**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

*IJCSMC, Vol. 2, Issue. 12, December 2013, pg.26 – 39*

**RESEARCH ARTICLE**

# Model Checking of Safety Properties for Complex Systems Using MWF Inactivation and MNWF Activation

**Atsushi Katoh[1], Shinichiro Haruyama[2], Naohiko Kohtake[3], Yoshiaki Ohkami[4]**

Graduate School of System Design and Management, Keio University
[1] katoh.atsushi@z7.keio.jp; [2] haruyama@sdm.keio.ac.jp; [3] kohtake@sdm.keio.ac.jp; [4] ohkami@sdm.keio.ac.jp

***Abstract*— One of the aims of model checking is to confirm a system's safety properties. Safety properties state that undesirable events never occur. This paper describes a method that derives comprehensive and rigorous safety properties for model checking. In general, an undesirable event in a system is abstract, and so safety properties corresponding to undesirable events cannot be directly applied to model checking. To be applicable to model checking, safety properties must be derived by interpreting undesirable events in a system using the system specifications. In the model checking of complex systems, some safety properties may be neglected or insufficiently specified because the functions and/or conditions in the systems are complex. The proposed method adopts the concepts of "Must Work Function (MWF) Inactivation" and "Must Not Work Function (MNWF) Activation" to solve these issues. Comprehensive and rigorous safety properties for model checking are derived according to these concepts. Undesirable events are embodied by rewriting knowledge according to the term rewriting system. The effectiveness of the proposed method is evaluated by applying it to a wireless rail crossing system. The results show that the derived safety properties offer a significantly improved degree of comprehensiveness and rigor.***

***Keywords*— Model Checking; Safety Property; Must Work Function (MWF); Must Not Work Function (MNWF); Term Rewriting System**

## I. INTRODUCTION

Improving the reliability of systems in which computers are embedded is important to the safety and security of our societies. Recently, such systems have become ubiquitous, and are increasingly complex because more advanced functions are required. This complexity leads to frequent failures [1].

Model checking [2] is an attractive method that improves the reliability of complex systems. Model checking verifies whether a system's state transitions satisfy necessary properties. These properties are derived from upper specifications, past accidents/failures, and so on, related to the system being analyzed. The properties can be classified into safety properties and 'liveness' properties [3]. In this paper, we focus on deriving safety properties for model checking. Safety properties state that undesirable events never occur. The undesirable events based on safety properties are represented abstractly. In the case of wireless rail crossing systems [4], [5], one example of an undesirable event is "Collision", which means that "The train collides with road users such as cars and pedestrians". In model checking, specific safety properties applicable to model checking must be derived from undesirable events.

 

In the model checking of complex systems, deriving specific safety properties has the following two problems:

- Some safety properties may be neglected.
- Some safety properties may be insufficiently specified.

These problems are illustrated in detail in Fig. 1. In general, the functions and/or conditions to be considered are easy to specify when model checking is applied to a simple system. The undesirable event can be embodied appropriately, and rigorous safety properties are derived. However, the functions and/or conditions become more complex when model checking is applied to a complex system. Therefore, model checking practitioners have difficulty embodying undesirable events in complex systems. This causes the problems listed above.

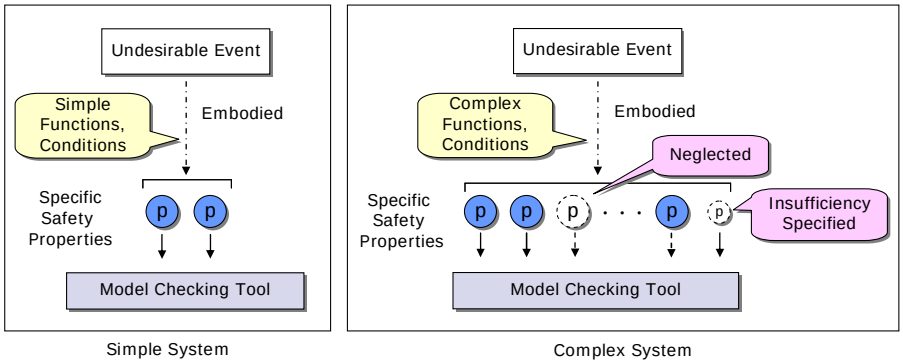


Fig. 1 Problem of deriving comprehensive and rigorous safety properties

This paper describes a method for deriving comprehensive and rigorous safety properties that are applicable to model checking by embodying an undesirable event with the concepts of "Must Work Function (MWF) Inactivation" and "Must Not Work Function (MNWF) Activation". An MWF is a function whose inadvertent shutdown would cause a hazard [6]. Similarly, an MNWF is a function whose inadvertent operation would cause a hazard [6]. In the proposed method, an undesirable event in a system is embodied by these ideas of "MWF Inactivation" and "MNWF Activation", which are expected to improve the comprehensiveness of the derived safety properties. The process of embodiment is conducted repeatedly. As a result, more comprehensive and rigorous events are derived. After deriving the embodied events from an undesirable event, safety properties are developed by negating ($\neg$) the embodied events. Embodying an undesirable event by MWFs and MNWFs is formalized using the term rewriting system [7]. This system is adopted because it allows the way of embodying undesirable events to be defined rigorously and clearly. Term rewriting system also facilitates intuitive understanding of the proposed method by illustrating the knowledge rewriting rules (see Fig. 7 and Fig. 8). MWFs and MNWFs corresponding to an undesirable event are identified from system specifications analyzed by model checking, and are stored in databases (DBs). The undesirable event is rewritten using the knowledge of MWFs and MNWFs stored in the DBs. In this paper, it is assumed that all functions and conditions of a system required for the proposed method are described in the system specifications. Additionally, variables used in the state transitions for model checking are made consistent with those used in the safety properties derived by the proposed method.

The effectiveness of the proposed method is evaluated by applying the method to a wireless rail crossing system. The results show that the derived safety properties are 47% more comprehensive and 58% more specific than the previous safety properties.

The remainder of this paper is organized as follows. Section II discusses some related work, before the proposed method is described in detail in Section III. An evaluation of the proposed method is presented in Section IV, and its effectiveness is then discussed in Section V. Finally, some conclusions and ideas for future work are given in Section VI.

## II. RELATED WORK

Fault Tree Analysis (FTA) [8], [9] was used to derive safety properties in model checking [10], [11]. FTA of an undesirable event in a system can be used to produce relevant safety properties by negating the basic events in the fault tree. Such methods enable safety properties to be systematically derived in model checking. However, the rules needed to identify lower events from the corresponding upper event in a fault tree are not defined in [10] and [11].

Considering the embodiment of undesirable events, [12] proposed a method using a Hazard Analysis Model based on Goal Oriented Requirement Analysis. The Hazard Analysis Model is composed of a Hazard Context

 

layer as level 1, Hazard State layer as level 2, Object Attribute layer as level 3, and Object Method layer as level 4. An undesirable event is embodied through levels 1 to 4. This work enables undesirable events to be embodied systematically. However, the rules needed to identify events in a lower layer from the corresponding event in an upper layer were again not given.

