International Journal of Engineering and Technology Innovation, vol. 15, no. 3, 2025, pp. 374-387

System-Theoretic Analysis of Hazard Causal Factors and Scenario Development for Complex Systems

Masakazu Takahashi[1,*], Meiting Liu[1], Yunarso Anang[2], Yoshimichi Watanabe[1]

[1]Department of Computer Science and Engineering, University of Yamanashi, Yamanashi, Japan

[2]Department of Statistical Computing, Politeknik Statistika STIS, Jakarta, Indonesia

Received 19 February 2025; received in revised form 19 July 2025; accepted 24 July 2025

DOI: https://doi.org/10.46604/ijeti.2024.14864

Abstract

This paper proposes a system-theoretic process and analysis (STPA) for complex systems (STPA_CS) to analyze the hazard factors arising from interactions between multiple components in a system. STPA_CS is realized by adding the following features to STPA: (1) using class diagrams to define control signal inputs and outputs of components, (2) defining component configurations by using composite configuration diagrams, (3) clarifying hazard occurrence processes by tracing the control signal exchange paths, and (4) facilitating seamless hazard analysis by detailing the class diagrams and repeating the hazard analysis. When STPA_CS and STPA are applied to a chiller system, STPA_CS clarifies fifty-one hazard factors and scenarios, while STPA clarifies thirty hazard factors and scenarios. This represents an improvement of 60% in the analysis results and demonstrates the advantages of STPA_CS.

Keywords: system-theoretic process and analysis, hazard analysis, hazard causal factor, hazard scenario

1. Introduction

Many mechanisms have been developed to produce industrial products or provide various services and are typically composed of hardware, software, operators, and their surrounding environment (components). As the configurations and functions of these systems become more complex to meet the growing needs of customers, undesirable events and accidents that were not anticipated during the design phase can occur, which must be prevented. In particular, the prevention of accidents arising from interactions between components (e.g., exchange of control signals and data) has become an issue, in addition to the prevention of accidents arising from failures of individual components. In recent years, to avoid accidents, it has become necessary to prevent a system from entering a state that would be a precursor to an accident (hazard). For this purpose, it is essential to analyze the hazard factors (called hazard analysis) and then employ suitable countermeasures.

System-theoretic process analysis (STPA) was proposed by Leveson [1] as a method for hazard factor analysis based on interactions. STPA can detect hazards caused by interactions between system components and is used for hazard analysis of industrial products in various domains.

In the automotive domain, by using hazard and operability studies (HAZOP) [2] and failure mode and effects analysis (FMEA) [3], in addition to using the functional resonance analysis method (FRAM) [4] and STPA, Sun et al. [5] identified the risks of automatic emergency braking in autonomous vehicles (AV) and proposed a method to avoid loss of functions by combining the strengths of each method to systematically identify such risks. Xing et al. [6] combined STPA and a finite state

* Corresponding author. E-mail address: mtakahashi@yamanashi.ac.jp

machine (FSM) to perform hazard analysis of the driving functions of an AV equipped with multiple automatic modes. By modeling the car and the environment, they were able to define unpredictable AV behaviors in detail and, consequently, identify more hazards.

In the railway transportation domain, Tonk and Boussif [7] surveyed the use of STPA for hazard analysis in railway systems. The applicable knowledge from system-theoretic accident and processes (STAMP; explained in section 2) and STPA was comprehensively reported to railway workers. Zhang et al. [8] conducted an STPA-based hazard analysis to develop safety countermeasures for the Intelligent Railway Driving Assistance System (IRDAS). The results were reflected in the development of IRDAS for the Mass Transit Railway in Hong Kong. Takahashi et al. [9] added functions to STPA, such as a method to describe hierarchical component structure diagram (CSD; explained in Section 2), a method to describe component behavior, and a method to trace control actions (CA; explained in Section 2). This extended STPA can perform hazard analysis in detail. As a result, when it was applied to a level crossing control system, it was able to detect appropriate hazard causal factors (HCF) and develop comprehensive hazard scenarios.

In the aerospace domain, Chen et al. [10] applied STPA to clarify the potential hazards of unmanned aerial vehicle (UAV) takeoff. The results indicated that, in addition to UAV component failure, the loss of CAs and feedback data (FBD; explained in Section 2) between the components, as well as operator errors, were hazard factors. Fugivara et al. [11] clarified the hazards associated with the launch of observation rockets by applying STPA to launch data and proposed various safety measures.

In the process plant domain, Sultana et al. [12] performed a hazard analysis of the process of transferring liquid natural gas (LNG) between ships. According to their results, STPA was able to detect more hazard factors than HAZOP. Shin et al. [13] improved the safety level of a nuclear reactor by using STPA, focusing on the operator's interaction with an Advanced Power Reactor 1400's instrument and control system in an emergency reactor shutdown scenario. Yousefi and Hernandez [14] applied STPA to the hazard analysis of process plants and found that more detailed hazard analysis could be performed by using both HAZOP and STPA.

In recent years, hazard analysis using artificial intelligence has been attempted, and Watanabe et al. [15] investigated the quality of the AI function. Charalampidou et al. [16] performed hazard analysis of a search-and-rescue drone system by using a large language model (LLM). In this method, the system's purpose, goal, component structure, and operation were first input into the LLM, followed by the STPA procedure. The LLM then defined unsafe control actions (UCA; explained in Section 2) and subsequently developed relevant hazard scenarios. In terms of defining UCAs and developing hazard scenarios, the LLM's hazard analysis results were better than those of an analyst.

STPA is a method for analyzing system hazards throughout the system development process. It is particularly suitable for conducting hazard analyses during the system planning, requirements definition, and preliminary design phases. However, problems arise after the detailed design phase. This is due to the increased number of newly developed components and the increased number of connections of components that need to be analyzed. The shortcomings of STPA are as follows:

[Problem 1] STPA does not define a method to describe the detailed component structure of the system and the behaviors of the components.

[Problem 2] STPA does not define a method for identifying hazard factors.

[Problem 3] STPA does not define a method for developing hazard scenarios.

To solve these problems, a method of hazard factor analysis based on the interaction of components is required. This paper proposes system-theoretic process analysis for complex systems (STPA_CS) by adding several features to STPA, which is a representative hazard analysis method. Additionally, a method for developing hazard scenarios is proposed by using the results of STPA_CS. STPA_CS enables causal analysis of hazards caused by interactions between the components that

compose a complex functional system, as well as the development of hazard scenarios, thereby improving its safety. STPA_CS adds the following features to STPA to solve the above-mentioned problems.

[Solution 1] Class diagrams of Unified Modeling Language (UML) [17] are used to describe a component's behaviors after it receives a CA. In addition, by using the relations between these classes and composite structure diagrams, the connections between components and invocations of CAs are described. Thus, Solution 1 addresses Problem 1.

