{contra}$ represents the index of one requirement in the jth pair of requirements in sequence contradiction, $ido2_j^{contra}$ represents the index of another requirement.

Algorithm 4: Judgment on the consistence of sequences of actions and states.[1]

(Corresponding to the function *IsOrderContradictory* in Algorithm 1).

```
Input: A set of index pairs corresponding to pairs of
       requirements with the same actions and states
       $R_{same\,a\&s}$; $R_{S\&S}$
Output: A set of index pairs corresponding to pairs of
        requirements with contradictory sequences of
        actions and states $O_{contra}$
Initialization: $O_{contra} \leftarrow \varnothing$;
for each index pairs $(id_{req1}, id_{req2})$ in $R_{same\,a\&s}$ do
    Obtain the corresponding requirements pair
      $(req_{id_{req1}}, req_{id_{req2}})$ in $R_{S\&S}$;
    Obtain the actions and states $(s^{A\&S}_{id_{req1}}, s^{A\&S}_{id_{req2}})$ and
      $(t_{id_{req1}}, t_{id_{req2}})$of $(req_{id_{req1}}, req_{id_{req2}})$;
    if $(t_{id_{req1}} = 1\&t_{id_{req2}} = 2)$ or
      $(t_{id_{req1}} = 2\&t_{id_{req2}} = 1)$ then
        Append $(id_{req1}, id_{req2})$ to $O_{contra}$;
    else
        Arrange an index for each action and state;
        // Arrange the sequence of actions
           and states of the each
           requirement according to its
           pattern
        $ID_{sequenceofreq1} \leftarrow$ ArrangeIDSequence($t_{id_{req1}}$);
        $ID_{sequenceofreq2} \leftarrow$ ArrangeIDSequence($t_{id_{req2}}$);
        if $ID_{sequenceofreq1} \neq ID_{sequenceofreq2}$ then
            Append $(id_{req1}, id_{req2})$ to $O_{contra}$;
        end
    end
end
return $O_{contra}$
```

After the above two steps, pairs of requirements that may be contradictory are identified (i.e., A_{contra}, S_{contra}, and O_{contra}) and need to be further evaluated in the next stage.

*3) **Stage 3: Judgement of Condition 2:*** For each pair of potentially contradictory requirements, it must be determined whether they need to be satisfied simultaneously. This is done by first identifying the types of the two requirements and then determining the situations according to the definition in Section III-E. If a pair of requirements falls under Situation 1 or Situation 2, they must be satisfied simultaneously and are therefore contradictory.

Otherwise, such a pair of requirements belongs to Situation 3 and must be further evaluated to determine whether their trigger conditions can co-occur. If they can occur simultaneously, the two requirements are contradictory. This work provides a tool called the event network for making such judgments. In the event network, risk sources, their causes, phenomena, and effects are considered as events and represented by circles. Circles of the same size represent events at the same system level. The larger the circle, the higher the abstraction level of the event and the lower the system level. Furthermore, each risk source in the network can be expanded into a fully connected network of causes, phenomena, and effects. The occurrence time of causes, phenomena, and impacts in the risk source is not strictly distinguished in this work. They are considered to co-occur because the timeline is unclear and difficult to obtain. Additionally, a red line with two arrows is used to indicate that events involved in different risk sources may occur simultaneously. Fig. 13 shows an example of an event network. In this example, risk source 2 (RC 2) has a higher abstract level (lower system level) than risk source 1 (RC 1) and is refined into two risk sources (i.e., risk source 3 (RC 3) and risk source 4 (RC 4)) that are at the same system level as risk source 1. The causes, phenomena, and effects of RC 3 are marked as C3.1, P3.1, P3.2, E3.1, E3.2, E3.3. Similarly, C4.1, C4.2, P4.1, P4.2, E4.1, E4.2, E4.3, E4.4, E4.5 are marked for RC 4. Additionally, link 1 indicates that RC 1 and E3.3 may occur simultaneously, while link 2 indicates that C3.1 and C4.1 may also occur simultaneously. Thus, RC 1 and the causes, phenomena, and effects of RC 3 may co-occur; the causes, phenomena, and effects of RC 3 and RC 4 may also co-occur.

After the above analysis, a set of pairs of contradictory requirements $R_{contradict}$ is obtained, as defined in (11).

$$R_{contra} = \{(id1_j^{contra}, id2_j^{contra}\}_{j=1,2,\cdots,n_{R_{contra}}} \tag{11}$$

where $id1_j^{contra}$ represents the index of one requirement in the jth pair of contradictory requirements, $id2_j^{contra}$ represents the index of another requirement, and $n_{R_{contra}}$ is the number of pairs of requirements in contradiction.

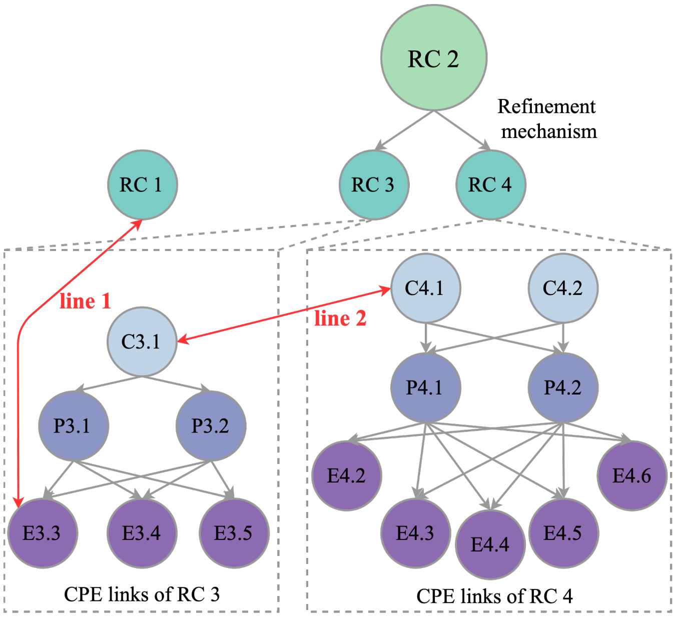

Fig. 13. An example of the event network.

V. Case Study

This work applies the proposed method to a typical factory. This factory is depicted by the software named SIEMENS Plant Simulation, as shown in Fig. 14. Its operation process consists of three stages: material delivery, production, and

[1] Time complexity: $\Theta(n)$, space complexity: $\Theta(n)$, where n is the number of index pairs in $R_{same\,a\&s}$.

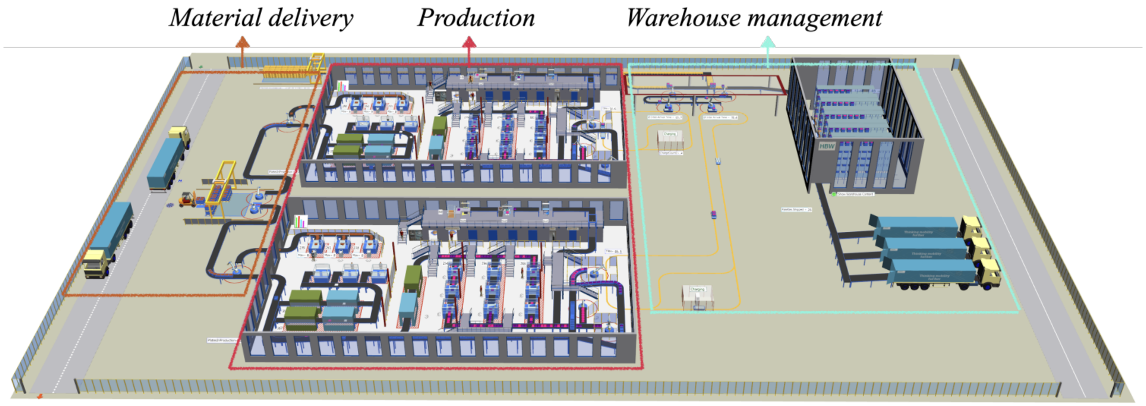

Fig. 14. The typical factory.

warehouse management. In the material delivery stage, the raw materials needed for production are transported to the factory by trucks. Workers use forklifts to unload the goods, and the materials are finally sent to the production line by conveyor belts. In the production stage, various devices process and transport materials with the cooperation of workers and the final products are placed on wire-guided AGVs by a mechanical arm. In the warehouse management stage, AGVs transfer the products to the warehouse; the workers use forklifts to transfer the products to the trucks for outgoing according to business needs.