Reference [13] proposed the safety analysis of distributed systems using the concepts of MWFs and MNWFs. MWFs and MNWFs corresponding to undesirable events are identified, enabling the effective safety analysis of distributed systems. However, the specific means of identifying MWFs and MNWFs is not described in [13].

## III. PROPOSED METHOD

In model checking, specific, rigorous safety properties are derived by embodying undesirable events in the system being analysed. The proposed method is shown in Fig. 2.

Once an undesirable event is provided to the proposed method, it is embodied by the two concepts of "MWF Inactivation" and "MNWF Activation". In embodying the undesirable event, multiple (or zero) "MWF Inactivation" events may be derived. The same is true for "MNWF Activation" events. Embodying an undesirable event in terms of "MWF Inactivation" and "MNWF Activation" is formalized by rewriting the system knowledge based on the idea of the term rewriting system. In the proposed method, an undesirable event is treated as knowledge. The events derived by rewriting this knowledge are also treated as knowledge, and events are embodied repeatedly by rewriting knowledge. MWFs and MNWFs corresponding to undesirable events are also treated as knowledge. They are identified in the system specifications, and are stored in MWF and MNWF DBs. The knowledge required for this rewriting process is then acquired from the MWF DB and MNWF DB. An undesirable event is said to be fully embodied when the rewriting rules cannot be applied to this knowledge. In other words, embodiment is complete when further "MWF Inactivation" or "MNWF Activation" events cannot be derived. After completing the embodiment, the edge events (those within the dotted boxes in Fig. 2) are negated (¬). The temporal logic operators "A" (All) and "G" (Globally) are added to the negated events if the safety properties are represented in Computational Tree Logic (CTL) [14], and the operator "G" is added if the safety properties are represented in Linear Temporal Logic (LTL) [14]. As a result, specific and rigorous safety properties are derived.

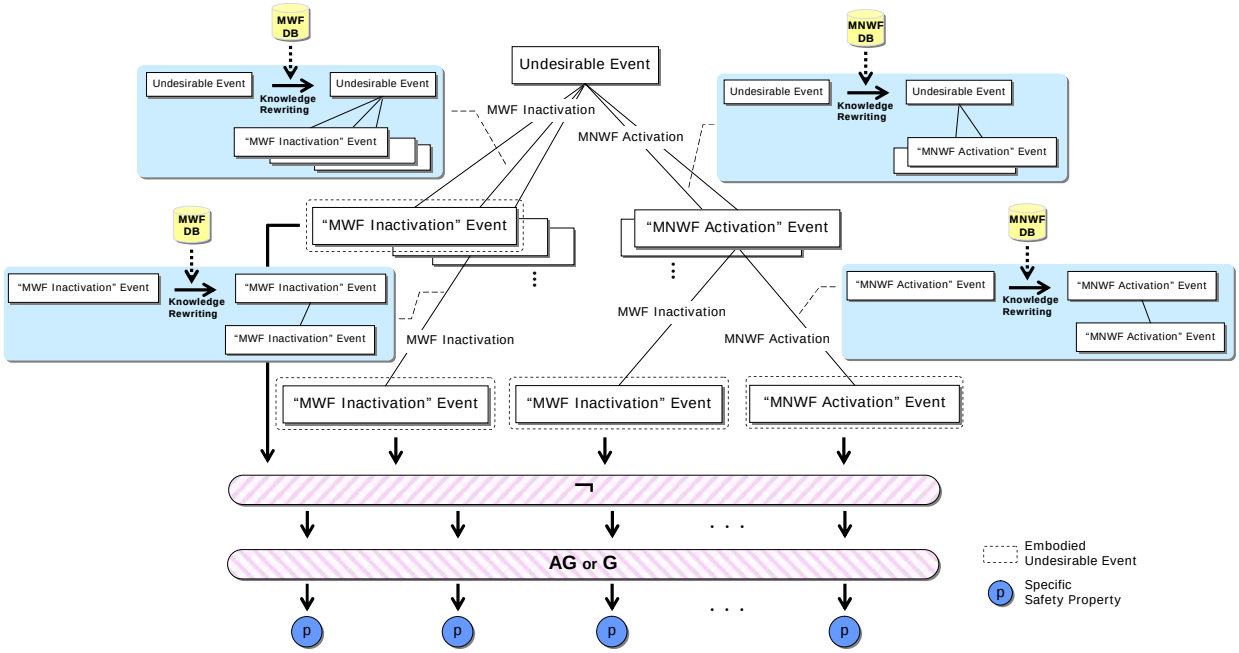


Fig. 2 Schematic of proposed method

### A. *Process of Proposed Method*

An undesirable event is provided as input to the proposed method. The safety properties applicable to model checking are then derived based on this undesirable event. The process of the proposed method is shown in Fig. 3. The following description corresponds to the blocks numbered 1–4 in Fig. 3.

1. MWFs and MNWFs corresponding to the undesirable event are identified from the specifications of the system being analysed. These are stored in the MWF DB and MNWF DB, respectively.

 

2. The undesirable event is embodied by knowledge rewriting based on the term rewriting system. Knowledge rewriting is conducted if an undesirable event to which the rules are applicable exists. The rules embody the undesirable event based on knowledge of MWFs and MNWFs stored in the DBs. As a result, "MWF Inactivation" and/or "MNWF Activation" events are derived. This embodiment is conducted repeatedly, as described in step 3.
3. If "MWF Inactivation" or "MNWF Activation" events are derived in step 2, the process returns to step 1. At that time, the "MWF Inactivation" events are replaced with undesirable events. Replacing means that the "MWF Inactivation" label is replaced by an undesirable event label. The contents of the "MWF Inactivation" event and the undesirable event are equivalent. "MNWF Activation" events are also replaced with undesirable events. The embodiment can be conducted repeatedly thanks to this replacing. Multiple "MWF Inactivation" and/or "MNWF Activation" events, that is, multiple undesirable events, may be derived depending on the results of step 2. In this case, the process is conducted for all undesirable events. Embodying the undesirable events is complete when no "MWF Inactivation" or "MNWF Activation" event is derived in step 2.
4. Embodied undesirable events are negated (¬). The temporal logic operators "A" and "G" are added to the negated events in the case of CTL, and "G" is added in the case of LTL. As a result, specific and rigorous safety properties are derived.