[Solution 2] The occurrence path of the CA is defined by tracing the invocation relations of the CAs between the components. The obtained path is examined to judge whether the CA causes a hazard or not when the guide words for detecting HCFs (explained in Section 2) occur. Thus, Solution 2 addresses Problem 2.

[Solution 3] Based on the occurrence path obtained in Solution 2, hazard scenarios can be developed by extracting partial paths from the occurrence of the HCFs to the output of the CA that causes the hazard and arranging them. Thus, Solution 3 addresses Problem 3.

STPA_CS is outlined as follows. First, the accident and hazards to be analyzed are defined. Second, the components that comprise the system, the relations between components, the CAs of the component, and the behaviors associated with inputting and/or receiving CAs and FBDs are defined using class diagrams. Third, it determines whether a CA causes a hazard by using the guide words for detecting CAs that cause the hazard, and the CAs to be analyzed are identified. Fourth, the paths that generate UCAs are clarified by tracing the invocation relations of the CAs. Fifth, it examines whether the UCAs are outputted when the items of the guide words for detecting HCFs occur in the components, CAs, and FBD on the occurrence path. If the hazard occurs, then the items corresponding to the guide words for detecting HCFs are considered to be the hazard factors. Finally, the hazard scenarios are developed by extracting partial paths from the point of occurrence of the hazard to the point of occurrence of the UCA. This analysis process is repeated based on the detailed design results as system development progresses. The HCF analysis can be realized throughout the system development process.

The remainder of this paper is organized as follows. Section 2 explains various hazard analysis methods, including STPA. Section 3 explains the proposed STPA_CS. Section 4 describes the application and evaluation of STPA_CS. Section 5 presents the conclusion and outlines future work.

2. STAMP and STPA

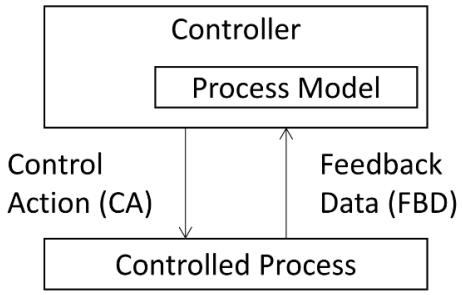

Fig. 1 STAMP model

This section explains the accident model used in the STPA and the STPA procedure. First, Fig. 1 depicts the STAMP accident model [18-19]. This model consists of a component that controls the other components (hereinafter, the controller) and a component that is controlled by the controller (hereinafter, the controlled process). These components send and receive CAs and FBDs by predicting the states of the other components. This is called interaction. As the system operates for a long period of time, it accumulates timing gaps of CAs and calculation errors of FBDs, so there is a difference between the state of the controlled process predicted by the controller (hereinafter, process model) and the actual state of the controlled process. The STAMP model assumes that a hazard occurs when this difference exceeds a tolerable range. Next, the procedure for

performing STPA is explained.

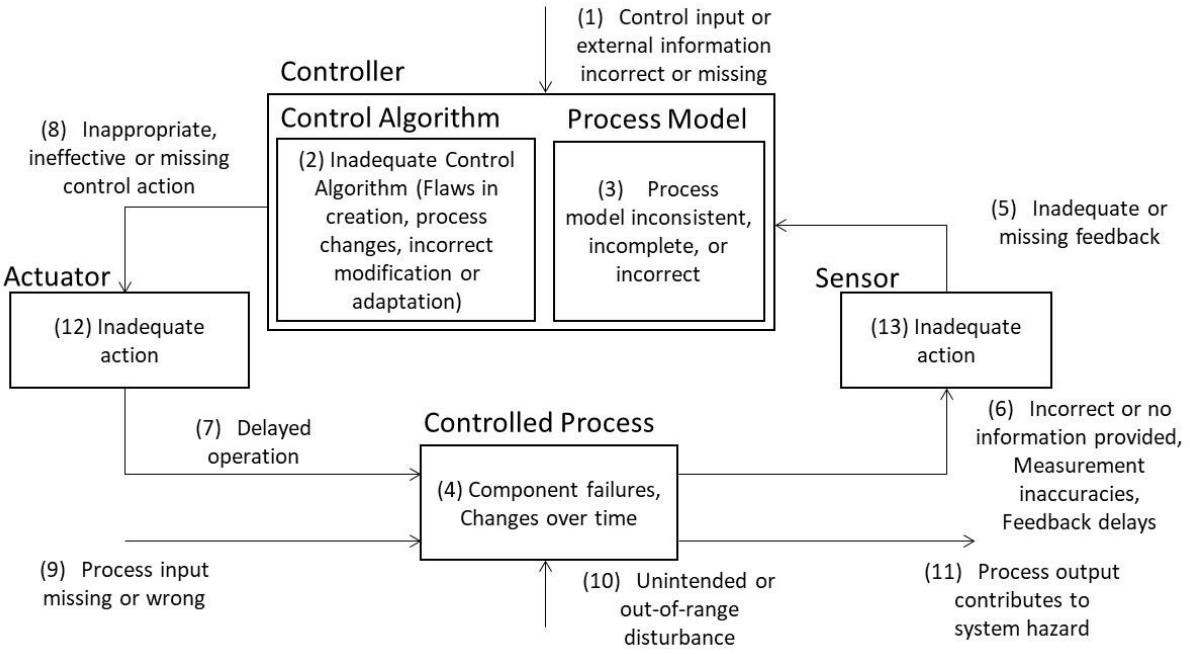

Fig. 2 Guide words for detecting hazards

Step 0.1: Identify the accident, hazards, and safety constraints.

An accident, hazards, and safety constraints are identified and analyzed.

Step 0.2: Develop a control structure diagram.

The components of the system, CAs, and FBDs to be analyzed are defined. This information is organized into a CSD. Fig. 1 and Fig. 2 show examples of CSDs.

Step 1: Extract unsafe control actions.

A hazard is judged to occur when the CA becomes a condition listed below. This condition is called a guide word. The combination of a guide word and a CA that causes a hazard is called a UCA. For example, when the combination guideword "providing" and "rotate motor CA" causes a hazard, "providing rotate motor CA" is a UCA. The guide words used for this judgement are called guide words for detecting UCAs (GWDUs):

- Not Providing causes a hazard (Not Providing).
- Providing causes a hazard (Providing).
- Too Early, Too Late, and Wrong Order cause a hazard (Too Early, Too Late, Wrong Order).
- Stopping Too Soon or Applying Too Long causes a hazard (Stopping Too Soon, Applying Too Long).

Step 2: Identify HCFs of unsafe control actions.