Although the factory has achieved a certain degree of automation, the devices used for handling goods in these three processes are not networked and thus need to be controlled manually by managers and work independently. In addition, there is no data link between the business system and the goods-handling system, resulting in the inability to produce and ship goods on demand. These two characteristics limit the efficiency of the value chain from business to production, so the owner of this factory decided to upgrade the factory to make it smarter. The initial plan of the smart factory is to network all the automated devices required for production and use the multi-AGV system (AGVS) with autonomous mobility to replace the existing logistics system and human resources. In this context, S&S requirements and their contradictions need to be analysed.

To demonstrate our approach more clearly, this case study simplifies the functional architecture of the smart factory to consist of only business systems and AGVS. There are only two AGVs and one control center in the AGVS. The following describes the four-phase process of discovering a contradiction in S&S requirements for the smart factory. In particular, this case introduces the elicitation process of only part of the S&S requirements for two typical risks because such an analysis is enough to demonstrate our methodology comprehensively. Additionally, the process of identifying a deeply hidden contradiction among these requirements is illustrated in this case study.

A. *Phase I: Define the Scope*

The ultimate task of the smart factory is to make the value chain from industrial production to business more efficient based on iCPSs.

Firstly, factory managers and business managers are identified as the stakeholders. In the opinion of business managers, the smart factory should provide a business system that focuses only on transaction orders and automatically performs the dispatch of goods required for orders. In the view of factory managers, production efficiency should be improved by equipping the factory with automated robots. Consequently, two functional tasks are obtained — business and production. Accordingly, the smart factory (at system level 0) is divided into the business system and AGVS (at system level 1). Then, the stakeholders involved are identified as the designers of the business platform and AGVS. With the help of stakeholders, AGVS-related laws and standards, including ISO 3691-4, ANSI/ITSDF B56.1a-2018, and GB/T 37669-2019, are identified. These standards guide the decomposition and list related risks of AGVS, e.g., an AGVS is maliciously attacked and controlled by hackers (threat).

Next, we perform further decomposition for AGVS since it involves both S&S risks, while the business system involves only security risks. AGVS is essentially a multi-robot collaborative system that consists of a control center responsible for global control tasks and AGVs responsible for various specific tasks. In this simplified case study, only two AGVs and one control center exist in the AGVS; and they are regarded as the minimum system objects according to the requirement of analysis. Thus, the organizational view of the smart factory is shown in Fig. 15.

B. *Phase II: Model the System*

From the organizational view in Fig. 15, we can determine that the system objects to be modelled are the smart factory and AGVS because they can be further decomposed. By exploring the smart factory, the involved objects and interactions are determined: The subsystem objects of the smart factory are the AGVS and the business system; both of them are connected to the cyber environment and exposed to the external iCPSs and the hackers; the AGVS receives the AGVS-task command

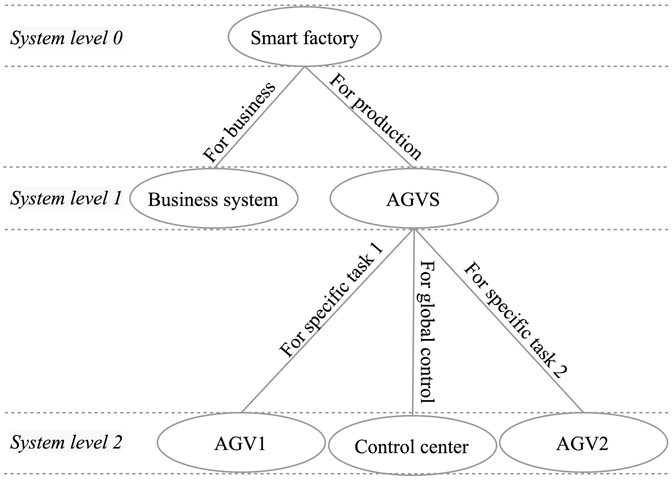

Fig. 15. Organizational view of the smart factory.

from the business system and feeds back AGVS-state data to the business system; the AGVS needs to handle the goods and may have to avoid contact with obstacles and users; users should control the business system. As a result, the ecosystem model of the smart factory is shown in Fig. 16. By similar analysis, the ecosystem model of AGVS is shown in Fig. 17.

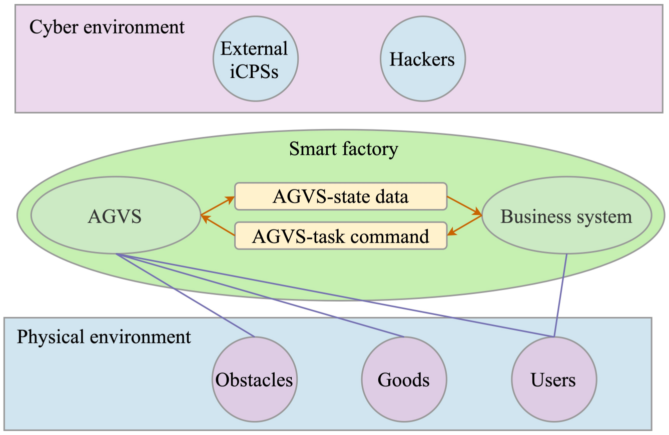

Fig. 16. Ecosystem model of the smart factory (Phase II).

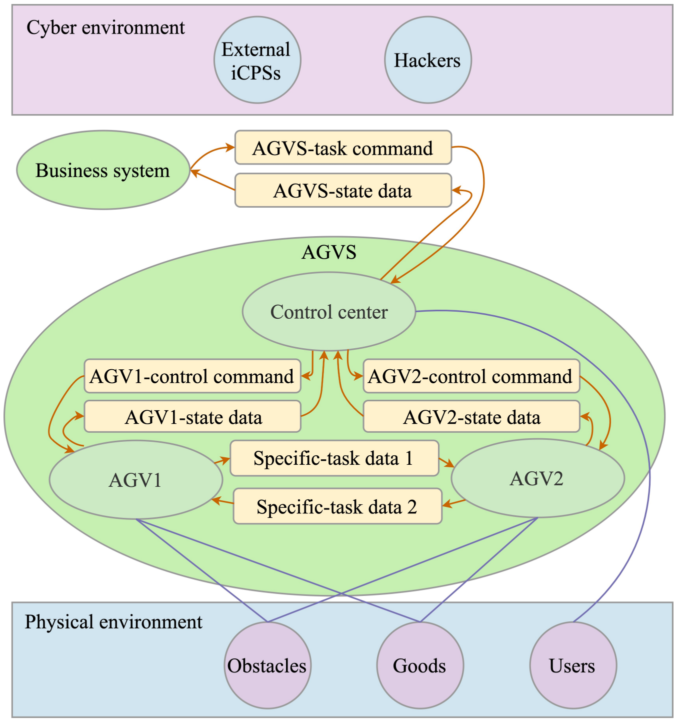

Fig. 17. Ecosystem model of the AGVS (Phase II).

C. Phase III: Obtain the Safety and Security Requirements

In order to cover a more comprehensive range of risks, i.e., different system levels and sources, this case study considers the following two risks obtained in Phase I as the object to be analysed. Then we apply CPEA for each risk to elicit S&S requirements.

- Risk source 1 (RC 1): AGV1 has abnormal contact with the obstacles (hazard).
- Risk source 2 (RC 2): The AGVS was maliciously attacked and controlled by the hackers (threat).

*1) **Stage 1: Identify the risk information**:* In this stage, the five types of risk information defined in Section III-C should be identified. RC 1 involves two objects, i.e., AGV1 and the obstacles. AGV1 is the only system object, so it is regarded as the object. Then we can determine that RC 1 is at system level 2 and the risk-related model is the ecosystem of AGVS. Fig. 18 shows the ecosystem of AGVS and the identified risk information of RC 1. In the model, AGV1 is represented by dark red.

The system objects (AGV2 and Control center), physical objects (Goods and Obstacles), and cyber objects (External iCPSs and Hackers) that interact with AGV1 are regarded as risk-related objects and represented by light red. Then, the information objects between AGV1 and the above system and physical objects, including the AGV1-control command, AGV1-state command, specific-tasks data 1, and specific-task data 2, are also considered as risk-related objects and represented by brown in Fig. 18; however, as AGV1 and the risk-related cyber objects are in the same cyber environment but do not directly interact with each other, interactive content between them is not included in the information objects. After the above analysis, the risk information about RC 1 is summarized in Table VI.

