A creation method of comprehensive cases and specifications for hardware and software combined test to detect undesirable events of an industrial product using HAZOP

Masakazu Takahashi, Kouji Ueno, Yunarso Anang and Yoshimichi Watanabe

ADepartment of Computer Science and Engineering, University of Yamanashi, Kofu, Japan; bDepartment of Statistical Computing, Politeknik Statistika STIS, Jakarta, Indonesia

ABSTRACT
This paper proposes a creation method of comprehensive cases and specifications for hardware and software combined test (HSCT) to detect undesirable events of an industrial product controlled by software using hazard and operability studies (HAZOP). By testing with created HSCT specifications, the proposed method detects undesirable events of the industrial product. To remove undesirable events in industrial products, it is important to confirm the behaviour of as many undesirable events as possible in HSCT. However, since the HSCT cases were manually created by engineers, the coverage of the HSCT cases was insufficient. This paper proposes a comprehensive creation method of HSCT cases and specifications to resolve the problem of insufficient coverage of HSCT cases. The following countermeasures are implemented to realize the proposed method. (1) Define the method for extracting the parameters used in HAZOP from the hardware and software specifications, and define the information necessary to create HSCT cases and specifications. (2) Prepare HAZOP guide words for HSCT. (3) Define the format of parameters and guide words and the procedure for creating HSCT cases by combining parameters and guide words. (4) Propose a creation procedure of HSCT specifications corresponding to the HSCT cases. (5) Propose an adequate HSCT procedure that includes methods (1)–(4). As a result of the application of the proposed method, 25% more adequate HSCT cases are created and 40% more undesirable events were detected in comparison with the manual creation. Additionally, the creation time was reduced by 27%.

ARTICLE HISTORY
Received 28 October 2021
Revised 14 February 2022
Accepted 19 April 2022

KEYWORDS
HAZOP; hazard analysis; safety analysis; industrial products; hardware and software combined test; test case; test specification

1. Introduction
This paper proposes the creation method of comprehensive cases and specifications for hardware and software combined test (HSCT) to detect undesirable events of an industrial product controlled by software using hazard and operability study (HAZOP), which is a representative risk analysis method.

Recently, industrial products have become more sophisticated in function and performance, and their configurations have become complicated. As a result, undesirable events such as behaviour that is different from the design intention and abnormal behaviour that was not expected at the design phase may occur, and those behaviours may adversely affect the safety of industrial products.

To prevent the occurrence of undesirable events, tests must be performed to ensure the appropriateness of the hardware parts and software parts of the industrial product. In addition, HSCT that combines the tested hardware parts and tested software parts must be conducted to ensure that the industrial product behaves as intended and that the industrial product does not behave abnormal operations that were not expected at the design phase. Especially, it is necessary to comprehensively confirm the behaviours that cannot be tested when the industrial product works actually, such as the abnormal value input and the task execution at abnormal timing. However, HSCT cases were missed because the engineer had manually created the HSCT cases and specifications. This is called the problem of HSCT comprehensiveness. Therefore, it was difficult to detect undesirable events in industrial products and remove their causes.

In the proposed method, HAZOP is used to comprehensively create HSCT cases to resolve the problem of HSCT comprehensiveness. HAZOP combines the parameters of industrial products and guide words and comprehensively creates deviated states that differ from the normal state. Furthermore, HAZOP analyses what kind of negative impacts will occur on industrial products when the deviated states occur. Here, the parameters are the characteristic quantities...
such as temperature, pressure, humidity, etc. that represent the state of the industrial product, and the guide words are the qualifiers that represent the deviation of the characteristic quantities. The qualifiers of examples of the guide words are large, small, fast, slow, etc. In the proposed method, the individual deviated states created using HAZOP are considered to be undesirable events of industrial products, and those are considered as the cases of HSCT. The problem of HSCT comprehensiveness is resolved by the following method. All input and output data and all input and output control signals to the software that controls the behaviours of industrial products are considered as parameters. Then, the information necessary to create the HSCT case and the specifications are added to the parameter, and the parameter format is defined. The information includes the device name, parameter name, parameter type, and sender/receiver. Then, referring to the existing guide words, the existing HSCT cases, and the existing specifications, the guide words for HCST such as data is too large, too small, etc., and control signals are too early, too late, etc. are prepared, and the format of the guide words is defined. By automatically combining the formatted parameters and the formatted guide words, the deviated states are comprehensively created. Those deviated states are considered the HSCT cases. Then, referring to HSCT cases, hardware design specifications, software design specifications, and operation manuals, HSCT specifications are created. At last, undesirable events in the industrial product are detected and removed, by conducting HSCT, and a safer industrial product is realized.

This paper was created by reviewing the results presented at SICE2021 [1] and adding the following items. Along with that, the chapter structure was reviewed, and additions and corrections were made.