The factors that cause a UCA are called HCFs. Step 2 identifies the HCFs of a UCA. First, the control loop in the CSD that includes the UCA is identified. It is investigated whether each item in this control loop causes UCA when each item becomes the conditions indicated by the guide words. These guide words are called guide words for detecting hazards (GWDHs). Fig. 2 shows the GWDHs in the system. If the UCA is issued when the conditions indicated by GWDHs occur, the conditions are considered HCFs. After the HCFs are determined, the corresponding hazard scenarios are described. These hazard scenarios show the process from the occurrence of the HCFs to the manifestation of the UCA, thereby improving

understanding of hazard occurrence and helping plan preventative countermeasures.

3. Solutions to the Problems of STPA

This section explains STPA_CS, which can be used for hazard analysis of systems with complex structures and functions. Section 3.1 explains the solutions that solve the problems of STPA, as follows: (1) a method to describe a component's structure and behavior by using class and composite structure diagrams, (2) a comprehensive procedure for detecting UCA occurrence paths and HCFs by tracing the exchange path of CAs, and (3) a comprehensive method for development of hazard scenarios that uses UCA occurrence paths. Section 3.2 defines the STPA_CS procedure, which can be used to perform appropriate hazard factor analysis.

3.1. Solutions to the problems of STPA

This section explains Solutions 1-3, which solve the problems of STPA described in Section 1. Section 3.1.1. describes a method to describe component behaviors. Section 3.1.2 describes a concrete procedure to detect HCFs. Section 3.1.3 describes a method to develop hazard scenarios.

3.1.1. Solution 1: Description of component behaviors and construction using class diagram and composite structure diagrams

The CSD of STPA can be used to describe the input and output relations of CAs and FBDs between the components, but it cannot describe the behavior when a CA is received. Therefore, STPA_CS describes this behavior by using the UML class diagram. Here, the correspondence between CSDs and class diagrams is explained. The components of a CSD represent hardware, software, person, and environment of the target system. These components are considered classes in UML class diagrams. CAs of a CSD represent the instructions exchanged between components, and CA exchanges represent interactions between components. CAs are considered as the methods of a class, and CA interactions are considered as relations between classes. The FBDs of a CSD are the values returned as the results of CA interactions and are considered return values of the CA. Consequently, FBDs are considered return values of class methods. Additionally, the data shared between CAs in a component is required to describe CA content details. However, CSDs cannot describe shared data. Therefore, in STPA_CS, this function is realized using UML class attributes.

Fig. 3 shows the correspondence between a CSD and a class diagram. For example, method_A_1, which is a CA sent by actor_A to component A, corresponds to method_A_1 in class A. In general, UML does not describe actors in class diagrams, but STPA_CS describes actors as a class because the system includes operators and equipment, which are strongly related to hazard occurrence.

In general, as the number of components in a CSD increases, the number of interactions between components increases, while the readability of the CSD decreases. Consequently, interactions that should be investigated could be neglected. Therefore, it is necessary to take countermeasures to improve readability and avoid neglecting interactions by integrating components into one group with sufficient granularity. Therefore, a CSD is not suitable for describing the complex structures of systems consisting of many components.

STPA_CS defines a component's structure by using the relations in the class diagram. When there are many components in the system, related components may be summarized in the structured classifier of a composite structure diagram. A composite structure diagram is a type of UML diagram that represents a group of classes and other CSDs. It can also define the inner data structures and interfaces. This mechanism describes class structures hierarchically and improves readability. When using a hierarchical CSD, the CAs and FBDs exchanged between classes and CSDs must be strictly consistent. All the

CAs and FBDs input into a composite structure diagram must also be inputted into the classes or other composite structure diagrams. Meanwhile, all the CAs and FBDs outputted from the composite structure diagram must be outputted to other classes or other composite structure diagrams. This consistency must be maintained even as the number of layers in a composite structure diagram increases.

Fig. 4 shows the hierarchical CSD using both a class diagram and a composite structured diagram. In this case, the system consists of classes A, B, and C. Class B is described using a composite structure diagram, and it consists of classes B1 and B2. The methods in class B are executed through the sub-classes. For example, class A invokes CA_{B11} in class B, which consists of classes B1 and B2, and class B1 contains CA_{B11}. As a result, FBD_{B11} is returned to class A through class B as the return value of CA_{B11}. Consistency must be maintained between the classes in the upper and lower layers. Therefore, Solution 1 solves Problem 1.