By a similar analysis process, the identified risk information of RC 2 is summarized in Table VII and shown in Fig. 19. However, RC 1 is at system level 2 while RC 2 is at system level 1; such a difference makes it difficult to identify potential contradictions in the S&S requirements. To deal with this issue, we need to apply the refinement mechanism to refine the objects involved in RC 2 to the same system level as RC 1, i.e., system level 2. According to the organizational view in Fig. 15, AGVS has three subsystems, i.e., AGV1, AGV2, and the control center. Therefore, RC 2 can be further divided into the following three risk sources:

- RC 2a: The control center was maliciously attacked and controlled by the hackers.
- RC 2b: AGV1 was maliciously attacked and controlled

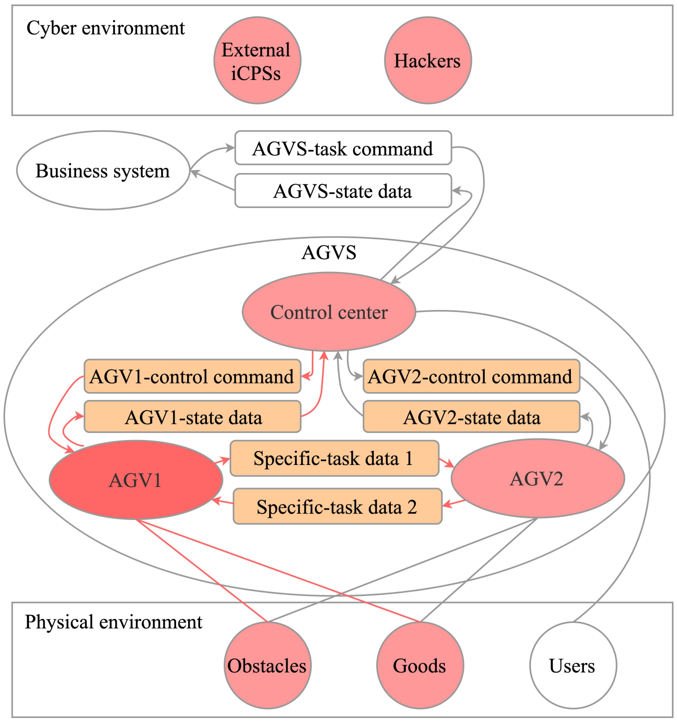

Fig. 18. Identified risk information of RC 1 (Phase III).

TABLE VI
RISK INFORMATION ABOUT RC 1.

Type of risk information		Identified risk information
System level of risk		System level 2
Safety-critical object		AGV1
Risk-related model		The ecosystem of AGVS
Risk-related objects	System objects	AGV2, control center
	Information objects	AGV1-control command, AGV1-state command, Specific-task data 1, Specific-task data 2
	Physical objects	Obstacles, Goods
	Cyber objects	External iCPSs, Hackers
Risk-related interaction	Direct interaction	AGV1–Control center, AGV1–AGV2, AGV1–Obstacles, AGV1–Goods
	Indirect interaction	AGV1–External iCPSs AGV1–Hackers

by the hackers.

- RC 2c: AGV2 was maliciously attacked and controlled by the hackers.

All of these three risks have corresponding requirements. Since the contradiction to be introduced in this case study occurs between the S&S requirements of RC 1 and RC 2c, we only introduce the analysis process of these two risks in the following. The identified risk information of RC 2c is obtained and shown in Fig. 20 and summarized in Table VIII.

2) ***Stage 2: Identify causes, phenomena, and effects:*** In this stage, the causes, phenomena, and effects of RC 1 and

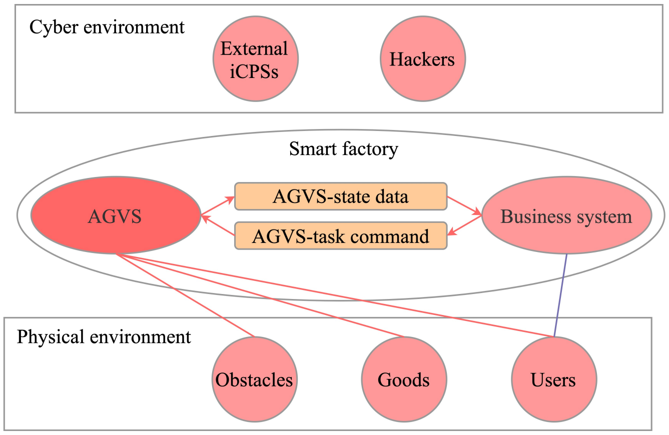

Fig. 19. Identified risk information of RC 2 (Phase III).

TABLE VII
RISK INFORMATION ABOUT RISK 2.

Type of risk information		Identified risk information
System level of risk		System level 1
Safety-critical object		AGVS
Risk-related model		The ecosystem of smart factory
Risk-related objects	System objects	Business system
	Information objects	AGVS-state data, AGVS-task command
	Physical objects	Obstacles, Goods, Users
	Cyber objects	External iCPSs, Hackers
Risk-related interaction	Direct interaction	AGVS–Business system, AGVS–Obstacles, AGVS–Goods, AGVS–Users
	Indirect interaction	AGVS–External iCPSs, AGVS–Hackers

TABLE VIII
RISK INFORMATION ABOUT RC 2C.

Type of risk information		Identified risk information
System level of risk		System level 2
Safety-critical object		AGV2
Risk-related model		The ecosystem of AGVS
Risk-related objects	System objects	AGV1, control center
	Information objects	AGV2-control command, AGV2-state command, Specific-task data 1, Specific-task data 2
	Physical objects	Obstacles, Goods
	Cyber objects	External iCPSs, Hackers
Risk-related interaction	Direct interaction	AGV2–Control center, AGV2–AGV1, AGV2–Obstacles, AGV2–Goods
	Indirect interaction	AGV2–External iCPSs, AGV2–Hackers

TABLE IX
CAUSES, PHENOMENA, AND EFFECTS OF RC 1.

Causes of RC 1		Causes-related objects	Causes-related interactions
C1.1	AGV1 did not correctly detect the obstacles at the right time.	AGV1, Obstacles	Energy
C1.2	AGV1 was deviated from the original route under the force of the goods.	AGV1, Goods	Energy
C1.3	AGV1 did not communicate correctly with the control center at the right time.	AGV1, Control center	AGV1-control command, AGV1-state data
C1.4	AGV1 did not communicate correctly with AGV2 at the right time.	AGV1, AGV2	Specific-task data 1 Specific-task data 2
C1.5	AGV1 was attacked and controlled by the hackers.	AGV1, Hackers	Information
Phenomena of RC 1		**Phenomena-related objects**	**Phenomena-related interactions**
P1.1	The AGV1-state data transmitted by AGV1 to the control center indicates abnormal contact between AGV1 and the obstacles.	AGV1-state data	AGV1–Control center AGV1–Obstacles
P1.2	The specific-task data 1 transmitted by AGV1 to AGV2 indicates abnormal contact between AGV1 and the obstacles.	Specific-task data 1	AGV1–AGV2 AGV1–Obstacles
Effects of RC 1		**Effects-related objects**	**Effects-related interactions**
E1.1	AGV1 will be damaged by the obstacles or damage the obstacles.	AGV1, Obstacles	Energy
E1.2	AGV1 will be damaged by the goods or damage the goods.	AGV1, Goods	Energy
E1.3	Communication failures between AGV1 and the control center.	AGV1, Control center	AGV1-control command, AGV1-state data
E1.4	Communication failures between AGV1 and AGV2.	AGV1, AGV2	Specific-task data 1 Specific-task data 2
E1.5	AGV1 will disconnect from the cyber environment or affect the devices therein.	AGV1, Cyber objects	Information

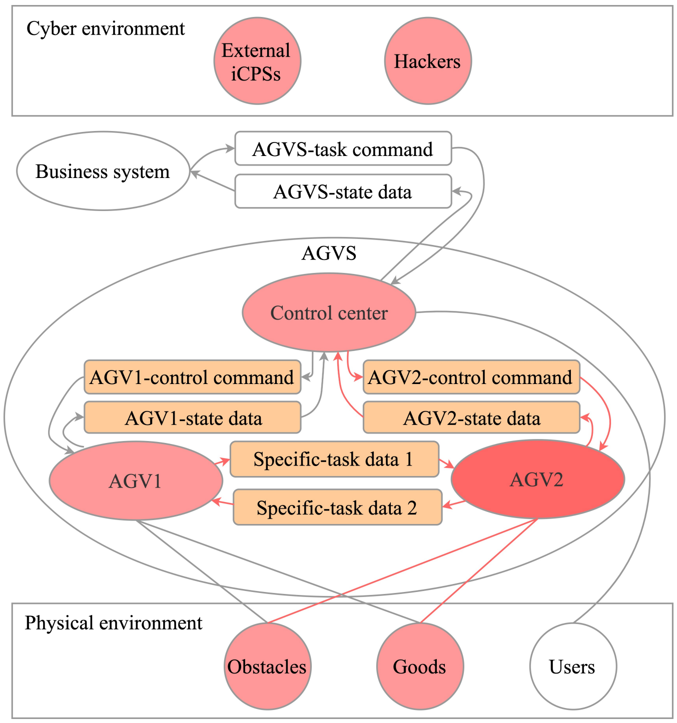

Fig. 20. Identified risk information of RC 2c (Phase III).