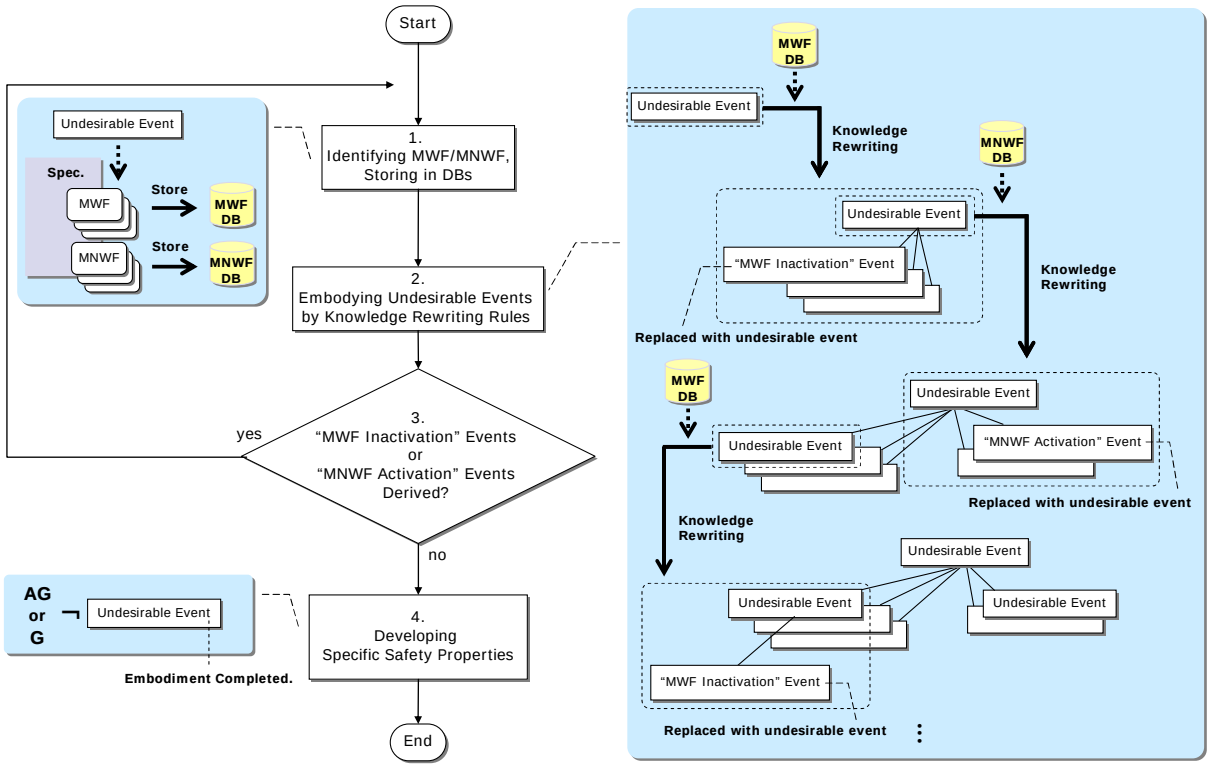


Fig. 3 Process of proposed method

### *B. Defining MWF, MNWF, MWF Inactivation and MNWF Activation*

Let F be the set of functions on the system being analysed by model checking, and let S be the set of conditions on the system. U is considered as the direct product of F and S (F×S). We can define the following predicates on U:

1) Active(f, s): System function f ∈ F is "active" under system condition s ∈ S. "Active" means that f is activated.
2) MW_Ctx(f, s): An undesirable event may happen if system function f is not "active" under system condition s.
3) MNWF_Ctx(f, s): An undesirable event may happen if system function f is not "inactive" under system condition s. "Inactive" means that f is not active.

 

First, MWFs and MNWFs are defined using these predicates as follows: "A certain system function $f \in F$ is MWF" means that there is at least one system condition $s \in S$ that makes MW_Ctx(f, s) true. In this case, the definition is formally represented by the logic function MWF(f) from domain F to range {true, false} as follows:

$$MWF(f): f \in F \rightarrow \text{truth value of } \exists s.\ MW\_Ctx(f, s) \quad (1)$$

"A certain system function $f \in F$ is MNWF" means that there is at least one system condition $s \in S$ that makes MNW_Ctx(f, s) true. In this case, the definition is formally represented by the logic function MNWF(f) from domain F to range {true, false} as follows:

$$MNWF(f): f \in F \rightarrow \text{truth value of } \exists s.\ MNW\_Ctx(f, s) \quad (2)$$

Next, "MWF Inactivation" and "MNWF Activation" are defined. "MWF Inactivation" is considered to be the circumstances whereby an undesirable event may happen if system function f is not "active" under system condition s, and f is not currently "active". Then, "MWF Inactivation" is formally represented as follows:

$$MWF\_Inactivation(f, s) \equiv MW\_Ctx(f, s) \wedge \neg Active(f, s) \quad (3)$$

In the case of wireless rail crossing systems [4], [5], "MWF Inactivation" is "(TrPos == appr_cr)∧¬(BaSts == closed)", where "(TrPos == appr_cr)" means "The train approaches the crossing" and "(BaSts == closed)" means "The barrier is closed".

"MNWF Activation" is considered to be the circumstances whereby an undesirable event may happen if system function f is not "inactive" under system condition s, and f is currently "active". Then, "MNWF Activation" is formally represented as follows:

$$MNWF\_Activation(f, s) \equiv MNW\_Ctx(f, s) \wedge Active(f, s) \quad (4)$$

In the case of wireless rail crossing systems, "MNWF Activation" is "¬(TrPos == pass_cr)∧(BaSts == open)", where "(TrPos == pass_cr)" means "The train passes the crossing" and "(BaSts == open)" means "The barrier opens".

### *C. Identifying MWF/MNWF and Storing into DB*

Fig. 4 illustrates the process of identifying MWFs and MNWFs corresponding to undesirable events, and storing them in the relevant DB. One undesirable event in a system is provided to the proposed method when the method is applied to the system. In the process of embodying undesirable events, events derived in terms of "MWF Inactivation" or "MNWF Activation" are also treated as undesirable events (cf. Section III.A). Those undesirable events are the input of the process in Fig. 4. The following description corresponds to the blocks numbered 1–4 in Fig. 4, where the parameter n denotes the total number of functions defined in the system specifications. Each step from 1 to 4 is processed by each system function $f_i \in F$ defined in the system specifications.

1. It is judged whether system function $f_i$ is an MWF of the undesirable event. Specifically, the method confirms whether system condition $s \in S$ exists such that the undesirable event may occur if system function $f_i$ is not active. This confirmation of MWF is conducted using the system specifications. If such a system condition s exists, step 2 is processed. The number of system conditions s for system function $f_i$ can be greater than one. If system condition s does not exist, step 2 is skipped.
2. The MWF and system condition in "MWF Inactivation" are stored together in the MWF DB. Specifically, system function $f_i$ and system condition s identified in step 1 are stored in the MWF DB as an MWF and the system condition under which $f_i$ becomes an MWF. In the case of the wireless rail crossing system [4], [5] in Fig. 4, "(BaSts == closed)" is identified as the MWF of the undesirable event "Collision", and "(TrPos == appr_cr)" is identified as the system condition under which this function becomes an MWF. The pair "(BaSts == closed)" and "(TrPos == appr_cr)" are linked to the undesirable event, and the linked pair and the event are stored in the MWF DB. More than one system condition s corresponding to $f_i$ can exist. In such cases, each pair of system condition s and system function $f_i$ are stored in the MWF DB.
3. It is judged whether system function $f_i$ is an MNWF of the undesirable event. Specifically, the proposed method confirms whether system condition $s \in S$ exists such that the undesirable event may happen if system function $f_i$ is not inactive. The confirmation of MNWF is conducted using the system specifications. If such a system condition s exists, step 4 is processed. The number of system conditions s for system function $f_i$ can be greater than one. If system condition s does not exist, step 4 is skipped.
4. The MNWF and system condition in "MNWF Activation" are stored together in the MNWF DB. Specifically, system function $f_i$ and system condition s identified in step 3 are stored in the MNWF DB as an MNWF and the system condition under which $f_i$ becomes an MNWF. In Fig. 4, "(BaSts == open)" is identified as the MNWF of the undesirable event "Collision", and "¬(TrPos == pass_cr)" is identified as the system condition under which this function becomes an MNWF. The pair "(BaSts == open)" and "¬(TrPos == pass_cr)" are linked to the undesirable event, and the linked pair and the

 

event are stored in the MNWF DB. More than one system condition s corresponding to $f_i$ can exist. In such cases, each pair of system condition s and system function $f_i$ are stored in the MNWF DB.