- Clarification of the concept and the description regarding test case creation using HAZOP,
- Definition of information extraction method required when creating HAZOP cases from design documents,
- Detailed evaluation of application results such as analysis of HSCT case and specification creation time,
- Clarification of application limits of the proposed method.

The rest of the structure of this paper is described. Section 2 describes the outline of HAZOP and related researches. Section 2.3 describes the proposed method. Section 4 describes the application and evaluation result of the proposed method. Then, Section 5 describes a summary and future issues.

2. Related works

Section 2.1 describes the test process of an industrial product controlled by the software. Section 2.2 describes the outline of HAZOP, and Section 2.3 describes related works.

2.1. Test process of industrial products

This section describes a typical test process for an industrial product whose behaviours are controlled by software [2–5]. Figure 1 shows the outline of the test process. The test process of the industrial product consists of the individual hardware test phase, individual software test phase and hardware and software combined test (HSCT) phase.

The individual hardware test phase includes the component level test (tests for minimum functional components), the unit level test (tests for hardware that combines multiple minimum functional components), the subsystem level test (tests for hardware that combines multiple units and with large functions), and the system-level test (tests for the whole industrial product). Those tests are carried out step by step, and it proves that the functions of the hardware are adequate. The number of layers of the hardware depends on the scale of the industrial product.

The individual software test phase includes unit test (tests of the functions of the smallest module unit), combined test (tests of the functions of the detailed design level that combines the smallest modules), functional test (tests of the functions of the preliminary design level), system test (tests of the requirements described in the requirement specifications), and operational test (tests of the actual operation of the industrial product). The individual software test is carried out step by step to prove that the software functions are adequate. The white-box test method (a method of determining test cases that cover all branches and all branch conditions from the source code) is used in the unit test stage. In the stage of combined test, functional test, and system test, the black-box test method (a method of determining test cases that investigate behaviours corresponding to all input groups (equivalence) from the contents described in the specifications) is used [6].

At the HSCT phase, the software that has been individually tested is installed into the hardware that has been individually tested. Then, by operating the industrial product according to the actual operation procedure, it confirms that the industrial product can exhibit appropriate functions and performance. At this time, in addition to the conduction of the normal operation procedure, the abnormal operation procedures that engineers can consider as possible are tested. In the case of testing related to the hardware failure, HSCT
is conducted by using the simulator that can simulate the failure because we cannot destroy the hardware (especially, individual production type products).

2.2. Outline of HAZOP

HAZOP is activities that causes the undesirable events in the system that are investigated by the specialists [7].

First, HAZOP is originally a method for analysing the accidents that occur when operating a chemical plant. In HAZOP, it considers that the plant is safe when the plant is operated according to the design intention but the plant becomes unsafe when the operation deviates from the design intention. Therefore, the engineers consider comprehensive deviations from the design intent and evaluate the negative impacts on the plant and various safety countermeasures when the deviation occurs. Deviations from the design intent are created by combining characteristic quantities (parameters) that represent the operating state of the plant and keywords (guide words) that represent the types of deviations prepared in advance. Here, deviations in HAZOP are equipment failures, operational errors, and abnormal measured values et al. The following advantages can be obtained by conducting HAZOP. The deviated states are created by combining guide words and parameters comprehensively, and the accidents related to the deviated state are analysed. As a result, it is possible to comprehensively analyse the accidents. Since HAZOP is performed manually, the deviated state is missed, or engineers sometimes miss the deviated states. However, since the effectiveness of HAZOP in risk analysis has been recognized, HAZOP has recently been applied to various fields other than chemical plants, such as electrical and electronic circuits, software development, medical care (surgery), and work diagnosis [8].

Next, the procedure of HAZOP is as follows. Firstly, the design concepts of the system are explained. The design concepts show the characteristics of the subsystems, such as input, function, activity, start point, goal, etc. Secondly, the target system is divided into several subsystems. Thirdly, deviations from the design concept are created by combining the parameters and guide words in the unit of the subsystem. Those are called deviated states. Table 1 shows the list of the guide words. Fourthly, the causes of the deviation and the results that occur the deviation are predicated. Finally, the countermeasures are considered in the case that the prospected results are not acceptable, that is the results are undesirable events. By conducting the above-mentioned tasks, the undesirable events of the system are removed.

2.3. Related researches

Related researches can be classified into researches for improvement of HAZOP implementation, researches of HAZOP application, and researches for the creation of test cases.