RC 2c should be identified. For RC 1, the causes (C1.1-C1.5), phenomena (P1.1-P1.2), effects (E1.1-E1.5), and the related objects and interactions are summarized in Table IX; for RC 2c, the causes (C2c.1-C2c.4), phenomena (P2c.1-P2c.2), effects (E2c.1-E2c.4), and their related objects and interactions are summarized in Table X. The analysis process of RC 1 and RC 2c is similar, therefore this paper takes the analysis of RC 1 as an example in the following.

The safety-critical object of RC 1 is a system object, i.e., AGV1. As such, the risk may arise from the following causes: (a) The abnormal contact between AGV1 and the risk-related physical objects (AGV1-Obstacles and AGV1–Goods) or the function failure of AGV1. In Table IX, "Energy" is used to summarize such causes-related energy interaction since it is so complex that the specific content of the energy is beyond the scope of our analysis; (b) The failed communication between AGV1 and the risk-related system objects (AGV1–Control center and AGV1–AGV2). Such causes-related information interactions are represented by specific information objects (e.g., Specific-task data 1) in Table IX; (c) The cyber-attack from cyber objects. The involved information interactions between AGV1 and cyber objects (AGV1–External iCPSs and AGV1–Hackers) are indirect, and only hackers have the motivation for cyber-attack; however, the interactive content is challenging to determine and thus is represented by the "Information" in Table IX.

After the above causes happen, RC 1 may occur and its risk phenomena are reflected by the information object transmitted by AGV1, i.e., the AGV1-state data and specific-task data 1. In Table IX, the phenomena-related objects are specified by the information objects; the corresponding interactions are denoted by the affected interactions and the transmitter and receiver of the information. For example, the AGV1-state data (phenomena-related object) is transmitted from AGV1 to the control center (phenomena-related interaction) and indicates the abnormal interaction between AGV1 and the obstacles (phenomena-related interaction).

Then, RC 1 may have a negative impact on the effects-related objects. Similar to the causes, the effects may include the abnormal behavior of the risk-related cyber objects (in the same network), physical objects (Obstacles and Goods) and system objects (AGV2 and Control center), as shown in Table IX.

TABLE X
CAUSES, PHENOMENA, AND EFFECTS OF RC 2C.

Causes of RC 2c		**Causes-related objects**	**Causes-related interactions**
C2c.1	AGV2 was invaded by the hackers through the AGV2-control command.	AGV2, Hackers	AGV2-control command
C2c.2	AGV2 was invaded by the hackers through the specific-task data 1.	AGV2, Hackers	Specific-task data 1
C2c.3	AGV2 was invaded by the hackers through the cyber environment.	AGV2, Cyber objects	Information
Phenomena of RC 2c		**Phenomena-related objects**	**Phenomena-related interactions**
P2c.1	The AGV2-state data transmitted by AGV2 to the control center indicates that AGV2 was maliciously attacked and controlled by the hackers.	AGV2-state data	AGV2–Control center AGV2–Hackers
P2c.2	The specific-task data 2 transmitted by AGV2 to AGV1 indicates that AGV2 was maliciously attacked and controlled by the hackers.	Specific-task data 2	AGV2–AGV1 AGV2–Hackers
Effects of RC 2c		**Effects-related objects**	**Effects-related interactions**
E2c.1	AGV2 will have abnormal contact with the obstacles.	AGV2, Obstacles	Energy
E2c.2	AGV2 will have abnormal contact with the goods.	AGV2, Goods	Energy
E2c.3	AGV2 will not be able to control itself.	AGV2	None
E2c.4	AGV2 will not be able to cooperate with the control center.	AGV2, Control center	AGV2-state data AGV2-control command
E2c.5	AGV2 will be used to attack the control center.	AGV2, Control center	AGV2-state data AGV2-control command
E2c.6	AGV2 will not be able to cooperate with AGV1.	AGV2, AGV1	Specific-task data 2 Specific-task data 1
E2c.7	AGV2 will be used to attack AGV1.	AGV2, AGV1	Specific-task data 2 Specific-task data 1
E2c.8	AGV2 will disconnect from the cyber environment or affect the devices therein.	AGV2, Cyber objects	Information

*3) **Stage 3: Elicit safety and security requirements**:* In this stage, the S&S precaution, detection, and response objectives are used to constrain the objects and interactions related to causes, phenomena, and effects. Additionally, the requirements template is adopted to express these constraints. To comprehensively explain the elicitation process, this study analyses one event from each of the causes, phenomena, and effects of RC 1.

Firstly, we consider the causes of RC 1. C1.1 and C1.2 have a system object (AGV1) and a physical object (e.g., Obstacles in C1.1) but no information objects; thus, these causes have no corresponding security requirements. C1.5 involves system object (AGV1) and cyber entity (Hackers), but the interactive information is not specific; thereby, the security requirements are difficult to analyse. C1.3 and C1.4 involve two system objects (e.g., AGV1 and Control center in C1.3) and specific information objects (e.g., AGV1-control command in C1.3); consequently, they have both S&S requirements. We take C1.3 as an example for analysis and only consider the AGV1-control command as the object to be designed in this example.

The corresponding S&S precaution requirements are expressed by the GeneralRequirement Pattern, as shown in Table XI.

Secondly, we consider the phenomena of RC 1. Both P1.1 and P1.2 involve an information interaction between system objects, such as the transmission of AGV-state data from AGV1 to the control center. Such an information object should belong to the S&S detection objectives, i.e., operability, QoS, utility, and non-repudiation. We take P1.1 as an example for requirements elicitation, and the corresponding S&S detection requirements are expressed by the GeneralRequirement Pattern, as shown in Table XII.

Finally, we consider the phenomenon-effect links of RC 2c, which have the corresponding S&S response requirements. Each phenomenon and effect can form a link, so there are ten links in RC 1. However, we only illustrate the corresponding requirements of (P1.2, E1.1) because one requirement in the pair of contradictory requirements in this case study comes from (P1.2, E1.1), as shown in Section V-D. The S&S response requirements for (P1.2, E1.1) are expressed by the ConditionalRequirement Pattern, as shown in Table XIII.

Additionally, the S&S response requirements for (P2c.2, E2c.7) of RC 2c are introduced in Table XIV as one of the contradictory requirements comes from the (P2c.2, E2c.7), as shown in Section V-D.

D. Phase IV: Identify the Contradictions

By applying the four-stage method to the S&S requirements obtained in Phase III, the contradiction between R21 and R32 is identified in our case study.

- **R21:** For safety response, if the specific-task data 1 indicates AGV1 was slightly damaged by the obstacles or damaged the obstacles, AGV1 shall receive the specific-task data 2.
- **R32:** For safety response, if the specific-task data 2 indicates that AGV2 was invaded by the hackers and lost control, AGV1 shall refuse the specific-task data 2.

These two requirements are formally defined as the $req_2 1$ and $req_3 2$ according to (1), as shown in (12) and (13).

$$req_2 1 = (1, 0, s_{21}^{A\&S}) \tag{12}$$

$$req_3 2 = (1, 0, s_{32}^{A\&S}) \tag{13}$$

The task to be performed for requirement R21 is “AGV1 shall receive the specific-task data 2”, which is the action sentence in $s_{21}^{A\&S}$; The tasks to be performed for requirement R32 is “AGV1 shall refuse the specific-task data 2”, which is the action sentence in $s_{32}^{A\&S}$. Specifically, the process of identifying this contradiction is as follows:

TABLE XI
SAFETY AND SECURITY PRECAUTION REQUIREMENTS FOR C1.3 OF RC 1.