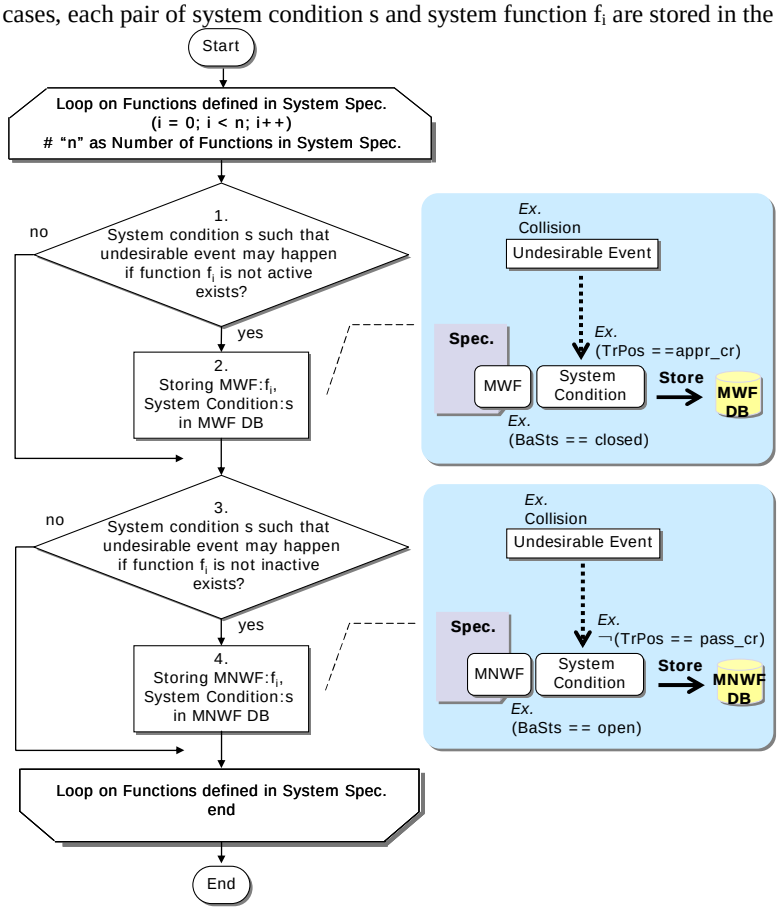


Fig. 4 Process of identifying MWF/MNWF and storing in the relevant DB

The structure of the MWF DB is shown in Fig. 5. This corresponds to the wireless rail crossing system example in Fig. 4. The MWF DB is composed of items such as "Undesirable Event", "MWF", "System Condition", and "MWF Inactivation Event". Knowledge of an "MWF Inactivation" event is developed when the DB function getMWFdeact() is executed, as described in Section III.D. Then, knowledge is formulated from information on the "MWF" and "System Condition" items according to (3) in Section III.B. The developed knowledge about "MWF Inactivation" event is stored in the MWF DB. An MWF and system condition pair is stored iteratively together with the corresponding undesirable events that replaced "MWF Inactivation" or "MNWF Activation" events. Therefore, multiple records of undesirable events are stored in the MWF DB, as shown in Fig. 5. The arrow in Fig. 5 shows that the "MWF Inactivation" event : "(TrPos == appr_cr)∧¬(BaSts == closed)" is replaced by an undesirable event, and the corresponding MWF (TrSndMsg == act_order) and system condition (TrDetPos == appr_cr) related to the undesirable event are stored in the MWF DB iteratively.

The MNWF DB is shown in Fig. 6. The concept of this DB is the same as that of the MWF DB.

| Undesirable Event | MWF | System Condition | MWF Inactivation Event |
|---|---|---|---|
| Collision | (BaSts == closed) | (TrPos == appr_cr) | (TrPos == appr_cr)∧ ¬(BaSts == closed) |
| | ··· | ··· | ··· |
| | ··· | ··· | ··· |
| (TrPos == appr_cr)∧ ¬(BaSts == closed) | (TrSndMsg == act_order) | (TrDetPos == appr_cr) | (TrDetPos == appr_cr)∧ ¬(TrSndMsg == act_order) |
| ··· | ··· | ··· | ··· |

Developed when executing **getMWFdeact()**

Fig. 5 Structure of MWF DB

 

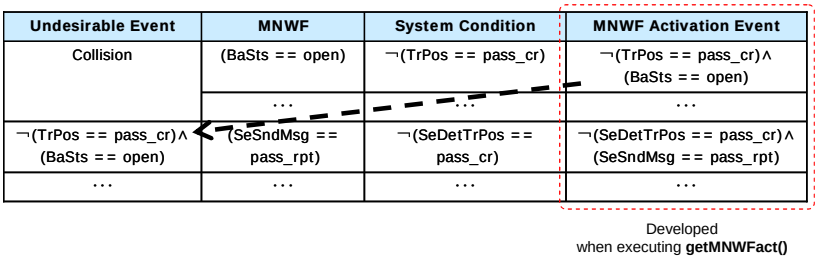


Fig. 6 Structure of MNWF DB

*D. Embodying Undesirable Events by Knowledge Rewriting Rules*

The embodiment of undesirable events is formalized using the concept of knowledge rewriting, which is based on the term rewriting system. The knowledge rewriting rules in the proposed method are as follows:

Rule 1: Embodiment in terms of "MWF Inactivation"

Rule 2: Embodiment in terms of "MNWF Activation"

If there is knowledge of an "Undesirable Event" where a rewriting rule is applicable, the knowledge is rewritten by one of the rewriting rules. Rule 1, shown in Fig. 7, depicts the knowledge rewriting rule based on the term rewriting system, where boxes denote knowledge. The function existUndEvtMWFDB() in the "Precondition on Applying Rules" in Fig. 7 judges whether the given argument, knowledge of "Undesirable Event", is stored under "Undesirable Event" in the MWF DB. "MWFdeactFlag" is a flag applied to the "Undesirable Event" when rewriting by Rule 1 is applied to the knowledge of this "Undesirable Event". Thus, "MWFdeactFlag" is used as control information to prevent Rule 1 being re-applied to the same "Undesirable Event". When rewriting according to Rule 1, the functions setMWFdeactFlg() and getMFdeact() are executed. The function setMWFdeactFlg() sets "MWFdeactFlag" for that "Undesirable Event". By executing setMWFdeact(), knowledge of "MWF Inactivation" events corresponding to the given argument ("Undesirable Event") is obtained from the MWF DB as a list. The knowledge of "MWF Inactivation" event is developed based on the knowledge of "MWF" and "System Condition" corresponding to that "Undesirable Event" in the MWF DB when setMWFdeact() is executed. Then, the knowledge of "MWF Inactivation" event is developed according to (3) in Section III.B. The developed knowledge "Event on MWF Inactivation" is stored in the MWF DB.