First, researches related to the improvement of HAZOP implementation are explained. Hulin et al. resolved the insufficiencies of the expression of the
Table 1. HAZOP guide words and meanings.

| Guide word     | Meaning                                      |
|----------------|----------------------------------------------|
| No or Not      | Complete negation of the design intent       |
| More           | Quantitative increase                        |
| Less           | Quantitative decrease                        |
| As well as     | Quantitative modification/increase           |
| Part of        | Quantitative modification/decrease           |
| Reverse        | Logical opposite of the design intent        |
| Other than     | Complete substitution                        |
| Early          | Relative to the clock time                   |
| Late           | Relative to the clock time                   |
| Before         | Relating to the order or sequence            |
| After          | Relating to the order or sequence            |

deviated states by defining the format of the parameters used in HAZOP and describing the deviated state using the format [9,10]. As a result, the readability and the sufficiency of the deviated states were improved. Kouno proposed a method to comprehensively clarify whether the deviated states occur by applying the guide word to the software requirement specifications [11]. Cui et al. proposed an integrated process hazard analysis environment that combines HAZOP, Layer of Protection Analysis (LOPA), Safety Requirements Specification (SRS), and Safety Integrity Level (SIL) validation for chemical plants [12].

Next, researches related to HAZOP application are explained. Hansen et al. conducted HAZOP on the software architecture of safety-critical systems described in Unified Modelling Language and clarified the inappropriate software part [13]. Hidaka et al. proposed a method for deriving software abnormal conditions by conducting HAZOP to improve the reliability and developed a back monitor system [14]. Kim et al. used HAZOP in the software approval process for highly safe software running on PLCs [15]. Jurkiewicz et al. proposed a determination method of whether there are insufficient use cases in the software requirement specifications described in a USE CASE diagram [16].

Finally, researches related to the creation of the test cases are explained. Hemmati et al. proposed a method of selecting high-priority test cases by focusing on similarities in test cases [17]. Catal et al. proposed a method to improve cost-effectiveness by prioritizing test cases to be executed based on the degree of impact of failures when performing regression testing [18]. Do et al. proposed a method for prioritizing test cases while considering time constraints, the number of external threats, the number of failures, etc. when performing regression tests [19]. Gay et al. reported that test cases that satisfy the higher structural coverage could detect serious failures when testing software [20].

As described above, various related studies have been carried out, but no research creates HSCT cases and specifications for detecting undesirable events of industrial products using HAZOP.

3. Proposed method

This section describes the proposed method. Section 3.1 explains the concept of the creation method for HSCT cases using HAZOP. Section 3.2 explains the HSCT procedure using the proposed method.

3.1. Concept of the proposed method

This subsection explains the concept of the method of creating HSCT cases and the necessary items to realize the proposed method.

3.1.1. Creation of comprehensive HSCT cases

The concept of comprehensive creation of HSCT cases using HAZOP in this paper is explained.

At first, HSCT cases are explained. When an undesirable event occurs in an industrial product, the measurable parameters of the industrial product become abnormal values. Abnormal values of parameters mean that the value of the parameter increases too much, or the value of the parameter decreases too low, for example. Since those states are considered to be the same as the deviated states of HAZOP, the proposed method considers those to be the HSCT.

Next, the comprehensive creation of HSCT cases is explained. It is necessary to clarify all parameters to create comprehensive HSCT cases. The proposed method defines a method for extracting all parameters from the hardware design specifications and the software design specifications (described later in 3.1.2), all guide words applicable to each parameter are prepared (described later in 3.1.3), and the format of parameters and guide words are defined (described later in 3.1.4). Then, by combining all parameters of the industrial product and all the guide words, comprehensive HSCT cases are created (described later in 3.1.5), and HSCT specifications based on HSCT cases are created (described later in 3.1.6). Finally, by conducting HSCT, undesirable events in industrial products are detected, and countermeasures are taken.

3.1.2. Parameters used in HAZOP

This subsection describes the information that the parameters should create HSCT cases and specifications and explains the parameter extraction method from the hardware and software design specifications.

At first, the information that the parameter should have is described. Since the parameters are measurable quantities of the target system, they are the data and control signals handled by the system. Data is input/output through input/output devices such as analog sensors, digital (on/off) sensors, switches (on/off), etc., and control signals are input/output through interrupt control devices and timers. Since the applicable guide words differ depending on the parameter data type, data type such as numeric, character, and boolean are used.
value is added to the information. Since the device status changes due to the change of the parameter, information on which device outputs the parameter or which device inputs the parameter is necessary to add. Furthermore, since the parameters are processed by the software part, the information of the corresponding software part is necessary to add. As a result, the parameters have information of one device name, one or more characteristic quantities (data name, data type, output destination (sender), or input source (receiver)), and one or more software parts.

Next, the method of extracting parameters from the specifications is explained. The design information of industrial products consists of the hardware design specifications and software design specifications. The parameters are extracted from the software specifications, and the control signals are extracted from the hardware specifications and software specifications. Software specifications are often written using the Data Flow Diagram (DFD) and Control Flow Diagram (CFD) of the structured analysis method [21] or Unified Modelling Language (UML) [22] of the object-oriented method. The parameter extraction method from each method is explained as follows.