	Safety precaution requirements for C1.3 of RC 1	**Safety precaution objectives**
R1	For safety precaution, AGV1 shall receive the AGV1-control command at the right time.	Controllability
R2	For safety precaution, the control center shall transmit the AGV1-control command at the right time.	Controllability
R3	For safety precaution, the AGV1-control command shall be complete.	Integrity
R4	For safety precaution, the AGV1-control command shall be available when AGV1 needs it.	Availability
R5	For safety precaution, AGV1 shall recognize errors in the AGV1-control command and execute the correct command.	Resilience
	Security precaution requirements for C1.3 of RC 1	**Security precaution objectives**
R6	For security precaution, the AGV1-control command shall be obtained by AGV1 merely.	Confidentiality
R7	For security precaution, the AGV1-control command shall be modified by the control center merely.	Integrity
R8	For security precaution, the AGV1-control command shall be available when AGV1 needs it.	Availability
R9	For security precaution, the reception action of AGV1 shall be configured by AGV1 merely.	Authenticity
R10	For security precaution, the transmission action of the control center shall be configured by the control center merely.	Authenticity
R11	For security precaution, the AGV1-control command shall be obtained by AGV1 merely.	Possession and Control
R12	For security precaution, the control center shall be responsible for the AGV1-control command.	Non-Repudiation

TABLE XII
SAFETY AND SECURITY DETECTION REQUIREMENTS FOR P1.1 OF RC 1.

	Safety detection requirements for P1.1 of RC 1	**Safety detection objectives**
R13	For safety detection, the AGV1-state data shall be within a specified scope.	Operability
R14	For safety detection, the transmission state of the AGV1-state data shall be perceived by the AGV1.	Observability
R15	For safety detection, the the reception state of the AGV1-state data shall be perceived by the control center.	Observability
R16	For safety detection, the AGV1-state data shall be transmitted by the AGV1 within the specified time.	QoS
R17	For safety detection, the AGV1-state data shall be received by the control center within the specified time.	QoS
	Security detection requirements for P1.1 of RC 1	**Security detection objectives**
R18	For security detection, the AGV1-state data shall include AGV1 state data, safety-critical data and security-critical data.	Utility
R19	For security detection, AGV1 shall be responsible for the AGV1-state data.	Non-Repudiation

TABLE XIII
SAFETY AND SECURITY RESPONSE REQUIREMENTS FOR (P1.2, E1.1) OF RC 1.

	Safety response requirements for (P1.2, E1.1) of RC 1	**Safety response objectives**
R20	For safety response, if the specific-task data 1 indicates AGV1 was slightly damaged by the obstacles or damaged the obstacles, AGV2 shall help AGV1 avoid the obstacles and then continue tasks by the specific-task data 2.	Fault tolerance
R21	For safety response, if the specific-task data 1 indicates AGV1 was slightly damaged by the obstacles or damaged the obstacles, AGV1 shall receive the specific-task data 2	Fault tolerance
R22	For safety response, if the specific-task data 1 indicates AGV1 was partially damaged by the obstacles or damaged the obstacles, AGV2 shall help AGV1 avoid the obstacles by the specific-task data 2.	Graceful Degradation
R23	For safety response, if the specific-task data 1 indicates AGV1 was partially damaged by the obstacles or damaged the obstacles, AGV1 shall receive only the safety-critical and security-critical specific-task data 2.	Graceful Degradation
R24	For safety response, if the specific-task data 1 indicates AGV1 was seriously damaged by the obstacles or damaged the obstacles, AGV2 shall stop AGV1 by the specific-task data 2.	Survivability
R25	For safety response, if the specific-task data 1 indicates AGV1 was seriously damaged by the obstacles or damaged the obstacles, AGV1 shall receive only the safety-critical and security-critical specific-task data 2.	Survivability
	Security response requirements for (P1.2, E1.1) of RC 1	**Security response objectives**
R26	For security response, if the specific-task data 1 indicates AGV1 was damaged by the obstacles or damaged the obstacles, the specific-task data 2 shall have only the security-related and safety-related data.	Utility
R27	For security response, if the specific-task data 1 indicates AGV1 was damaged by the obstacles or damaged the obstacles, AGV1 shall receive only the security-related and safety-related specific-task data 2.	Utility
R28	For security response, if the specific-task data 1 indicates AGV1 was damaged by the obstacles or damaged the obstacles, the specific-task data 2 shall be modified only by AGV2 and obtained only by AGV1.	Possession and control
R29	For security response, if the specific-task data 1 indicates AGV1 was damaged by the obstacles or damaged the obstacles, AGV2 shall be responsible for the specific-task data 2.	Non-Repudiation

TABLE XIV
SAFETY AND SECURITY REQUIREMENTS FOR (P2C.2, E2C.7) OF RC 2C.

	Safety response requirements for (P2c.2, E2c.7) of RC 2	**Safety response objectives**
R30	For safety response, if the specific-task data 2 indicates that AGV2 was invaded but not controlled by the hackers, AGV1 shall receive the specific-task data 2 and then correct errors in the specific-task data 2.	Fault tolerance
R31	For safety response, if the specific-task data 2 indicates that AGV2 was invaded and partially controlled by the hackers, AGV1 shall receive the safety-critical and security-critical specific-task data 2 and then correct errors in the safety-critical and security-critical specific-task data 2.	Graceful Degradation
R32	For safety response, if the specific-task data 2 indicates that AGV2 was invaded by the hackers and lost control, AGV1 shall refuse the specific-task data 2.	Survivability
	Security response requirements for (P2c.2, E2c.7) of RC 2	**Security response objectives**
R33	For security response, if the specific-task data 2 indicates that AGV2 was maliciously attacked and controlled by the hackers, AGV1 shall receive only the safety-critical or security-critical specific-task data 2.	Utility
R34	For security response, if the specific-task data 2 indicates that AGV2 was maliciously attacked and controlled by the hackers, AGV1 shall refuse the specific-task data 2.	Possession and control
R35	For security response, if the specific-task data 2 indicates that AGV2 was maliciously attacked and controlled by the hackers, AGV2 shall be responsible for the specific-task data 2.	Non-Repudiation

*1) **Stage 1: Preprocessing the S&S requirements.**:* In this stage, the action sentences in R21 ($s_{21}^{A\&S}$) and R32 ($s_{32}^{A\&S}$) are extracted respectively. Next, these actions are grouped into the same group due to the same subject (AGV1) and object (the specific-task data 2).

*2) **Stage 2: Judgement of Condition 1.**:* In this stage, the relationship between $s_{21}^{A\&S}$ and $s_{32}^{A\&S}$ should be judged. With the help of Algorithm 1,Algorithm 2, Algorithm 3 and Algorithm 4, the contradiction between "receive" and "refuse" is identified; thereby, $s_{21}^{A\&S}$ and $s_{32}^{A\&S}$ is in contradiction, their corresponding requirements R21 and R32 are regarded as the potential pair of contradictory requirements.

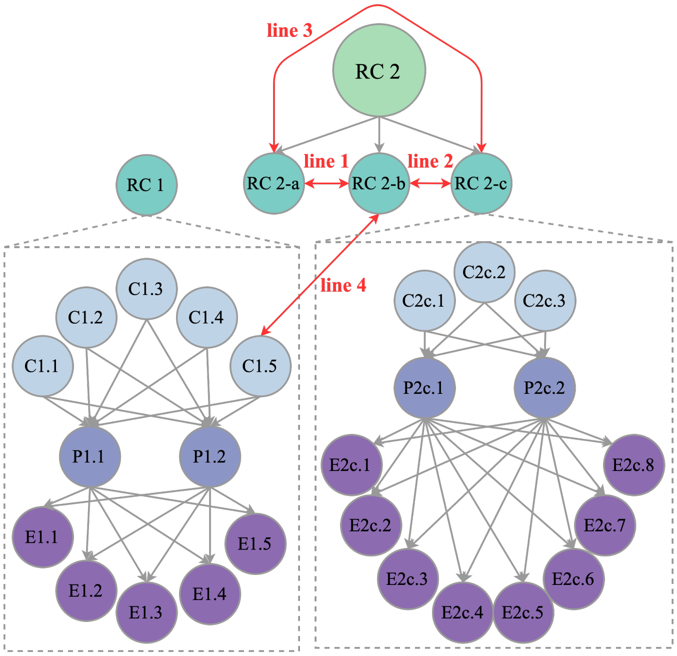

Fig. 21. The event network in this case study. Line 1, line 2, and line 3 indicate that AGV1, AGV2, and control center may be maliciously attacked at the same time. Line 4 indicates that TODO. This mean that the cases, phenomena, and effects of risk sources are possible to co-occur.