Rule 1 can be explained using the example of a wireless rail crossing system (Fig. 7). Knowledge about the "Collision" event is stored in the MWF DB, and "MWFdeactFlag" is not set for "Collision". In other words, there is an MWF for an undesirable event, and knowledge rewriting by Rule 1 has never been conducted. Then, the "Collision" event is rewritten to include "Collision", "(TrPos == appr_cr)∧¬(BaSts == closed)", and so on. The example of Fig. 7 corresponds to the wireless crossing system in Fig. 4 and Fig. 5.

 

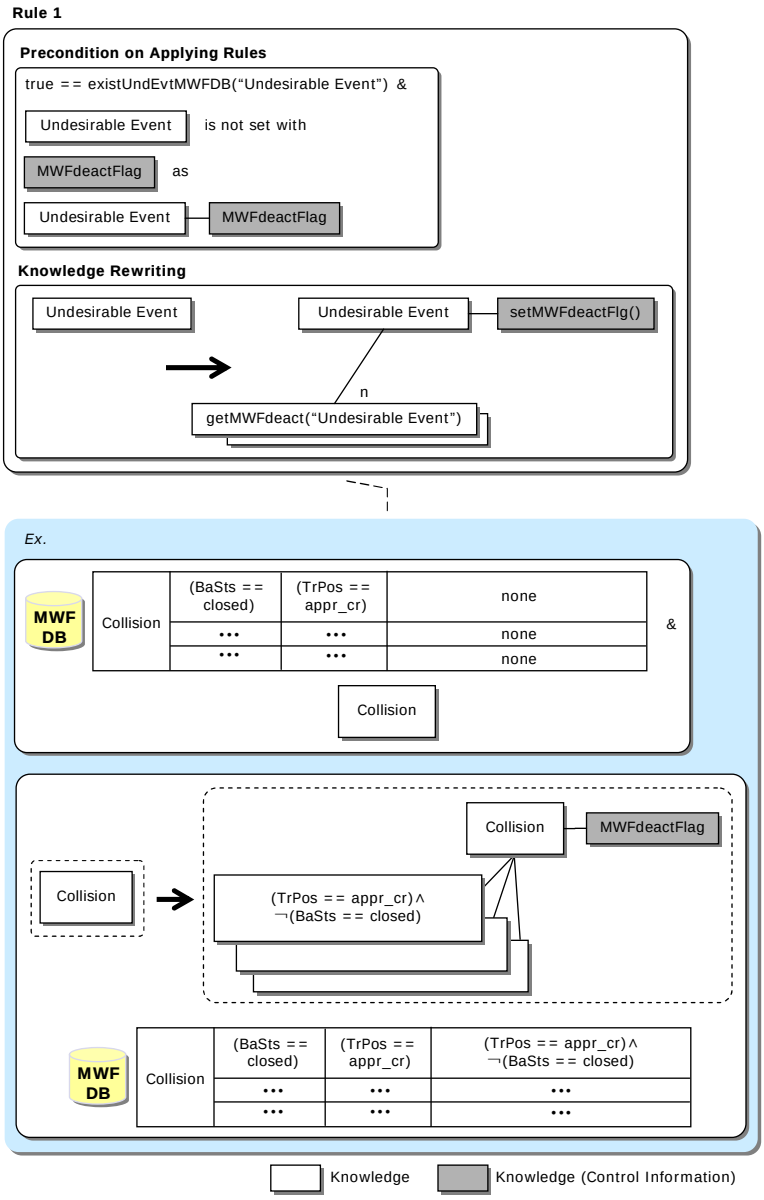


Fig. 7 Rule 1: Embodiment in terms of MWF Inactivation

Rule 2 is shown in Fig. 8. The concept of Rule 2 is similar to that of Rule 1.

 

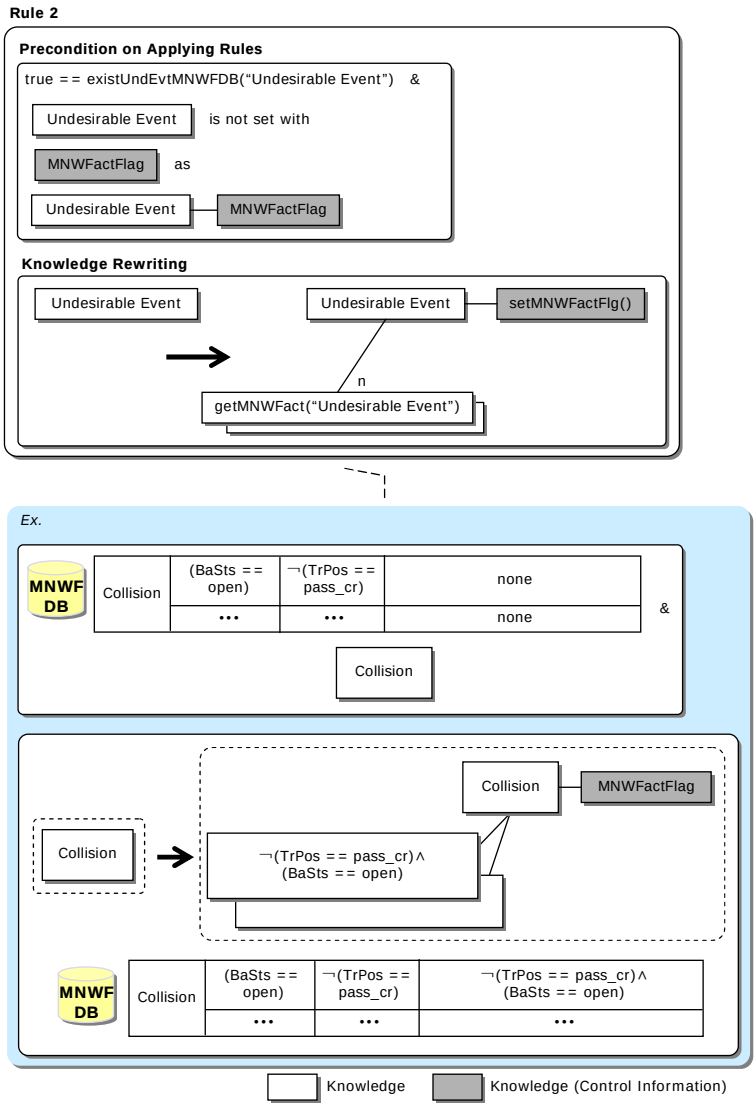


Fig. 8 Rule 2: Embodiment in terms of of MNWF Activation

As described in Section III.A, "MWF Inactivation" event is replaced by undesirable event after the knowledge rewriting process of Rule 1. Similarly, "MNWF Activation" event is replaced by undesirable event after the knowledge rewriting process of Rule 2. If those undesirable events exist in the MWF DB or the MNWF DB, the knowledge rewriting rule is again applied to the undesirable events.