Figure 2 describes an example of the parameter extraction method when the specifications are written in the structured method. Data is the data flow between a source or sink and a process in a Data Context Diagram (DCD). Output destination and input source of the data are determined by the direction of the data flow in DCD. And the data type is obtained from the data dictionary. The control signal is the control flow in the Control Context Diagram (CCD). The output destination and input source of the control signal are determined by the direction of the control flow in the CCD. Since the control signal type is only on/off, it is a digital type. In addition to these documents, the control signals input or output to the industrial products described in the electrical interface control document are also parameters.

Figure 3 describes the extraction method when the specifications are written in the object-oriented method. Data and control signals are messages between actors and lifelines in sequence diagrams. The output destination and input source of the data and control signal types are determined by the direction of the message. The data and control signal types are determined from the return value of the message. As in the case of
the structured method, the control signals described in the electrical interface management document are also parameters.

By conducting the above-mentioned tasks, all parameters of industrial products are extracted.

3.1.3. Guide words used in HAZOP

This section describes the guide words used to create the deviated states in HAZOP.

In the conventional HAZOP, the guide words and parameters are manually combined to create the deviated states. Therefore, there were problems such as omission in the combination of the guide word and the parameter, and the creation of a large number of inadequate deviated states in which the inadequate parameter and the guide word were combined. In the proposed method, as described in 3.1.2, the parameters are classified into numerical type, character type, string type, or boolean type. From this, the guide word groups that can be applied to the numeric type, character string type, and Boolean type are defined. As a result, it is possible to prevent the lack of deviated states and creation of the inadequate deviated states. Table 2 shows the guide word groups. Some reasons for selecting the guide words are shown as follows:

- Because the control software does not compare the size of characters (string), the guide words, such as greater and smaller, are excluded.
- Because the meaning of Not equal and Reverse in Boolean is the same, Not equal is excluded from the Boolean guide word group.

3.1.4. Formats of parameters and guide words

This subsection describes the formats of parameters and guide words. HAZOP creates the deviated states by combining the parameters and guide words. Accordingly, it is desirable to combine, describe, and read those data easily. Additionally, it is desirable to deal with those data formally to improve task efficiency. Therefore, the proposed method describes the parameters and guide words using YAML Ain’t Markup Language (YAML) and preserves those descriptions as text files [23]. The YAML format is suitable for describing the data structure and the readability is superior to other markup languages.

The proposed method describes the program portions as supplemental information of the parameters in addition to the parameters and guide word groups. Hereinafter, those are called the configuration files. Figure 4 shows the format of the configuration files. The varieties of the configuration files are specified using the kind term (identifier) in the configuration files.

In the case that the configuration file describes the parameters, the name of equipment is described in the name field and the name of the parameter, type, and from/to equipment are described in the parameters field. In the case that the configuration file describes the program, the equipment name is described in the name field and the name of the program is described in the parameters field. In the case that the configuration file describes the guide word, the guide word group is described in the name field and the name of the guide word is described in the parameter.

3.1.5. Creation of deviated states

This section shows the creation procedure of the deviated states by using the formatted parameters and formatted guide words. The creation procedure is as follows:

1. One “name of equipment” is selected from the parameter configuration file. This is regarded as “from equipment (the equipment sending parameter).”
2. One “name of parameter” that the value of input/output type is output is selected from the parameter configuration file. This parameter is regarded as “investigating parameter.” And “name of equipment” that is received parameter is obtained. This is regarded as “to equipment.”
3. The type of “investigating parameter” is obtained from the parameter configuration file. One guide word corresponding to the type of the “investigating parameter” is selected. This is regarded as “investigating guide word.”
4. The programs related to the “from equipment” and “to equipment” are selected from the program configuration file. Those programs portions are regarded as “programs.”
5. The deviated state is created by combining as follows; “from equipment” + “to equipment” + “investigating parameter” + “investigating guide word” + “programs.”
6. Tasks (1)–(5) are conducted to all output parameters and guide words. Those are the deviated states created, and those are regarded as the HSCT items.

In this research, the deviated states creation tool from the configuration files was developed. The tool was written in Ruby script. This tool creates the deviated states comprehensively by combining the parameters and guide words in the guide word group.
3.1.6. Creation of HSCT specifications

This subsection describes the development method of HSCT specifications. Figure 5 shows the development method of HSCT specification. To develop an HSCT specification, an HSCT case (a deviated state), hardware design specifications, software design specifications, and operational specifications are necessary. The developing procedure of HSCT specification is described as follows:

1. Target equipment cell of HSCT specification is filled in "from equipment" of the HSCT case.
2. Affected equipment cell of the HSCT specification is filled in "to equipment" of the HSCT case.
3. Parameter + guide word cell of the HSCT specification is filled in "investigating parameter" + "investigating guide word".
4. Prospected result is considered by referring to the hardware design specifications and the software design specifications when "parameter + guide word" occurs. As a result, HSCT specifications are developed.
5. Relate programs cell is filled in "programs."
6. Actual result cell and accept/reject cell of the HSCT are filled after the HSCT is conducted.