*3) **Stage 3: Judgement of Condition 2.**:* In this stage, we should judge if R21 and R32 need to be satisfied at the same time. As R21 and R32 are conditional requirements, their trigger conditions (i.e., RC 1 and RC 2c) should be analysed with the help of the event network shown in Fig. 21. In this figure, part of the events at the same system level involved in RC 1 and RC 2c are described, including RC 1, and the causes, phenomena, and effects of RC 2c. In Fig. 21, the red lines with arrows (link 1, link 2, link 3, and link 4) indicate that the events they point to may occur simultaneously. As shown in link 1, link 2, and link 3, RC 2a (the control center was maliciously attacked and controlled by the hackers), RC 2b (AGV1 was maliciously attacked and controlled by the hackers), and RC 2c (AGV2 was maliciously attacked and controlled by the hackers) may happen at the same time, which indicates that their cases, phenomena, and effects are possible to co-occur. Further, as shown in link 4, C1.5 (AGV1 was attacked and controlled by the hackers) and RC 2b (AGV2 was maliciously attacked and controlled by the hackers) may co-occur if the whole AGVS is invaded and controlled by the hackers. As such, RC 1 and RC 2c may happen simultaneously, i.e., the triggers of R21 and R32 could co-occur. Therefore, the contradictory tasks of R21 and R32 should be performed simultaneously, which means there is a contradiction between R21 and R32.

VI. DISCUSSION

A. Advances Compared with Related Studies

Arguably, our methodology furnishes a systematic and novel framework for identifying contradictions in S&S requirements. It particularly overcomes two significant drawbacks found in related studies.

Firstly, we offer a more comprehensive and accurate definition of contradictions in S&S requirements as compared to existing studies. Among the relevant studies, Novak and Treytl [34] do not give a clear definition of contradictions; Gu et al. [36], Sun et al. [35] and Agbo and Mehrpouyan [37] regard contradiction as caused by two requirements requiring opposite behaviors or states under the same conditions; most of these studies only consider the contradictions between safety and security [33]–[36]. Contrarily, we define a contradiction in S&S requirements as the presence of a pair of S&S requirements that necessitate simultaneous fulfillment, and these requirements either impose conflicting actions or states on the same object, or demand inconsistent sequences for identical actions and states. Our definition broadens aspects that earlier studies may have overlooked. Specifically, it considers the situation where the co-existence of different conditions may culminate in contradictory states or actions; it also covers the possibility of contradictions both within safety requirements and security requirements, as opposed to solely between the two.

Secondly, this work considers synergy with the S&S requirements elicitation while unifying the their expression, providing a suitable context for identifying contradictions. Arguably, two factors need to be considered in order to provide a better premise for identifying contradictions in S&S requirements. On the one hand, the expression of requirements should facilitate the identification of contradictions while remaining understandable. We represent the S&S requirements utilizing structured natural language. This approach not only helps the decomposition of requirements and detection of contradictions but also mitigates the comprehensibility issues associated with the formal methods cited in the works of Gu et al. [36], Sun et al. [35], Agbo and Mehrpouyan [37]. On the other hand, a unified representation that expresses requirements using similar components (e.g., subject, predicate, and object) and logic structures (e.g., if-then and once-then) can help detect contradictions. Such a unified representation must be based on a consistent approach to eliciting requirements, as variations would lead to inconsistent descriptions of identical components, diverse expression structures of similar requirements, and inefficient use of S&S risk information. Our approach effectively considers this factor. In detail, we introduce a tailored S&S requirements template to guarantee uniform expression structures across all S&S requirements. We employ the CPEA method to standardize the elicitation procedure for S&S requirements. Additionally, we offer a conceptual iCPSs model for S&S requirements analysis. Collectively, these methodologies contribute to diminishing ambiguity and inconsistency in S&S requirements, cultivating a better environment for contradiction identification.

Unfortunately, due to the limited implementation details provided by relevant studies, which mostly present an idea for identifying requirement contradictions, a quantitative comparative analysis is not feasible. Nevertheless, inspired by the work of Guzman et al. [78], we suggest a potential way for testing and comparing our approach. Guzman et al. exploited the criteria of “completeness” and “required effort” for evaluating and contrasting their risk identification methodology. Similarly, we could perform contradiction identification methods on the same requirements set and evaluate their abilities of contradictions identification by the criterion “completeness”, namely the ratio of identified to actual contradictions. Further, we could record the time requisite to understand and implement different methods on an identical case study to evaluate their “required effort”.

B. Limitations and Potential Improvement

The method proposed in this paper has five main constraints to be improved. First, we do not guarantee that the S&S objectives used in the method are comprehensive, although they are widely used. In fact, there are currently no S&S objectives that have been proven to be complete. Second, although we specify the process of CPEA in detail, there may be inconsistencies in its analysis results due to differences among analysts. Such inconsistencies may lead to slight deviations in the requirements, although these deviations will not be significant. Third, some functions of the algorithms provided in this work must be carried out manually. For example, the function used for judging verb relationships in Algorithm 2 is conducted by analysts. We argue that natural language processing technology may help automate such functions. Fourth, our method is time-consuming and resource-intensive. This is reflected in both the requirements elicitation and contradictions identification processes. During requirements elicitation, CPEA must be applied to each potential risk source to determine its causes, phenomena, and effects, and S&S requirements must then be obtained for each. This series of processes requires an unpredictable amount of time and resources. However, this is common in the S&S field where analysis must be comprehensive and rely on domain knowledge and expert experience. To address this, we are developing tools such as requirement writing software with prompts to expedite the process. Our algorithms for identifying contradictions are also time-consuming and resource-intensive, with a maximum time complexity of $\Theta(n^2)$ and a maximum space complexity of $\Theta(n)$, where n is the number of requirements in the input set $R_{S\&S}$. Nevertheless, these algorithms represent just a typical implementation of our idea, and we are currently exploring more efficient implementations. Finally, our proposed contradiction conditions are sufficient but not necessary, potentially resulting in unidentified contradictions. This limitation is difficult to avoid due to the challenge of achieving completeness in the S&S field. The limited number of real-world contradiction cases also makes determining necessary conditions for contradictions difficult. Therefore, our options are limited. We can only strive to provide a comprehensive definition of contradictions and identify as many potential contradictions as possible.

VII. Conclusion

This study proposes a novel and systematic methodology to identify contradictions in the S&S requirements of iCPSs, effectively addressing a research gap in the field where relevant methodologies are scarce, and facilitating the reduction of often neglected risks within iCPSs. Moreover, the theories advanced in this research are worthy of further study. For instance, the conceptual model proposed for iCPSs could be further developed for system modelling, the suggested S&S requirement template could be broadened to represent functional requirements, and the performance of the contradictions identification method could be further improved.

In our future work, in addition to addressing the above shortcomings, we will seek a methodology to eliminate contradictions in S&S requirements. It is possible that contradictions may not be resolved, but rather replaced by other requirements that can achieve similar objectives while having a lower likelihood of being contradictory. This may involve the problem of optimal searching in the space of requirements. Additionally, the S&S requirements after contradiction elimination should be further verified, which requires that the requirements template can support formal methods.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant Nos. 92067109, 61873119, and 62211530106), the Shenzhen Science and Technology Program (Grant Nos. ZDSYS20210623092007023 and GJHZ20210705141808024), and the Educational Commission of Guangdong Province (Grant No. 2019KZDZX1018).

References

[1] A. Colombo, S. Karnouskos, O. Kaynak, Y. Shi, and S. Yin, "Industrial cyberphysical systems: A backbone of the fourth industrial revolution," *IEEE Ind Electron Mag*, vol. 11, no. 1, pp. 6–16, 2017.

[2] R. Ma, P. Cheng, Z. Zhang, W. Liu, Q. Wang, and Q. Wei, "Stealthy attack against redundant controller architecture of industrial cyber-physical system," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9783–9793, 2019.

[3] F. Tao, Q. Qi, L. Wang, and A. Y. Nee, "Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019.

[4] T. Aven, Y. Ben-Haim, H.-B. Andersen, T. Cox, D.-E. Lopez, and M. Greenberg, "Society for Risk Analysis," *Encyclopedia of Science and Technology Communication.*, 2012.

[5] G. Sabaliauskaite and A. P. Mathur, "Aligning cyber-physical system safety and security." Complex Systems Design and Management Asia, 2015, pp. 41–53.

[6] E. Lisova, I. Šljivo, and A. Čaušević, "Safety and security co-analyses: A systematic literature review," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2189–2200, 2018.

[7] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[8] W. Duo, M. Zhou, and A. Abusorrah, "A survey of cyber attacks on cyber physical systems: Recent advances and challenges," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 5, pp. 784–800, 2022.