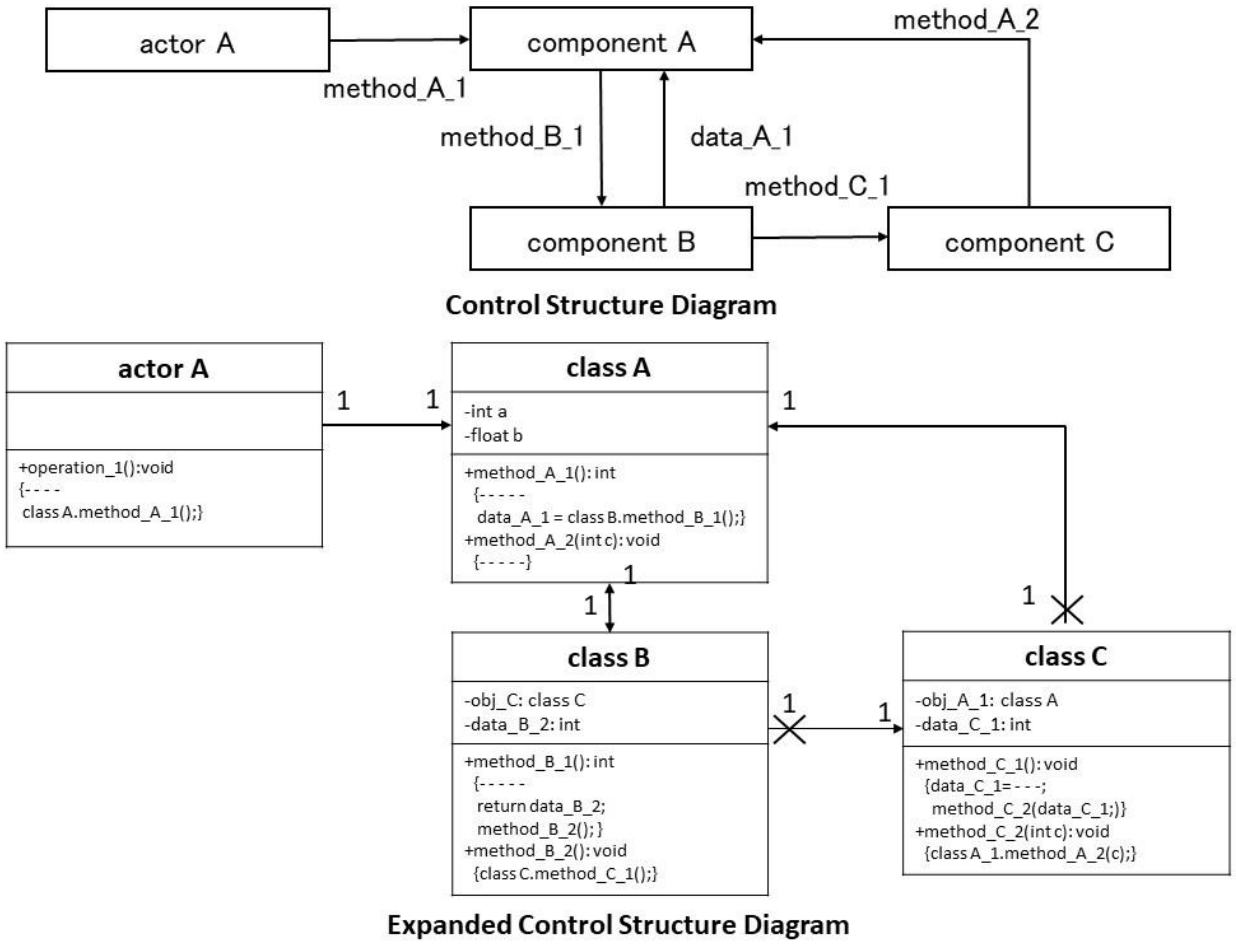

Fig. 3 Correspondence relationships between CSD and class diagrams

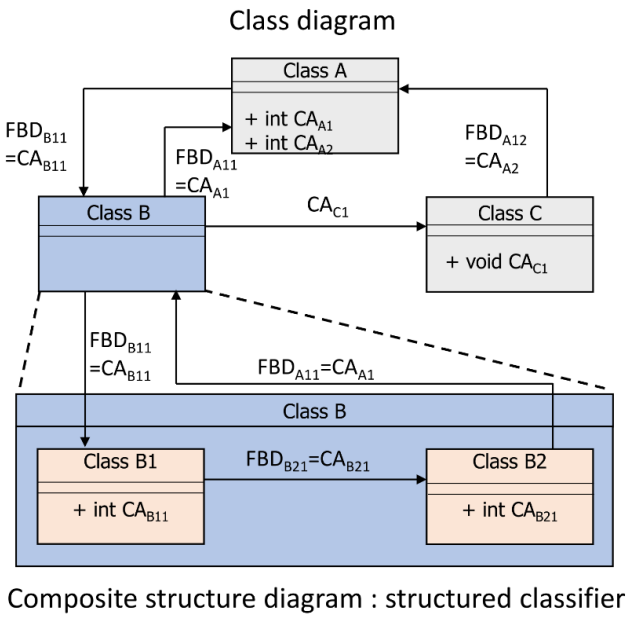

Fig. 4 Hierarchical class diagram

3.1.2. Solution 2: Systematic method for hazard factor analysis

In STPA, a CA that causes a hazard when it becomes the condition indicated by a GWDU is considered a UCA. Then, the corresponding HCFs are identified by investigating whether the UCA occurs when the item indicated by a GWDH occurs in components, CAs, or FBDs included in the control loop containing the UCA. The UCA and HCFs are then used as inputs to create a hazard scenario. However, the procedure for creating such a scenario is undefined. Although STPA can detect HCFs mainly through the interaction between two components, HCFs due to interactions between many components may also exist in real systems. With STPA, it is difficult to detect such HCFs.

STPA_CS provides a method for detecting the HCFs caused by interactions between components in complex configurations, as well as a clear procedure to create hazard scenarios (described in Section 3.1.3). The HCF detection procedure of STPA_CS is as follows. First, similar to STPA, STPA_CS uses GWDUs to identify CAs that may cause a hazard. These CAs are considered candidates for UCA. Second, the exchange paths between components are traced, starting from the candidate UCAs, to identify their occurrence paths using the program slicing technique [20]. Third, the method determines whether a candidate UCA becomes a UCA when the item indicated by a GWDH occurs in components, CAs, or FBDs on the occurrence path of the candidate UCA. Confusing candidate UCAs with UCAs in this procedure leads to ambiguity in the procedures for HCF detection and hazard scenario creation. Therefore, this study refers to candidate UCAs as CA_{mch} to clearly distinguish them from UCAs.

The UCA occurrence path across multiple system components is obtained as follows. First, CA_0, which can become CA_{mch} by using GWDUs, is decided and taken as a starting point. Second, CA_1, which calls CA_0 (CA_{mch}), is searched from all the methods in all the classes. Third, CA_2, which calls CA_1, is searched from all the methods in all the classes. This process is repeated until no CA is left to call, and the last CA (CA_n) is considered to be the endpoint. The string in reverse order from the starting to ending points (that is, CA_n, …, CA_2, CA_1, CA_0 (CA_{mch})) is considered to be the occurrence path of CA_0 (CA_{mch}). When the item shown in the GWDH occurs, it is assessed whether CA_{mch} becomes a UCA. If it does, then the corresponding GWDHs are considered HCFs.

Fig. 5 shows the procedures for detecting the UCA occurrence path and HCFs. If composite structure diagrams are used, then they are converted into one large-scale CSD by merging multiple layers into a single layer. The dotted arrow in this figure indicates the CA to be traced. It is assumed that method_A_2 is CA_{mch}. Method_A_2 is called by method_C_2 in class C.

Method_C_2 is called by method_C_1 in class C. Method_C_1 is called by method_B_2 in class B. Method_B_2 is called by method_B_1 in class B. Method_B_1 is invoked by method_A_1 in class A. Finally, Method_A_1 is called by operation_1 in actor A. Given that no traceable calling relations exist, the tracking is complete. As a result, the occurrence path of CA_{mch} is as follows: operation_1 → method_A_1 → method_B_1 → method_B_2 → method_C_1 → method_C_2 → method_A_2. Finally, when items shown in GWDH occur on the occurrence path, the method assesses whether CA_{mch} becomes a UCA. In the following, it is assumed that the inappropriate algorithm of method_B_2 in class B is the HCF. As a result, the UCA, HCFand CA_{mch} occurrence paths are clarified. As a result, Solution 2 solves Problem 2.