3.2. Proposed creation procedure of HSCT cases and specifications

This subsection explains the procedure of the proposed method. Figure 6 shows the creation procedure of HSCT cases and specifications. The procedure of the proposed method is described as follows:

1. The parameter file used in HAZOP is defined using the information of industrial product’s hardware design specifications and software design specifications. When using commercial hardware and software, the drawings and specifications created by their manufacturers are used. When using individually developed hardware and software, the hardware specifications and the software specifications created by the engineers are used.
2. Applicable guide word group shown in Section 3.1.3 is selected by referring to the type in the parameter file. According to the procedure described in Section 3.1.5, the deviated states are created by combining the parameter file and guide words file.
3. The deviated states that cause the undesirable events are selected by investigating the created deviated states. The selected deviated states are regarded as the HSCT cases.
4. The HSCT specifications are developed by referring to the HSCT cases, the hardware design specifications, and the software design specifications.
5. The HSCT is conducted using the HSCT specifications.
6. The hardware design specifications and the software design specifications are modified based on the results of the HSCT. By repeating from tasks (1) to (6), the adequate industrial product is realized. Here, tasks (5) and (6) are out of the proposed method.

4. Application and evaluation

To evaluate the usefulness of the proposed method, we developed a simple robot system and applied the proposed method to the robot. Then, we conducted an HSCT, detected the undesirable events, and removed the causes of undesirable events. Section 4.1 describes the system configuration of the target system. Section 4.2 describes the application results when applying the proposed method. Section 4.3 describes the comparison of the test coverage and working time between the proposed method and the manual method. And Section 4.4 discusses the evaluation of the proposed method.

4.1. Outline of application system

We developed a simple robot that identifies the red-coloured object and accesses it. This robot has a part
of the functions of the robot that harvests tomatoes (hereinafter robot) [24]. Figure 7 shows the appearance of the robot. The robot consists of the Raspberry Pi 3 Model B (hereinafter computer), the ARM Microcomputer (herein after microcomputer), motors for moving (herein after motor), web camera (Logiool C992n, herein after web camera), and mobile battery (Monalisa JF-PEACE9BK). As the battery is just connecting to the robot, the battery is out of the scope of the analysis. Except for the robot, this system has a client PC that sends the instructions to the robot. The robot and the client PC are connected using wireless LAN. The reason why the robot is selected as an application is that the hardware of the robot is easy to purchase and the source code is disclosed. The operational procedure of the robot is as follows:

1. The robot program on the computer is executed by the client PC.
2. The robot receives the image from the Web camera and identifies the red object. OpenCV [25] is used to identify the red object.
3. The robot controls the motor by selecting one of the straight-ahead, right-turn, and left-turn behaviour to access the red object.
4. The robot accesses the red object by repeating the behaviours of (2) and (3) and stops when the robot accesses the red object close enough.
Table 3 shows the outline of the programs. The OS and libraries were excluded because the source code was not accessible.

The parameter, program part, and guide word group configuration files were created according to the procedure of the proposed method. In this application experiment, the hardware and software design information necessary to create the configuration files was created by the authors based on the information published by the robot manufacturer. The reason why the authors created the design information is that the software design information created by the robot manufacturers was insufficient. Figure 8 shows a part of the software design information (DFD) created by the authors. Figure 9 shows the parameter configuration file, Figure 10 shows the program part configuration file, and Figure 11 shows the guide word group configuration file.

4.2. Application results of the proposed method – detected hazards applying the proposed method

As the input of the configuration files explained in Section 4.1, 74 deviated states were created automatically using the developed tool. Figure 12 shows the sample deviated states (extracted). “-Expression” shows the deviated state, and “-expression” contains information, such as “from equipment, to equipment, parameter + guide word + target programs.” For example, “from Camera to Computer runningState No” means that the guide word “no” is applied to the runningState that is outputted from camera to computer. “Target” after the expression shows the target program portions. The analysis process is explained using an example of “from camera to computer runningState No” in Figure 12. This deviated state can be interpreted as “there is no active signal from the camera to the computer.” The reasons why the signal is lost are considered as follows: the physical snapping of the wire or failure of the hardware, and the bugs in the programs, etc. The former can be dealt with by the replacement of the hardware periodically. The cause of the latter cases was that the value of the buffer was fixed because the output from the camera to the computer was ceased when the termination procedure did not finish adequately. As a result, the picture from the camera cannot be obtained even if the camera is working. This is an undesirable event. As a result of analysing other results of the HSCT specifications, 20 causes of the undesirable event could be detected. A part of undesirable events are as follows:

- The processing of the picture is not conducted even if the picture input from the camera is obtained.
- The error picture (black screen) is obtained because the time gap between the activation of the camera and the request of the picture is too short.
- The motor continues to act because of the abnormal program termination.
- Different control signals are output to the motor by multiple processes.
- The program is forcibly stopped by shutting down the microcomputer.
Figure 8. Software preliminary design and detailed specification created by the authors – extracted.

| kind: | parameter |
|-------|-----------|
| - name: | Computer |
| - name: | Microcomputer |
| parameters: | parameters: |
| - runningState: | runningState: |
|   boolean: |   boolean: |
| to ClientPC | to ClientPC |
| - powerState: | powerState: |
|   numeric: |   numeric: |
| to ClientPC | Computer |
| - networkState: | computerInput: |
|   boolean: |   numeric: |
| from Camera | from |
| - cameralInput: | computer |
|   string: | |
| from Camera | |
| - microcomputerOutput: | motorOutput: |
|   string: |   numeric: |
| to microcomputer | to Motor |
| - instruction: | name: |
|   string: | Motor |
| from ClientPC | parameters: |
| - name: | Camera |
| parameters: | parameters: |
| - runningState: | name: |
|   Boolean: | ClientPC |
| to Microcomputer | parameters: |
| - computerOutput: | - command: |
|   String: |   string: |
| from Microcomputer | to computer |

Figure 9. Configuration file of parameters.

| kind: | program |
|-------|---------|
| - name: | Computer |
| - name: | Microcomputer |
| parameters: | parameters: |
| - robotInit.sh | robot.c |
| - robotStart.sh | main.c |
| - robotEnd.sh | name: |
| - name: | Motor |
| parameters: | parameters: |
| - main.c | - main.c |
| - name: | ClientPC |
| parameters: | parameters: |
| - client.c | - client.c |

Figure 10. Configuration file of program parts.
4.3. Comparison with the existing method and proposed method – breakdown of numbers of HSCT cases and working time of each task

We asked two engineers who have a fundamental knowledge of safety analysis and 2–3 years of experience in safety analysis to create the HSCT cases and specifications. One created HSCT cases and specifications using the proposed method (described in 4.2) and the other created the HSCT cases and specifications based on his skills and experience (hereinafter manual operation). The HSCT was conducted using the cases and specifications created by each engineer to evaluate whether undesirable events could be detected.

First, the comparison results regarding the coverage of the created HSCT cases and the number of detected undesirable events are described. Table 4 shows the comparison results. The number of the HSCT cases created by the proposed method was 74, the number of overlapped cases was 0, the number of missed cases was 0, and the actual number of the HSCT cases and specifications was 74. Here, missed case refers to the case where the parameters and guide words that must be combined are not combined when the HSCT case is created. The cause may be the engineer’s miss judgment or a simple omission. Overlapped case refers to the case where the meanings of test cases created by combining the parameter and different guide words become the same meaning. The cause is that the engineer forgot to delete the duplicated HSCT cases because the engineer had to create and check a large number of HSCT cases. This is because the parameters and guide words that can be combined were prepared in advance, so that inadequate combinations of parameters and guide words did not occur. And 20 undesirable events were detected by the HSCT cases and the specifications created by the proposed method. On the other hand, the number of the HSCT cases created by manual operation was 96, the number of overlapped cases was 22, the
number of missed cases was 15, and the actual number of the HSCT cases and the specifications was 59. And 14 undesirable events were detected by the HSCT cases and the specifications created by manual operation. Here, all actual HSCT cases created by the manual operation are included in the HSCT cases created by the proposed method. The reason for this is that, as described above, the proposed method can create the comprehensive and adequate combinations of parameters and guide words. From the above, it was found that the proposed method can create 25% more adequate HSCT test cases and specifications and can detect 42% more undesirable events.

Second, the working time of the HSCT cases and specifications are described. The work items correspond to tasks (1)–(4) in Figure 6. Table 5 shows the comparison results. The creation time of the HSCT cases is evaluated. In the case of the proposed method, the working time of task (1) is 2 h. The working time of task (2) was 0 h, the working time of task (3) was 0 h, and the total working time was 2 h. Task (1) is the task of extracting parameters by referring to the design information of the hardware and software and is the important task of creating HSCT cases in the proposed method. Tasks (2) and (3) can be conducted automatically using the result of task (1), so it does not take much time. On the other hand, in the case of manual operation, the working time of task (1) was 0 h, the working time of task (2) was 2 h, the working time of task (3) was 4 h, and the total working time was 6 h. Creating configuration files is not included in the manual operation. As mentioned in the evaluation of coverage rate, the test cases were insufficient in the manual operation.