[9] L. Piètre-Cambacédès and M. Bouissou, "Modeling safety and security interdependencies with bdmp (boolean logic driven markov processes)," in *2010 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2010, pp. 2852–2861.

[10] S. Kriaa, L. Pietre-Cambacedes, M. Bouissou, and Y. Halgand, "A survey of approaches combining safety and security for industrial control systems," *Reliability Engineering and System Safety*, vol. 139, pp. 156–178, 2015.

[11] S. Chockalingam, D. Hadžiosmanović, W. Pieters, A. Teixeira, and P. van Gelder, "Integrated safety and security risk assessment methods: a survey of key characteristics and applications," in *Critical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 10–12, 2016, Revised Selected Papers 11*. Springer, 2017, pp. 50–62.

[12] M. Wolf and D. Serpanos, "Safety and security in cyber-physical systems and internet-of-things systems," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 9–20, 2017.

[13] G. Kavallieratos, S. Katsikas, and V. Gkioulos, "Safesec tropos: Joint security and safety requirements elicitation," *Computer Standards and Interfaces*, vol. 70, p. 103429, 2020.

[14] S. Kriaa, M. Bouissou, and Y. Laarouchi, "A model based approach for SCADA safety and security joint modelling: S-cube," *IET Conference Proceedings*, 2015.

[15] I. Friedberg, K. McLaughlin, P. Smith, D. Laverty, and S. Sezer, "STPA-SafeSec: Safety and security analysis for cyber-physical systems," *Journal of Information Security and Applications*, vol. 34, pp. 183–196, 2017.

[16] N. C. Guzman, D. K. M. Kufoalor, I. Kozine, and M. A. Lundteigen, "Combined safety and security risk analysis using the ufoi-e method: A case study of an autonomous surface vessel." Proceedings of the 29th European Safety and Reliability Conference, Lower Saxony, Germany, 2019, pp. 22–26.

[17] N. H. Carreras Guzman and I. Kozine, "Uncontrolled flows of information and energy in cyber-physical systems," *European Safety and Reliability Association Newsletter*, pp. 2–3, 2018.

[18] N. H. C. Guzman, I. Kozine, and M. A. Lundteigen, "An integrated safety and security analysis for cyber-physical harm scenarios," *Safety science*, vol. 144, p. 105458, 2021.

[19] N. Subramanian and J. Zalewski, "Assessment of safety and security of system architectures for cyberphysical systems," IEEE. 2013 IEEE International Systems Conference (SysCon), 2013, pp. 634–641.

[20] A. J. Kornecki, N. Subramanian, and J. Zalewski, "Studying interrelationships of safety and security for software assurance in cyber-physical systems: Approach based on bayesian belief networks," IEEE. 2013 Federated Conference on Computer Science and Information Systems, 2013, pp. 1393–1399.

[21] R. Kumar and M. Stoelinga, "Quantitative security and safety analysis with attack-fault trees," IEEE. 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), 2017, pp. 25–32.

[22] G. Sabaliauskaite, L. S. Liew, and J. Cui, "Integrating autonomous vehicle safety and security analysis using stpa method and the six-step model," *International Journal on Advances in Security*, vol. 11, no. 1, 2, pp. 160–169, 2018.

[23] F. Asplund, J. McDermid, R. Oates, and J. Roberts, "Rapid integration of cps security and safety," *IEEE Embedded Systems Letters*, vol. 11, no. 4, pp. 111–114, 2018.

[24] A. Tantawy, S. Abdelwahed, and A. Erradi, "Cyber lopa: An integrated approach for the design of dependable and secure cyber-physical systems," *IEEE Transactions on Reliability*, 2022.

[25] A. Gu, Z. Yin, C. Cui, and Y. Li, "Integrated functional safety and security diagnosis mechanism of cps based on blockchain," *IEEE Access*, vol. 8, pp. 15 241–15 255, 2020.

[26] C. Ponsard, G. Dallons, and P. Massonet, "Goal-oriented co-engineering of security and safety requirements in cyber-physical systems," vol. 9923, 09 2016, pp. 334–345.

[27] M. Brunner, M. Huber, C. Sauerwein, and R. Breu, "Towards an integrated model for safety and security requirements of cyber-physical systems," IEEE. 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), 2017, pp. 334–340.

[28] S. Japs, "Security & safety by model-based requirements engineering," in *2020 IEEE 28th International Requirements Engineering Conference (RE)*. IEEE, 2020, pp. 422–427.

[29] A. Pasquini, D. P. Eames, and J. Moffett, "The integration of safety and security requirements," in *Computer Safety, Reliability and Security: 18th International Conference, SAFECOMP'99 Toulouse, France, September 27–29, 1999 Proceedings 18*. Springer, 1999, pp. 468–480.

[30] L. Piètre-Cambacédès and M. Bouissou, "Modeling safety and security interdependencies with bdmp (boolean logic driven markov processes)," IEEE. 2010 IEEE International Conference on Systems, Man and Cybernetics, 2010, pp. 2852–2861.

[31] J. P. Thomas IV, "Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis," Ph.D. dissertation, Massachusetts Institute of Technology, 2013.

[32] X. Lyu, Y. Ding, and S. H. Yang, "Safety and security risk assessment in cyber-physical systems," *IET Cyber-Physical Systems: Theory and Applications*, vol. 4, pp. 221–232, 2019.

[33] C. Menon and S. Vidalis, "Towards the resolution of safety and security conflicts," in *2021 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2021, pp. 1–6.

[34] T. Novak and A. Treytl, "Functional safety and system security in automation systems-a life cycle model," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, 2008, pp. 311–318.

[35] M. Sun, S. Mohan, L. Sha, and C. Gunter, "Addressing safety and security contradictions in cyber-physical systems," Citeseer. Proceedings of the 1st Workshop on Future Directions in Cyber-Physical Systems Security (CPSSW'09), 2009.

[36] T. Gu, M. Lu, and L. Li, "Extracting interdependent requirements and resolving conflicted requirements of safety and security for industrial control systems," IEEE. 2015 First International Conference on Reliability Systems Engineering (ICRSE), 2015, pp. 1–8.

[37] C. Agbo and H. Mehrpouyan, "Conflict analysis and resolution of safety and security boundary conditions for industrial control systems," in *2022 6th International Conference on System Reliability and Safety (ICSRS)*. IEEE, 2022, pp. 145–156.

[38] J. Dunjó, V. Fthenakis, J. A. Vílchez, and J. Arnaldos, "Hazard and operability (HAZOP) analysis. A literature review," *Journal of Hazardous Materials*, vol. 173, no. 1-3, pp. 19–32, 2010.

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

[40] D. Pereira, C. Hirata, R. Pagliares, and S. Nadjm-Tehrani, "Towards combined safety and security constraints analysis," in *International conference on computer safety, reliability, and security*. Springer, 2017, pp. 70–80.

[41] C. S. Cho, W. H. Chung, and S. Y. Kuo, "Using Tree-Based Approaches to Analyze Dependability and Security on IC Systems in Safety-Critical Systems," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1118–1128, 2018.

[42] C. Schmittner, Z. Ma, and P. Smith, "Fmvea for safety and security analysis of intelligent and cooperative vehicles," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2014, pp. 282–288.

[43] W. Young and N. Leveson, "Systems thinking for safety and security." Proceedings of the 29th Annual Computer Security Applications Conference, 2013, pp. 1–8.

[44] X. Chen, L. Han, J. Liu, and H. Sun, "Using safety requirement patterns to elicit requirements for railway interlocking systems," IEEE. 2016 IEEE 24th International Requirements Engineering Conference Workshops (REW), 2016, pp. 296–303.

[45] M. Riaz, J. Slankas, J. King, and L. Williams, "Using templates to elicit implied security requirements from functional requirements-a controlled experiment." Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement, 2014, pp. 1–10.

[46] L. Provenzano, K. Hänninen, J. Zhou, and K. Lundqvist, "An ontological approach to elicit safety requirements," IEEE. 2017 24th Asia-Pacific Software Engineering Conference (APSEC), 2017, pp. 713–718.

[47] A. Souag, C. Salinesi, R. Mazo, and I. Comyn-Wattiau, "A security ontology for security requirements elicitation," Springer. International symposium on engineering secure software and systems, 2015, pp. 157–177.