### *E. Developing Safety Properties*

Specific and rigorous safety properties are developed based on the undesirable events embodied in terms of "MWF Inactivation" and "MNWF Activation". The safety properties state that undesirable events never occur. Therefore, the embodied undesirable events are negated (¬), and temporal logic operators are applied to the negated events. When the safety properties are represented in CTL, the temporal logic operators A and G are applied to the negated events. When the safety properties are represented in LTL, the temporal logic operator G is given. In the case of the wireless rail crossing system, "AG ¬((TrPos == appr_cr)∧¬(BaSts == closed))" is developed as a safety property based on the embodied undesirable event "(TrPos == appr_cr)∧¬(BaSts == closed)" when the safety property is expressed in CTL.

 

## IV. EVALUATION

We conduct an evaluation with the proposed method and confirm its effectiveness. First, safety properties are derived by traditional means. Next, specific and rigorous safety properties are derived using the proposed method. We confirm that the proposed method produces safety properties that were not derived by traditional means. We also confirm that the safety properties derived by traditional means can be improved by the proposed method.

### *A. Application Example*

We choose the wireless rail crossing system [4], [5] as an evaluation case. This is a complex system based on an actual development by German Railways, and is controlled by a computer system. An image of the wireless rail crossing system is shown in Fig. 9. When a train approaches, a wireless message is sent to the rail crossing. The rail crossing is equipped with signals and barriers for road users such as cars and pedestrians. Road users wait behind the barriers when the signals are turned on. Sensors at the rail crossing determine when the train has passed.

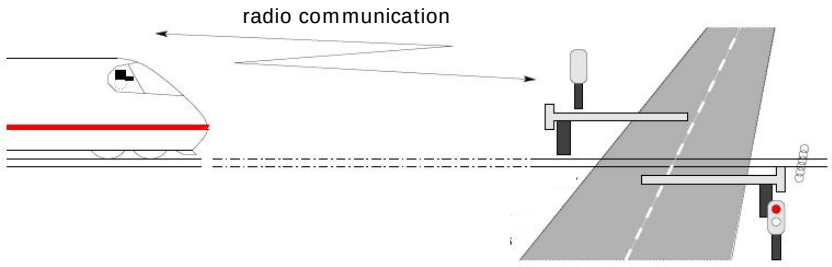


Fig. 9 Wireless rail crossing system

### *B. Subjects for Evaluation*

We chose four engineers as the subjects of the evaluation. We call these subjects A, B, C, and D. They have 15, 13, 11, and 4 years of work experience related to computer systems, respectively. Each subject has experience of model checking.

### *C. Deriving Specific Safety Properties*

Before starting to derive safety properties for model checking, the subjects familiarised themselves with the wireless rail crossing system according to the specifications in [5]. They were able to refer to [5] while deriving safety properties by traditional means and the proposed method. In the evaluation, "Collision" event was selected as the undesirable event for the wireless rail crossing system. The safety properties derived by traditional means and using the proposed method are as follows. The specific safety properties are represented in CTL.

*1) Traditional means*: Each subject derived safety properties against a "Collision" event according to their work experience on computer systems and their own key points of model checking. As an example, the specific safety properties derived by subject A are shown in Fig. 10. The IDs expressed within the square brackets are unique to each derived safety property.

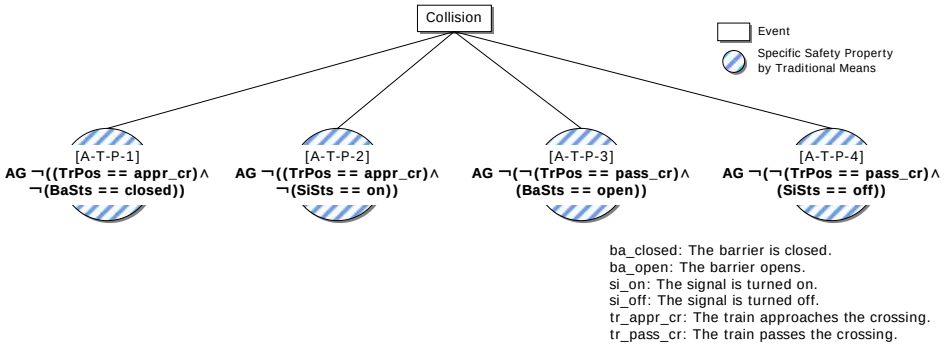


Fig. 10 Derived safety properties using traditional means (Subject A)

 

*2) Proposed Method*: Subjects were taught to use the proposed method after they had derived safety properties by traditional means. Each subject derived specific and rigorous safety properties against the "Collision" event using the proposed method. As a specific example, the safety properties derived by subject A are shown in Fig. 11. Again, the IDs expressed within square brackets are unique to each derived event and safety property. The terms "MWF Inactivation" or "MNWF Activation" between events expresses how the events were derived. For example, the event A-P-E-3 was derived in terms of "MWF Inactivation" from "Collision", and event A-P-E-3-3 was derived as an "MNWF Activation" from A-P-E-3. Events in parentheses (e.g., (A-P-E-1-1) in the A-P-E-2-1 box) denote identical events. Events within dotted boxes are the last embodied events in terms of "MWF Inactivation" and "MNWF Activation".