Finally, the working time necessary to create the HSCT specifications is evaluated. The proposed method takes 20 h, and the manual method takes 24 h. As a result of comparing the working time of the test specifications, it was found that the working time can be reduced by about 27%. This was because the proposed method created the minimum HSCT specifications, but manual operation created inappropriate HSCT specifications (overlapped and missed cases). The working time necessary to create one HSCT specification was the same in both cases.

4.4. Evaluation of the proposed method

We consider the application results of the proposed method.

The advantages of the proposed method will be described. Some of the detected undesirable events were likely not to be detected by the individual hardware test and individual software test. For example, for the undesirable event of "acquiring an incorrect image because the time from the camera startup to the program's image request was too short", both the Web camera and the software were operating adequately to meet their respective requirements. No problem could be detected by individual hardware tests and software tests. However, in the proposed method, the problem was found that the system did not operate normally because the image was displayed in black for several frames between the time when the main program called the camera and the time when the camera started up. In this way, the proposed method was able to comprehensively create the HSCT cases and the specifications for undesirable events, so those complex undesirable events could be detected. In the proposed method, the information necessary to create the deviated state was expressed in a file format with high readability and operability, which enabled formal handling and automatically created the HSCT cases. Furthermore, by defining a guide word group for each parameter type, the creation of an inadequate combination of deviated states was prevented. Those improved the efficiency of the HSCT case creation and the rate of adequate HSCT cases.

The disadvantages of the proposed method will be described. In the proposed method, the parameters were measurable items. Therefore, it was not possible to examine undesirable events that were caused by the not-measured characteristic quantities in the industrial
products. For example, although it deviates from the original usage of HAZOP, the HSCT cases that are created by the experienced engineer using the information other than parameters (unmeasured physical quantities, internal software variables, information about the environment in which the plant is located, a combination of multiple information, etc.) cannot be created by the proposed method. As a countermeasure, we will consider a method of creating the HSCT cases based on the specified internal data, the multiple combinations of parameters, and the experiences of the engineer. Furthermore, in the proposed method, the HSCT specifications were created manually, so the creation efficiency could not be significantly improved. In the future, the automatic creation method of the HSCT specifications will be considered to improve creation efficiency.

5. Conclusion

In this paper, we proposed a method for comprehensively and efficiently creating the HSCT cases using HAZOP, and we proposed a method for creating the HSCT specifications for detecting undesirable events in industrial products based on the created HSCT cases. The proposed method avoided to create overlapped HSCT cases and missed cases. As a result, the proposed method created 25% more adequate HSCT cases and detected 42% more undesirable events. Furthermore, the working time required to create the HSCT specifications could be reduced by 27%. The reason why the working time of HSCT specifications could be reduced is that the inappropriate HSCT cases were not created by applying the proposed method. There was no change in the working time necessary to create one test specification.

From the above, it was found that the proposed method could comprehensively and efficiently create the HSCT cases, the created HSCT specifications could detect the undesirable events by using the appropriate HSCT specifications and were helpful to propose adequate countermeasures against undesirable events. The proposed method became to contribute to improving the quality of industrial products.

Future issues will be described. First, we will realize a method to automatically create the HSCT specifications from the HSCT cases. Second, we will realize a mechanism to specify parameters that cannot be measured directly and create the deviated states. Third, we will consider and add guide words that will detect more undesirable events to the guide word groups. Fourth, by improving and expanding the proposed method, we will create test cases after HSCT, such as operational tests and factory acceptance tests. Finally, we will apply the proposed method to various types of industrial products and feedback on the findings of the applications to the proposed method and support tools.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This research was supported by the Grant-in-Aid for Scientific Research (C) of the Japan Society for the Promotion of Science: [Grant Number 19K04920], title, “Integrated analysis method for hazard caused by software interaction cooperating with multiple safety analysis methods.” and “Building a Quality Model for Embedded Software including Machine Learning Calculation (21K04560).”

Notes on the contributors

Masakazu Takahashi received B.S. degree in 1988 from Rikkyo University, Japan, and M.S. degree in 1998, Ph.D. degree in 2001, both in Systems Management from University of Tsukuba, Japan. He was with Ishikawajima-Harima Heavy Industries Co., Ltd. from 1988 to 2004. He was with Shimane University, Japan from 2005 to 2008. He is presently a professor in University of Yamanashi, Japan. His research interests include software engineering and safety.

Kouji Ueno received B.E. degree in 2021 from University of Yamanashi. He is with Fujitsu cloud technologies limited from 2021.