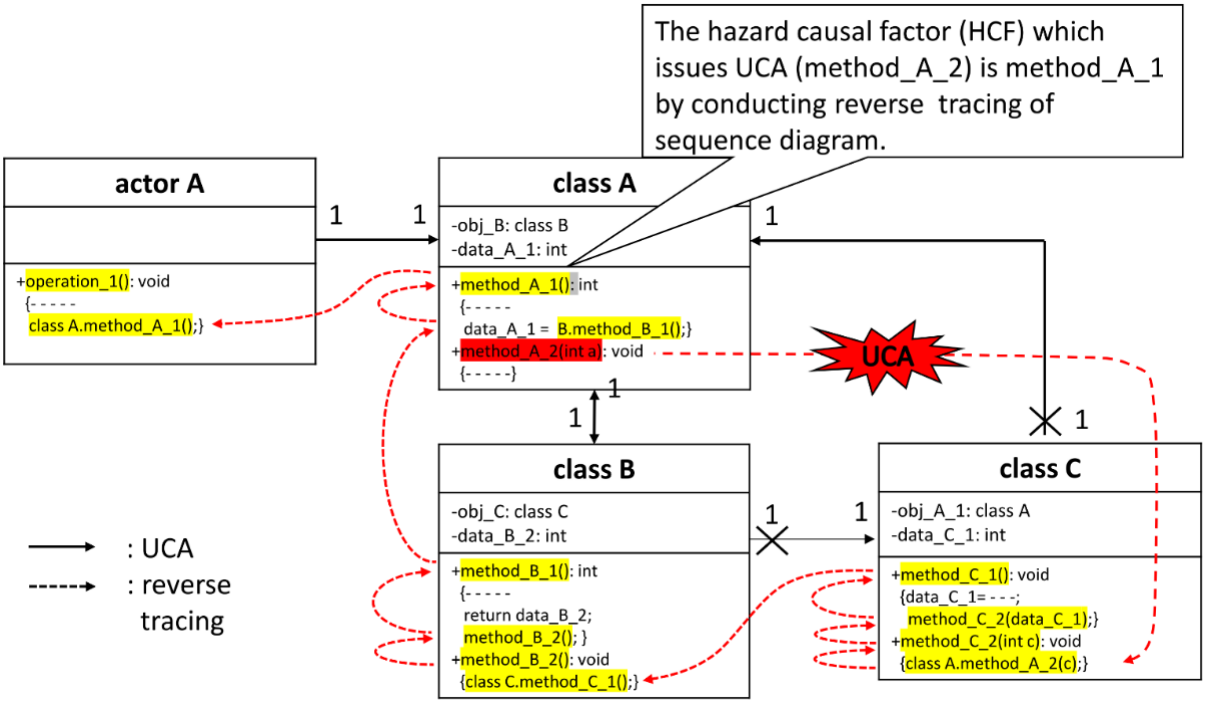

Fig. 5 Definition of a process from UCA occurrence to HCF occurrence

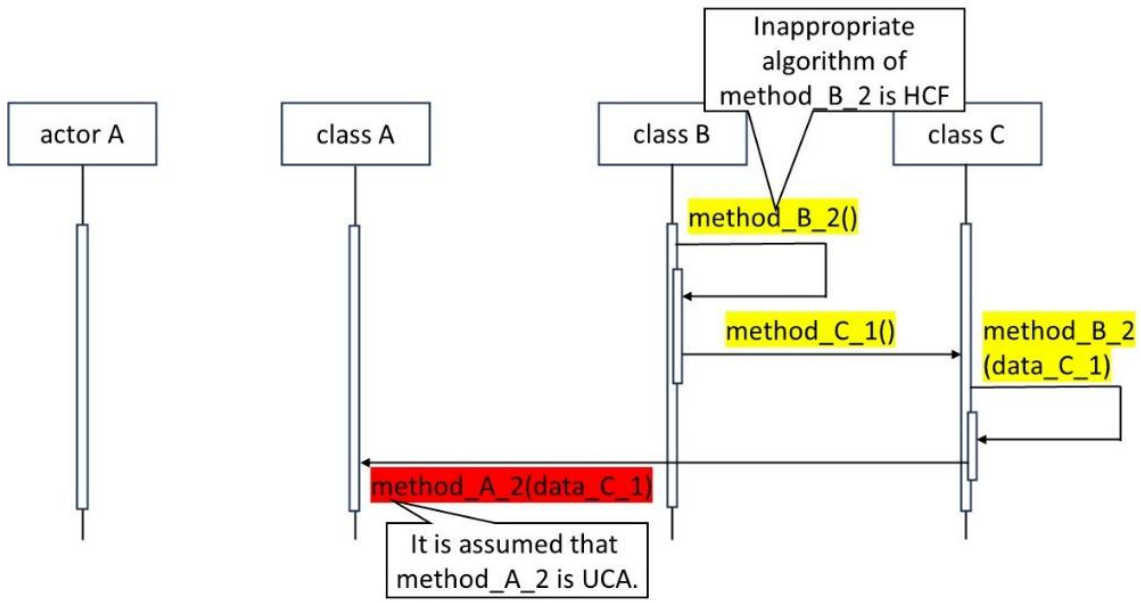

Fig. 6 Sequence diagram from HCF occurrence to UCA occurrence

3.1.3. Solution 3: Hazard scenario development method

STPA_CS develops hazard scenarios by extracting parts of the CA_{mch} occurrence path between HCF occurrence and

CA_{mch} (UCA) occurrence. When the understandability and readability of a hazard scenario comprising multiple branches and repeat structures are poor, it is described using UML sequence diagrams. In the case depicted in Fig. 5, the hazard scenario is as follows: method_B_2 in class B → method_C_1 in class C → method_C_2 in class C → method_A_2 (UCA) in class A. For reference, Fig. 6 shows the sequence diagram corresponding to this hazard scenario. As a result, Solution 3 solves Problem 3.

3.2. Procedure of STPA_CS

STPA_CS, which can be applied to systems with many components and complex functions, is realized by combining Solutions 1-3. Fig. 7 shows the steps of STPA_CS. In Step 0.1, the accident to be analyzed is identified. The associated hazards and safety constraints are also identified. In Step 0.2, methods of classes and the relations between classes are defined by developing class diagrams and composite structure diagrams instead of CSDs, as opposed to CSDs alone in STPA. Step 1 decides the CA_{mch} that causes hazards when used with GWDHs. In Step 2.1, the CA_{mch} occurrence path is obtained by tracing the invocation relations between CAs in class diagrams and composite structure diagrams. In Step 2.2, it is assessed whether CA_{mch} becomes a UCA by applying the items shown in the GWDHs to the items on the CA_{mch} occurrence path. If CA_{mch} becomes a UCA, then the GWDH items are considered HCFs. Finally, the partial path from the occurrence of HCF to the occurrence of HCFs is extracted from the CA_{mch} occurrence path, and hazard scenarios are developed by lining up the partial paths in reverse order. If the content of the hazard scenario is complex, then UML sequence diagrams are created.