[48] T. Ishimatsu, N. G. Leveson, J. P. Thomas, C. H. Fleming, M. Katahira, Y. Miyamoto, R. Ujiie, H. Nakao, and N. Hoshino, "Hazard analysis of complex spacecraft using systems-theoretic process analysis," *Journal of Spacecraft and Rockets*, vol. 51, no. 2, pp. 509–522, 2014.

[49] W. Young and N. Leveson, "Systems thinking for safety and security," *ACM International Conference Proceeding Series*, pp. 1–8, 2013.

[50] P. Salini and S. Kanmani, "Effectiveness and performance analysis of model-oriented security requirements engineering to elicit security requirements: a systematic solution for developing secure software systems," *International Journal of Information Security*, vol. 15, no. 3, pp. 319–334, 2016.

[51] J.-M. Bruel, S. Ebersold, F. Galinier, M. Mazzara, A. Naumchev, and B. Meyer, "The role of formalism in system requirements," *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–36, 2021.

[52] T. Nguyen, "Form-l: A modelica extension for properties modelling illustrated on a practical example," no. 96, Linköping University Electronic Press. Proceedings of the 10 th International Modelica Conference; March 10-12; 2014; Lund; Sweden, 2014, pp. 1227–1236.

[53] J. P. Bowen, M. Hinchey, and E. Vassev, "Formal Requirements Specification," *Encyclopedia of Software Engineering*, pp. 321–332, 2010.

[54] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, "Easy approach to requirements syntax (ears)," in *2009 17th IEEE International Requirements Engineering Conference*. IEEE, 2009, pp. 317–322.

[55] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel, "Relax: Incorporating uncertainty into the specification of self-adaptive systems," in *2009 17th IEEE International Requirements Engineering Conference*. IEEE, 2009, pp. 79–88.

[56] W. J. Lee, S. D. Cha, and Y. R. Kwon, "Integration and analysis of use cases using modular petri nets in requirements engineering," *IEEE Transactions on software engineering*, vol. 24, no. 12, pp. 1115–1130, 1998.

[57] D. Zhu, H. Tan, and S. Yao, "Petri nets-based method to elicit component-interaction related safety requirements in safety-critical systems," *Computers and Electrical Engineering*, vol. 71, pp. 162–172, 2018.

[58] A. Naumchev, "Seamless object-oriented requirements," in *2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)*. IEEE, 2019, pp. 0743–0748.

[59] S. Konrad and B. H. Cheng, "Real-time specification patterns," in *Proceedings of the 27th international conference on Software engineering*, 2005, pp. 372–381.

[60] W. Jue, Y. Song, X. Wu, and W. Dai, "A semi-formal requirement modeling pattern for designing industrial cyber-physical systems," in *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1. IEEE, 2019, pp. 2883–2888.

[61] E. A. Lee, "Cyber physical systems: Design challenges," in *2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC)*. IEEE, 2008, pp. 363–369.

[62] L. Hu, N. Xie, Z. Kuang, and K. Zhao, "Review of cyber-physical system architecture," in *2012 IEEE 15th international symposium on object/component/service-oriented real-time distributed computing workshops*. IEEE, 2012, pp. 25–30.

[63] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[64] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing letters*, vol. 3, pp. 18–23, 2015.

[65] E. A. Lee, "The past, present and future of cyber-physical systems: A focus on models," *Sensors*, vol. 15, no. 3, pp. 4837–4869, 2015.

[66] P. Derler, E. A. Lee, and A. S. Vincentelli, "Modeling cyber–physical systems," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 13–28, 2011.

[67] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *2014 IEEE international conference on automation, quality and testing, robotics*. IEEE, 2014, pp. 1–4.

[68] N. H. C. Guzman, M. Wied, I. Kozine, and M. A. Lundteigen, "Conceptualizing the key features of cyber-physical systems in a multi-layered representation for safety and security analysis," *Systems Engineering*, vol. 23, no. 2, pp. 189–210, 2020.

[69] R. Alguliyev, Y. Imamverdiyev, and L. Sukhostat, "Cyber-physical systems and their security issues," *Computers in Industry*, vol. 100, pp. 212–223, 2018.

[70] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *2011 international conference on wireless communications and signal processing (WCSP)*. IEEE, 2011, pp. 1–6.

[71] N. H. C. Guzman and A. G. Mezovari, "Design of iot-based cyber–physical systems: A driverless bulldozer prototype," *Information*, vol. 10, no. 11, p. 343, 2019.

[72] D. B. Parker, *Fighting computer crime: A new framework for protecting information*. John Wiley & Sons, Inc., 1998.

[73] J. Andress, *The basics of information security: understanding the fundamentals of InfoSec in theory and practice*. Syngress, 2014.

[74] B. R. Williams and J. Adamson, *PCI Compliance: Understand and implement effective PCI data security standard compliance*. CRC Press, 2022.

[75] R. C. Reid and A. H. Gilbert, "Using the parkerian hexad to introduce security in an information literacy class," in *2010 Information Security Curriculum Development Conference*, 2010, pp. 45–47.

[76] C. Paulsen, B. JM, and W. Bartol, "N.: Criticality analysis process model," Tech. rep, Tech. Rep., 2018.

[77] C. Schmittner, T. Gruber, P. Puschner, and E. Schoitsch, "Security application of failure mode and effect analysis (fmea)," in *Computer Safety, Reliability, and Security: 33rd International Conference, SAFECOMP 2014, Florence, Italy, September 10-12, 2014. Proceedings 33*. Springer, 2014, pp. 310–325.

[78] N. H. C. Guzman, J. Zhang, J. Xie, and J. A. Glomsrud, "A comparative study of stpa-extension and the ufoi-e method for safety and security co-analysis," *Reliability Engineering and System Safety*, vol. 211, p. 107633, 2021.

[79] M. Burmester, E. Magkos, and V. Chrissikopoulos, "Modeling security in cyber–physical systems," *International journal of critical infrastructure protection*, vol. 5, no. 3-4, pp. 118–126, 2012.

[80] J. Li, Y. Liu, T. Chen, Z. Xiao, Z. Li, and J. Wang, "Adversarial attacks and defenses on cyber–physical systems: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5103–5115, 2020.

[81] Q. Do, B. Martini, and K.-K. R. Choo, "The role of the adversary model in applied security research," *Computers & Security*, vol. 81, pp. 156–181, 2019.

Zhicong Sun received his B.E. degree in Communication Engineering from Harbin Institute of Technology, China, in 2020 and his M.Eng degree from the Department of Computer Science and Engineering at Southern University of Science and Technology, China, in 2023. He is currently a PhD student at Hong Kong Polytechnic University. His research interests include the safety and security of cyber-physical systems.

Yulong Ding received his B.A.Sc. and M.A.Sc. degrees in Chemical Engineering from Tsinghua University, China, in 2005 and 2008, respectively. He earned his Ph.D. degree in Chemical Engineering from The University of British Columbia, Canada, in 2012. He is currently a Research Associate Professor at the Department of Computer Science and Engineering at Southern University of Science and Technology. His primary research interests include the industrial Internet of Things and low-power wide area networks (LPWANs).

Ke Pei received his Ph.D. degree in Communication and Information Systems from Xidian University in 2002. He is currently the Chief AI Technical Expert at the RAMS lab in Huawei's Munich Research Center. His responsibilities include idea initiation, prototype design and verification of relevant capabilities, and defining intelligent architecture and solutions for next-generation systems, cloud, and automotive. His research interests include time series data intelligent processing (AIOps), reliable deep learning, and safe decision learning for autonomous systems. Prior to joining Huawei, Dr. Pei was a Distributed Member of Technical Staff (DMTS) at Lucent Bell Labs and was accepted as a member of the Alcatel-Lucent Technical Academy (ALTA). He has over 10 years of solid experience in new product development and software architecture for cloud/NFV solutions, distributed databases, intelligent network applications, and data mining/analysis.

Shuang-Hua Yang (Senior Member, IEEE) received his B.S. degree in Instrument and Automation and his M.S. degree in Process Control from the China University of Petroleum (Huadong), Beijing, China, in 1983 and 1986, respectively. He earned his Ph.D. degree in Intelligent Systems from Zhejiang University, Hangzhou, China, in 1991. He is currently the Director of the Shenzhen Key Laboratory of Safety and Security for Next Generation of Industrial Internet at Southern University of Science and Technology, China, and also serves as the Head of the Department of Computer Science at the University of Reading, UK. His research interests include cyber-physical systems, the Internet of Things, wireless network-based monitoring and control, and safety-critical systems. He is a Fellow of IET and InstMC in the UK and an Associate Editor of IET Cyber-Physical Systems: Theory and Applications.