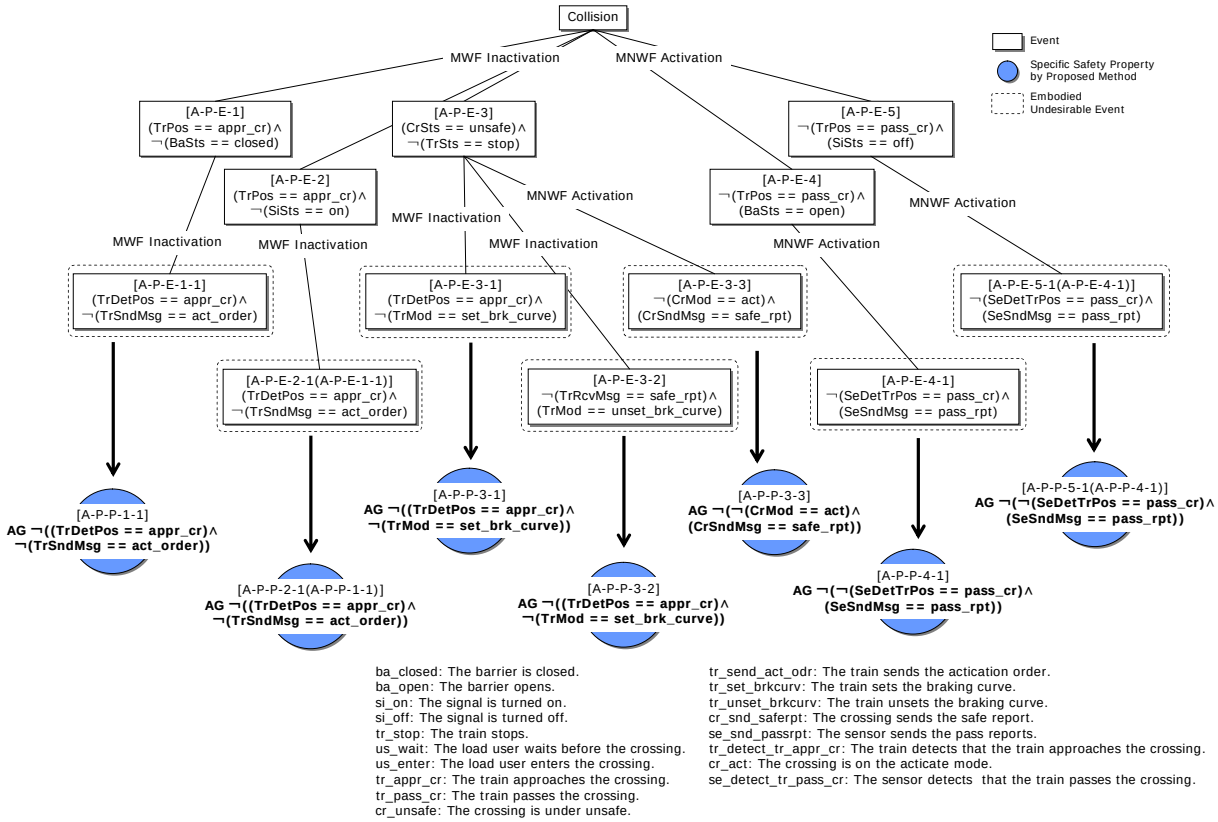


Fig. 11 Derived safety properties using the proposed method (Subject A)

## *D. Results*

A comparison of the safety properties derived by subject A using traditional means (Fig. 10) and the proposed method (Fig. 11) is shown in Fig. 12. In this figure, A-P-E-1 derived by the proposed method becomes A-T-P-1 derived by traditional means when being negated and given temporal logic operators. Therefore, A-P-E-1 and A-T-P-1 are regarded as one. Similarly, A-P-E-2 and A-T-P-2, A-P-E-4 and A-T-P-3, and A-P-E-5 and A-T-P-4 are the same. As a result of the evaluation, it can be seen that safety properties A-P-P-3-1, A-P-P-3-2, and A-P-P-3-3 derived by the proposed method were not derived by traditional means. Based on this result, the comprehensive nature of deriving safety properties by the proposed method has been confirmed. In addition, the specific and rigorous safety properties A-P-P-1-1, A-P-P-2-1 (A-P-P-1-1), A-P-P-4-1, and A-P-P-5-1 (A-P-P-4-1) derived by the proposed method were embodied from A-T-P-1, A-T-P-2, A-T-P-3, and A-T-P-4, respectively, given by the traditional means. This result demonstrates that the specificity of the safety properties has been improved by the proposed method.

 

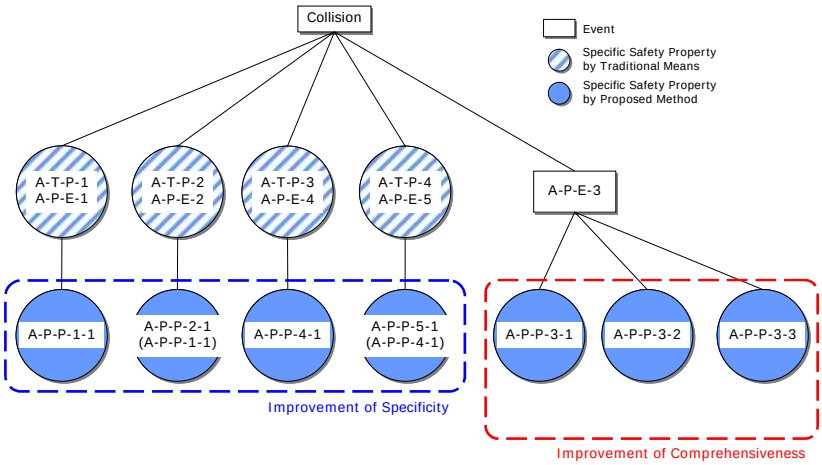


Fig. 12 Comparison of derived safety properties (Subject A)

Table I shows the evaluation results for each subject. In this table, column 2 denotes the number of safety properties derived by traditional means, and column 3 shows the enhancement by the proposed method. In Fig. 12, such enhancements are A-P-P-3-1, A-P-P-3-2, and A-P-P-3-3. Column 4 shows the number of safety properties that have been made more specific by the proposed method. In Fig. 12, A-P-P-1-1, and A-P-P-4-1 are examples of this. Events A-P-P-2-1 and A-P-P-5-1 are the same as A-P-P-1-1 and A-P-P-4-1, respectively. Therefore, they are not counted in Table I. Column 5 gives the relative improvement in terms of the additional safety properties derived by the proposed method. In the case of subject A, this improvement in comprehensiveness is calculated from the number of safety properties given by traditional means, 4, and the additional safety properties given by the proposed method, 3, i.e., ¾ = 75%. The final column shows the relative improvement in terms of the number of safety properties embodied by the proposed method. In the case of subject A, the improvement is 50%, which is calculated from the number of safety properties given by traditional means, 4, and the number of embodied safety properties given by the proposed method, 2.

From the results of the evaluation, we confirmed that the comprehensiveness was improved by the proposed method in subjects A, B, and C. We also confirmed that the proposed method improved the specificity of the safety properties in all subjects.

TABLE I RESULTS OF EVALUATION FOCUSED ON COMPREHENSIVENESS AND SPECIFICITY

| Subject | Safety Properties by Traditional Mean | Num. of Safety Properties additionally Derived by Proposed Method | Num. of Safety Properties embodied by Proposed Method | Improvement of comprehensiveness | Improvement of specificity |
|---|---|---|---|---|---|
| A | 4 | 3 | 2 | 75% | 50% |
| B | 6 | 3 | 3 | 50% | 50% |
| C | 5 | 3 | 3 | 60% | 60% |
| D | 4 | 0 | 3 | 0% | 75% |
| Total | 19 | 9 | 11 | 47% | 58% |

### *E. Model Checking with Derived Safety Properties*

We applied model checking to the wireless rail crossing system with the safety properties derived by the proposed method. UPPAAL [15], where properties are represented in CTL, was used as the model checking tool. As a result, it was confirmed that the wireless rail crossing system could be verified with the safety properties derived by the proposed method.