Step 0.1 : Define an accident and a hazard analyzed
Step 0.2 : Develop a class and composite structure diagram
Step 1 : Identify an unsafe control action (UCA) which causes the hazard
Step 2.1 : Decide an CA_{mch} occurrence path by tracing the invocation relationships between CAs
Step 2.2 : Investigate whether items of GWDHs cause UCA or not, and decide HCFs
Step 3 : Develop hazard scenarios (and sequence diagrams, if necessary)

Fig. 7 Steps of STPA_CS

4. Application and Evaluation of STPA_CS

This section describes the application of STPA_CS and the results. Section 4.1 describes the chiller system to which STPA_CS was applied. Section 4.2 presents the results of this application. By applying STPA_CS and STPA to a hazard in which the chiller system freezes, STPA_CS was able to detect seven UCAs and create fifty-one hazard scenarios, while STPA was able to detect seven UCAs and thirty HCFs and create thirty hazard scenarios. Section 4.3 evaluates this application. STPA_CS detected more HCFs and developed hazard scenarios because it followed a concrete analysis procedure without omission, detected HCFs across multiple components, and followed a well-defined hazard scenario development procedure.

4.1. Overview of target system

STPA_CS was used for hazard analysis of the chiller system. Fig. 8 presents an overview of the chiller system. It removes heat from a refrigerant in its vapor compression cycle and uses this heat to maintain the temperature of industrial equipment, measuring equipment, and chemical equipment. The chiller system consists of a chiller, air-handling unit (AHU), cooling tower, pumps, valves, water filter, and pipes. The chiller itself consists of an evaporator, a compressor, a condenser, and an expansion valve. The refrigeration system operates as follows. First, the refrigerant is cooled using water that is cooled in the cooling tower. Second, the refrigerant cools the chilled water sent to the AHU by vapor heat as the refrigerant evaporates in the evaporator. Third, the air cooled by the chilled water is blown into the building by the fan and cools the building. This chiller system was used by Shaharyar et al. [21] to study protection against cyberattacks.

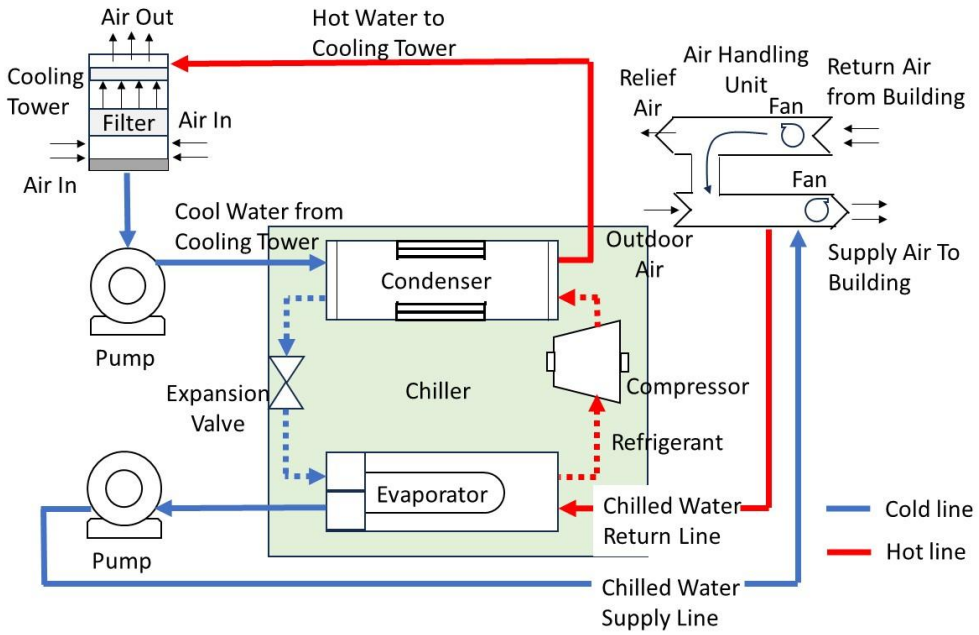

Fig. 8 Overview of the chiller system (Fig. 2, Ref. [21])

4.2. Results of hazard analysis for the chiller system

The hazard analysis was performed following the steps of STPA_CS. In Step 0.1, an accident and a hazard were defined. The accident was defined as physical damage to the chiller system, and the hazard was defined as freezing of the chiller system (the "temperature too low"). In Step 0.2, the class diagrams of the chiller system were developed. Fig. 9 presents an overview of the class diagrams (excerpt). In Step 1, the UCAs were investigated. The results indicated that the "temperature too low" hazard was caused by seven UCAs. Now, the case in which "compressor rotation speeds up CAs" was not provided (Not Providing "compressor rotation speeds up CAs") is discussed. In Step 2.1, the occurrence path of CA_{mch} was clarified. In Fig. 9, the red dotted arrow indicates the obtained path. In Step 2.2, it was investigated whether the UCA occurred when the items shown in the GWDH occurred. As a result, the three HCFs listed below were detected, and the case in which the equipment caused a failure was excluded from the HCFs because the hazard certainly occurred:

(1) Set temperature is incorrect (external information incorrect);

(2) The chiller controller algorithm is inadequate (inadequate algorithm); and

(3) The current temperature received from the evaporator is incorrect (incorrect control input).

The partial path from the occurrence of the HCFs to the occurrence of the UCA was extracted from the CA_{mch} occurrence path, and the following hazard scenarios were developed:

(1) The chiller controller receives a "too low temperature setpoint" from the chiller system and sends the incorrect temperature settings to the chiller controller. The compressor speed increases excessively, and the temperature of the chiller system becomes too low (external information incorrect).

(2) The chiller controller's temperature control algorithm is erroneous. As a result, the chiller controller indicates too many working CAs to the compressor, and the chiller temperature decreases excessively (inadequate algorithm).

(3) The evaporator receives the current temperature, which is higher than the actual temperature because of the disturbance. As a result, the chiller controller's temperature control algorithm provides too many CAs to the compressor, and the temperature of the chiller system becomes too low (control input incorrect).

For reference, Fig. 10 depicts the sequence diagram corresponding to the hazard scenario described above. These are "the

temperature of the chiller becomes too low" scenarios caused by other HCFs on the same occurrence path. The HCFs are shown in the dialogue bubbles in Fig. 10.

Finally, the overall results of the hazard analysis are as follows. STPA_CS detected fifty-one HCFs and developed fifty-one hazard scenarios for the seven UCAs detected in Step 1.1 as a result of the hazard analysis. When an engineer with five years of experience performed the hazard analysis for the same accident and hazards, the engineer detected seven UCAs as well as thirty HCFs and developed thirty hazard scenarios. All the UCAs detected and hazard scenarios developed by the engineer were included in the UCAs and hazard scenarios outputted by STPA_CS. The details of the twenty-one differences in the HCFs and hazard scenarios were ten misapplications of GWDHs and eleven hazards that were caused between multiple classes. STPA was not able to detect these HCFs. These results confirmed the superiority of STPA_CS.