Ynarso Anag received B.E. degree in 1995, M.E. degree in 1997, Ph. D. degree in 2017 both in software engineering from University of Yamanashi, Japan. He was with SYNC Information System, Inc., Japan from 2000 to 2007, as a senior system engineer. From 2008, he is with Institute of Statistics (Sekolah Tinggi Ilmu Statistik), Jakarta, Indonesia, as an assistant professor in computer science. His research interests include software engineering.

Yoshimichi Watanabe received the B.E. and M.E. degrees in computer science from University of Yamanashi, Japan in 1986 and 1988 respectively and received D.E. degree in computer science from Tokyo Institute of Technology, Japan in 1995. He is presently a professor of the Department of Computer Science and Engineering at University of Yamanashi. His research interests include software development environment and software quality.

References

[1] Takahashi M, Ueno K, Anag Y, et al. A comprehensive creation method of hardware and software combined test specifications for industrial product controlled by software using HAZOP. Proc SICE2021. 2021: 443–448.

[2] International Electrical Commission: IEC61508. Functional safety of electrical/electronic/programmable electronic safety-related systems. Switzerland: International Electrical Commission; 2010.

[3] International Organization for Standardization: ISO2 6262. Road vehicles – functional safety. International Organization for Standardization; 2011.

[4] GAMP forum. GAMP5 risk-based approach to complaint GxP computerized systems. USA: International Society of Pharmaceutical Engineers; 2008.

[5] International Electrical Commission: IEC62304. Medical device software – software lifecycle process. Switzerland: International Electrical Commission; 2006.

[6] Myers G, Sandler C, Badgett T. The art of software testing. 3rd ed. Hoboken, NJ: John Wiley & Sons; 2011.
[7] International Electrical Commission: IEC61882. Hazard and operability studies (HAZOP studies) – application guide. Switzerland: International Electrical Commission; 2016.

[8] International Electrical Commission: IEC 61882:2016. Hazard and Operability Study (HAZOP studies) – application guide. Switzerland: International Electrical Commission; 2016.

[9] Hulin B, Tschachtli R. Identifying software hazards with modified CHAZOP. Proc First Int Conf Performance, Safety Robustness Complex Syst Appl. 2011: 7–12.

[10] Hulin B, Tschachtli R. Generating unambiguous and more complete HAZOP expressions. Reliability, Risk and Safety. 2010: 7.

[11] Kouno T. An application of HAZOP to risk analysis of software requirement specification. Proc Japan Symposium on Software Testing 2012. 2012: 6.

[12] Cui L, Shu Y, Wang Z, et al. Hasilt: an intelligent software platform for HAZOP, LOPA, SRS and SIL verification. Reliab Eng Syst Saf. 2012;108:56–64.

[13] Hansen K, Wells L, Maier T. Proceedings of 2nd Nordic workshop on the unified modeling language (NWUML). HAZOP analysis of UML-based software architecture descriptions of safety-critical systems. PP.59-78. 2004.

[14] Hidaka T, Yamazaki F, Nakamoto Y, et al. A method of deriving anomaly detection rule by HAZOP analysis. IPSJ SIG Technical Report. 2009;2009-EMB-14 (1):8.

[15] Kim J, Kim S. Proceedings of International Conference on future generation communication and networking FGCN 2012). Software qualification approach for safety-critical software of the embedded system. (2012). p.175–180.

[16] Jurkiewicz J, Nawrocki J, Ochodek M, et al. HAZOP-based identification of events in use cases: an empirical study. Empir Softw Eng. 2015;20:82–109.

[17] Hemmati H, Arcuri A, Briand L. A scalable model-based testing through test case diversity. ACM Trans Softw Eng Methodol. 2013;22(1):1–42. Article No. 6.

[18] Catal C, Mishra D. Test case prioritization: a systematic mapping study. Softw Qual J. 2012;21:445–478.

[19] Do H, Mirarab S, Tahvildari L, et al. The effects of time constraints on test case prioritization: a series of controlled experiments. IEEE Trans Software Eng. 2010;36(6):593–617.

[20] Gay G, Staats M, Whalen M, et al. The risks of coverage-directed test case generation. IEEE Trans Software Eng. 2015;41(8):803–819.

[21] SESSAME WG2. Structured modeling for embedded software development, Shoeisha, 2006. (in Japanese).

[22] SESSAME WG2. Object oriented modeling for embedded software development, Shoeisha, 2006. (in Japanese).

[23] YAML.org. YAML Ain’t Markup Language (YAML™) Version 1.2(online). [cited 2021 Apr 23]. https://yaml.org/spec/1.2/spec.html.

[24] Japanese Raspberry Pi Users Group. Raspberry Pi [application] introduction, p.163–200. Tokyo: Gijutsu-Hyohon Co., Ltd.; 2013; (in Japanese).

[25] Itseez O. [cited 2021 Apr 23] https://opencv.org.