## V. DISCUSSION

The effectiveness of the proposed method has been demonstrated by the evaluation results in Section IV. The comprehensiveness of the rigorous safety properties derived by subjects A, B, and C was improved by the proposed method. In Fig. 12, safety properties A-P-P-3-1, A-P-P-3-2 and A-P-P-3-3 given by subject A were not derived by traditional means. In the proposed method, undesirable events are embodied in a systematic manner using the two concepts of “MWF inactivation” and “MNWF activation”. This enables the proposed method to improve the comprehensiveness of the derived safety properties. For subject D, no improvement in

 

comprehensiveness was observed. We now examine the reasons for this based on the properties derived by this subject. A comparison of the safety properties derived by traditional means and using the proposed method is shown in Fig. 13. As this figure shows, a safety property derived by traditional means (e.g., D-T-P-1) corresponds to an event derived by the proposed method (D-P-E-1). D-T-P-2 and D-P-E-2 have the same relation. The derived safety properties given by traditional means (D-T-P-3) are the same as those from the proposed method (D-P-P-3). The relation between D-T-P-4 and D-P-P-1-1 is the same. It is postulated that subject D already thinks in terms of "MWF Inactivation" and "MNWF Activation" based on his experiences and skills. The wireless rail crossing system may also be an easily understandable system for subject D. These factors may explain why there were no differences between the safety properties derived by traditional means and by the proposed method. If the target of the proposed method is a system that does not have physical behaviours or spatial arrangements whose behaviours are difficult to understand, the improvement in the comprehensiveness of safety properties given by the proposed method will be more noticeable.

We confirmed that the proposed method improved the specificity of safety properties in all subjects. In Fig. 12, a safety property derived by traditional means (e.g., A-T-P-1) corresponds to a safety property derived by the proposed method (A-P-E-1). The specific safety property A-P-P-1-1 was derived from this event in terms of "MWF Inactivation". Similarly, the safety property A-P-P-4-1 was derived from A-T-P-3/A-P-E-4 in terms of "MNWF Activation". As shown in Section III.C, the process of identifying MWFs and MNWFs in the proposed method is iterative. It is considered that this iteration process for identifying MWFs and MNWFs contributes to the embodiment of appropriate safety properties in the proposed method.

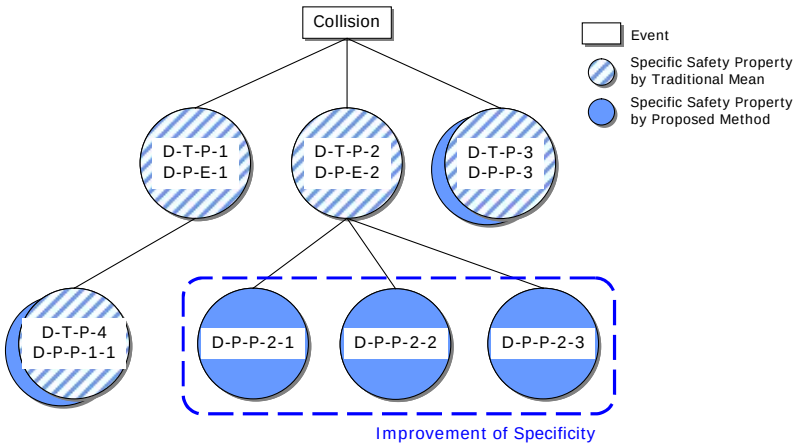


Fig. 13 Comparing derived safety properties (Subject D)

## VI. CONCLUSIONS

This paper described a method for deriving specific, rigorous safety properties in model checking. In the proposed method, an undesirable event in a system is embodied by the concepts of "MWF Inactivation" and "MNWF Activation". Through these, specific safety properties can be derived from the undesirable event. A wireless rail crossing system was chosen to evaluate the proposed method. The results confirmed that the proposed method produced safety properties that were not derived by traditional means. Also, it was confirmed that the safety properties derived by traditional means were further embodied by the proposed method. From these evaluation results, we confirmed that the proposed method improves the comprehensiveness and specificity of derived safety properties.

Further research should be focused in the following directions. The proposed method is based on a prerequisite that undesirable events have already been identified. However, methods to identify undesirable events are not well established. Identification of undesirable events is very difficult before they occur, and they are sometimes not recognized as undesirable until they occur. In the future, we will establish a systematic method to identify undesirable events, and establish a more comprehensive model checking method for complex systems by incorporating this method with that proposed in this paper.

## REFERENCES

[1] N. G. Leveson, *Safeware: System Safety and Computers*, New York, USA: Addison-Wesley, 1995.
[2] M. E. Clarke, O. Grumberg, and E. D. Long, "Model checking and abstraction," *ACM T Progr Lang Sys.*, vol. 16, pp. 1512–1542, Sep. 1994.
[3] B. Alpern and B. F. Schneider, "Defining liveness," *Inform Process Lett.*, vol. 21, pp. 181–185, Oct. 1985.
[4] *Betriebliches Lastenheft fur Funkfahrbetrieb. stand 1.10*, 1996.

 

[5] F. Hansel, J. Poliak, R. Slovak and E. Schnieder, "Reference Case Study 'Traffic Control Systems' for Comparison and Validation of Formal Specifications Using a Railway Model Demonstrator," *Integration of Software Specification Techniques for Applications in Engineering*, pp. 96–118, 2004.
[6] *Computer-Based Control System Safety Requirements*, National Aeronautics and Space Administration, SSP 50038 Revision B, 1995.
[7] J. W. Klop, *Term rewriting systems, Handbook of Logic in Computer Science, Volume 2. Background: Computational Structures*, Oxford University Press, Oxford, UK, pp. 1–116, 1992.
[8] *Fault Tree Handbook with Aerospace Applications*, National Aeronautics and Space Administration, version 1.1, 2002.
[9] *Fault Tree Handbook*, U.S. Nuclear Regulatory Commission, NUREG-0492, 1981.
[10] S. Henry and M. J. Faure, "Elaboration of invariant safety properties from fault-tree analysis," in *Proc. IMACS-IEEE Computational Engineering in Systems Applications*, 2003.
[11] S. I. Barragan and M. J. Faure, "From Fault Tree Analysis to Model Checking of Logic Controllers," in *Proc. 16th International Federation of Automatic Control World Congress*, 2005.
[12] B. Yan, M. Nakamura and K. Matsumoto, "Deriving Safety Properties for Home Network System Based on Goal-Oriented Hazard Analysis Model," *International Journal of Smart Home*, vol. 3, pp. 67–79, Jan. 2009.
[13] A. T. Morris and M. J. Massie, "Analyzing Distributed Functions in an Integrated Hazard Analysis," in *Proc. AIAA Infotech@Aerospace Conference*, AIAA-2010-3486, 2010.
[14] E. Clarke, E.A. Emerson and A.P. Sistla, "Automatic verification of finite-state concurrent systems using temporal-logic specifications," *ACM Trans. Program Lang Syst.*, vol. 8, pp. 244–263, Apr. 1986.
[15] G. K. Larsen, P. Pettersson and W. Yi, "UPPAAL in a nutshell," *International Journal on Software Tools for Technology Transfer*, vol. 1, pp. 134–152, Dec. 1997.

 