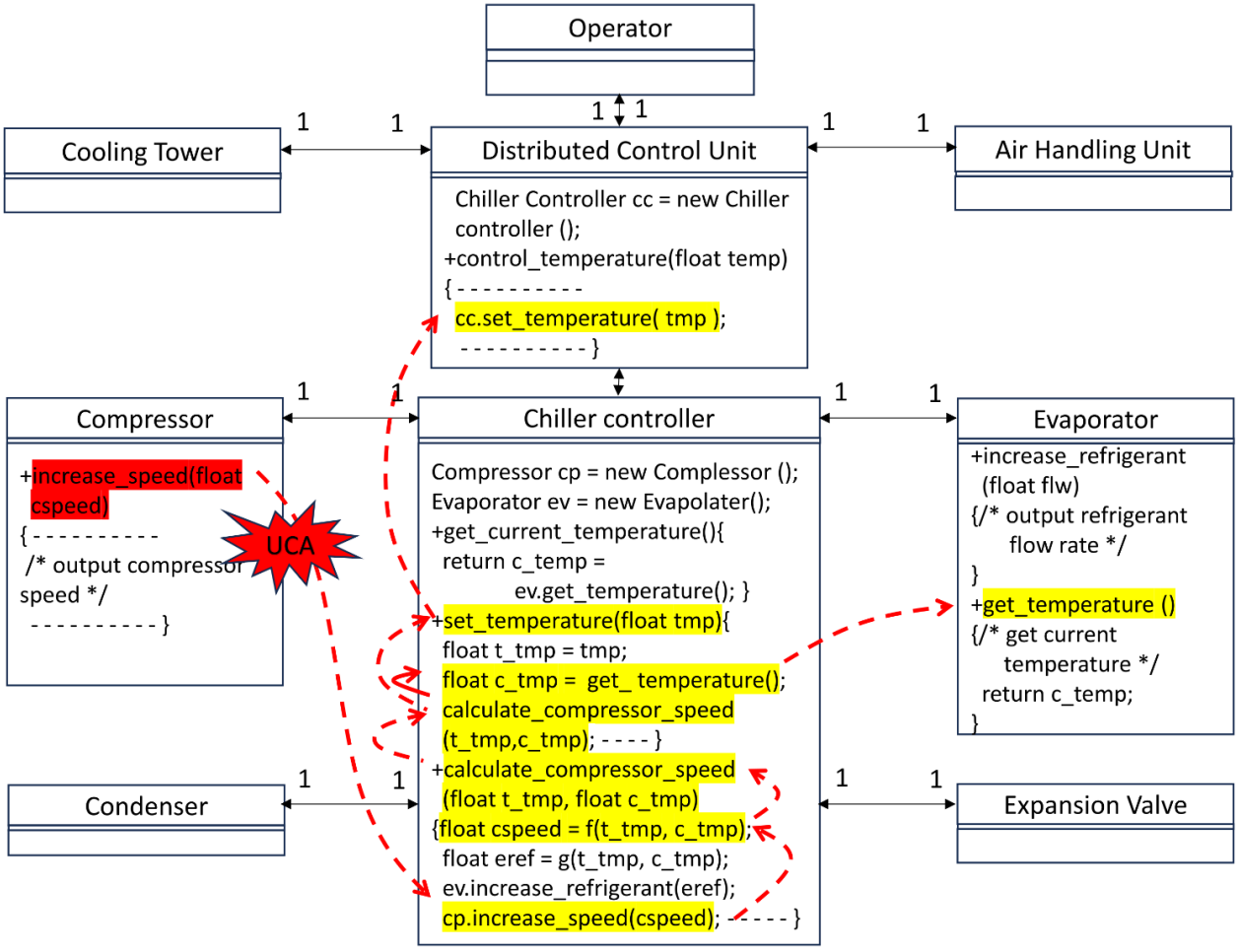

Fig. 9 Class diagram of chiller system (excerpt)

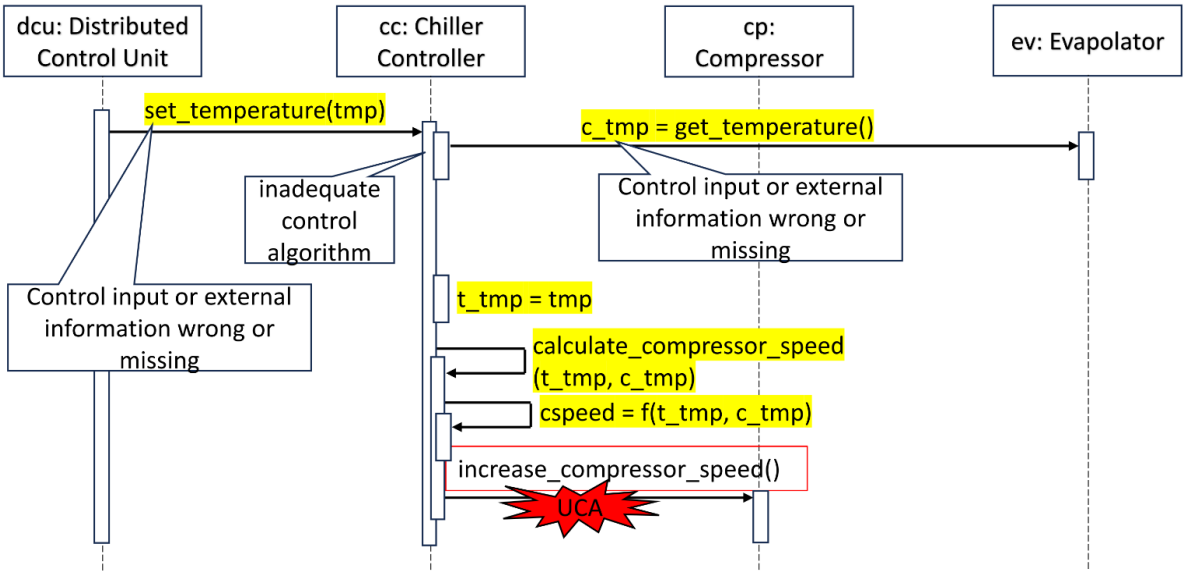

Fig. 10 Sequence diagram corresponding to "the temperature of the chiller becomes too low" hazard scenario

4.3. Comprehensive evaluation of STPA_CS

When STPA_CS was used for hazard analysis of the actual system, many additional HCFs were detected, and hazard scenarios were developed compared to those obtained using STPA. The reasons for this difference are as follows. (1) Since STPA_CS concretely defined the HCF detection procedure, the GWDH application was not omitted. (2) STPA_CS was able to detect the HCFs among the multiple classes. (3) STPA_CS concretely defined the procedure for hazard scenario development. As a result, STPA_CS was able to perform appropriate hazard analysis regardless of the analyst's skills and experience.

The limitations of these systems are as follows. (a) Since the task of tracking the HCFs in plural classes is complex, errors may occur if it is performed manually. (b) The identification of HCFs from the list of candidate HCFs is time-consuming. (c) The development of hazard scenarios, which are described using the actual names of the system components and their behaviors, is a manual task. (4) Reviewing the numerous hazard scenarios developed by STPA_CS is time-intensive. It is expected that these limitations will be overcome by developing supporting tools.

5. Conclusion

This paper proposed STPA_CS for factor analysis of the hazards caused by systems with complex functions and structures. To this end, the following features were added to STPA:

(1) The functions of the system component can be described in detail by using the methods of a class.

(2) The connections between the system components can be described by using the relations between the classes. In addition, the classes can be described hierarchically by using composite structure diagrams.

(3) HCFs can be systematically clarified by tracing the calling relations of the methods between classes.

(4) As system development progresses, the hazard analysis can be detailed step by step by using class diagrams. In addition, hazard analysis can be performed seamlessly.

By using these features, STPA_CS was able to solve Problems 1-3 described in Section 1 and was able to systematically detect the HCFs of the system with complex functions and structures, which STPA could not detect. When STPA_CS was applied to the chiller system's "too low temperature" hazard, it detected fifty-one HCFs and developed fifty-one hazard scenarios, while STPA detected thirty HCFs and developed thirty hazard scenarios. Accordingly, STPA_CS improved the

analysis result by 60%. In particular, STPA_CS detected HCFs that occurred in the components without direct interaction. These results confirmed the superiority of STPA_CS.

In future work, STPA_CS will be applied to diverse systems, and the results will be fed back to it. Although STPA_CS can comprehensively detect HCFs, manual analysis is time-consuming and prone to omissions. The development of support tools for STPA_CS will contribute to more efficient hazard analysis.

Acknowledgment

This research was supported by a Grant-in-Aid for Scientific Research (C) of the Japan Society for the Promotion of Science, grant number 22K04616.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] N. G. Leveson, Engineering a Safer World, Cambridge: The MIT Press, 2011.
[2] IEC 61882:2016 Hazard and Operability Studies (HAZOP Studies) - Application Guide, 2nd ed., IEC Standard 61882, 2016.
[3] IEC 60812:2018 Failure Modes and Effects Analysis (FMEA and FMECA), 3rd ed., IEC Standard 60812, 2018.
[4] E. Hollnagel, FRAM: The Functional Resonance Analysis Method, Boca Raton: CRC Press, 2012.
[5] L. Sun, Y. Li, and E. Zio, "Comparison of the HAZOP, FMEA, FRAM, and STPA Methods for the Hazard Analysis of Automatic Emergency Brake Systems," ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering, vol. 8, no. 3, article no. 031104, 2022.
[6] X. Xing, T. Zhou, J. Chen, L. Xiong, and Z. Yu, "A Hazard Analysis Approach based on STPA and Finite State Machine for Autonomous Vehicles," Proceedings of IEEE Intelligent Vehicles Symposium (IV), IEEE, pp. 150-156, 2021.
[7] A. Tonk and A. Boussif, "Application of Systems Theoretic Accident Model and Processes in Railway Systems: A Review," IEEE Access, vol. 12, pp. 99872-99893, 2024.
[8] S. Zhang, T. Tang, and J. Liu, "A Hazard Analysis Approach for the SOTIF in Intelligent Railway Driving Assistance Systems Using STPA and Complex Network," Applied Science, vol. 11, no. 16, article no. 7714, 2021.
[9] M. Takahashi, D. Morimoto, Y. Anang, and Y. Watanabe, "A Proposal of Hazard Analysis Method using Structured System Theoretical Process Analysis," SICE Journal of Control, Measurement, and System Integration, vol. 16, no. 1, pp. 192-202, 2023.
[10] J. Chen, Y. Lu, S. Zhang, and P. Tang, "STPA-based Hazard Analysis of a Complex UAV System in Take-off," Proceedings of IEEE International Conference on Transportation Information and Safety (ICTIS), IEEE, pp. 774-779, 2015.
[11] S. Fugivara, A. V. D. Merladet, and C. H. N. Lahoz, "STPA Analysis of Brazilian Sounding Rockets Launching Operations," Microgravity Science and Technology, vol. 33, article no. 43, 2021.
[12] S. Sultana, P. Okoh, S. Haugen, and J. E. Vinnem, "Hazard Analysis: Application of STPA to Ship-to-Ship Transfer of LNG," Journal of Loss Prevention in the Process Industries, vol. 60, pp. 241-252, 2019.
[13] S. Shin, S. Lee, S. Shin, I. Jang, and J. Park, "STPA-Based Hazard and Importance Analysis on NPP Safety I&C Systems Focusing on Human-System Interactions," Reliability Engineering & System Safety, vol. 213, article no. 107698, 2021.
[14] A. Yousefi and M. R. Hernandez, "Using a System Theory based Method (STAMP) for Hazard Analysis in Process Industry," Journal of Loss Prevention in the Process Industries, vol. 61, pp. 305-324, 2019.
[15] Y. Watanabe, Y. Anang, and M. Takahashi, "Quality Model and Quality Characteristics Evaluation Suitable for Software 2.0," International Journal of Engineering and Technology Innovation, vol. 14, no. 3, pp. 309-320, 2024.
[16] S. Charalampidou, A. Zeleskidis, and I. M. Dokas, "Hazard Analysis in the Era of AI: Assessing the Usefulness of ChatGPT4 in STPA Hazard Analysis," Safety Science, vol. 178, article no. 106608, 2024.
[17] Unified Modeling Language Specification Version 2.5.1, https://www.omg.org/about/, 2017.

[18] Introduction to STAMP/STPA – A New Safety Analysis Method based on Systems Thinking –, Tokyo: Information Technology Promotion Agency, 2016. (in Japanese)
[19] Introduction to STAMP/STPA – Practical Edition –, Tokyo: Information Technology Promotion Agency 2017. (in Japanese)
[20] M. Acharya and B. Robinson, "Practical Change Impact Analysis Based on Static Program Slicing for Industrial Software Systems," Proceedings of the 33rd International Conference on Software Engineering (ICSE11), Association of Computing Machinery, pp. 746-755, 2011.
[21] S. Khan and S. Madnick, "Protecting Chiller Systems from Cyberattack using Systems Thinking Approach," Network, vol. 2, no.4, pp. 606-627, 2022.