Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 196107, 11 pages
http://dx.doi.org/10.1155/2015/196107

Research Article

Extended FRAM by Integrating with Model Checking to Effectively Explore Hazard Evolution

Guihuan Duan, Jin Tian, and Juyi Wu

School of Reliability and Systems Engineering, Beihang University, Beijing 100191, China

Correspondence should be addressed to Jin Tian; tianjin@buaa.edu.cn

Received 8 June 2015; Accepted 12 October 2015

Academic Editor: Yuanchang Xie

Functional Resonance Analysis Method (FRAM), which defines a systemic framework to model complex systems from the perspective of function and views accidents as emergent phenomenon of function's variability, is playing an increasingly significant role in the development of systemic accident theory. However, as FRAM is typically taken as a theoretic method, there is a lack of specific approaches or supportive tools to bridge the theory and practice. To fill the gap and contribute to the development of FRAM, (1) function's variability was described further, with the rules of interaction among variability of different functions being determined and (2) the technology of model checking (MC) was used for the analysis of function's variability to automatically search the potential paths that could lead to hazards. By means of MC, system's behaviors (normal or abnormal) are simulated and the counter example(s) that violates the safety constraints and requirements can be provided, if there is any, to improve the system design. The extended FRAM approach was applied to a typical air accident analysis, with more details drawn than the conclusions in the accident report issued officially by Agenzia Nazionale per la Sicurezza del Volo (ANSV).

1. Introduction

With the increasing complexity in sociotechnical systems, accident models are playing significant roles in explaining why an accident occurs, and the ways that hazards go to an accident can be identified based on the understanding of accident causes. At present, there are three categories of accident models widely acknowledged: the causal sequence models represented by domino model [1], the epidemiological models like Swiss cheese model [2], and the systematic models such as FRAM (Functional Resonance Analysis Method) [3], and STAMP (Systems-Theoretic Accident Model and Processes) [4]. Different from the former two categories in which accident was considered as a sequence of a series of unexpected incidents or as a result of combinations among factors involving human, working environment and media, the systematic models take accident as emergence due to nonlinear interactions among technical, human, and organizational factors within sociotechnical systems and explain the accident by determining the latent deviations of the system operations from what they should be.

As a typical systemic accident model, FRAM is capable of comprehensively analyzing complex sociotechnical systems from the perspective of function and describing interactions and couplings among the functions. Based on FRAM, an accident would be taken as "resonance" of multiple functions' variability, which is an innovative perspective to look at accident and effectively assist safety analysis and accident investigation. Nevertheless, FRAM is on the way of continuous development for the reasons below.

On one hand, the descriptions of function's variability and the spreading rules among functions need to be further elaborated to ensure the rigor and comprehensiveness of variability analysis. In terms of the classic FRAM, six aspects of a function are described and the variability going between aspects of upstream and downstream functions is explained with a rough categorization of "timing" and "precision," and it lacks details such as the rules about how the variability spreads from a function to another. Hence, we believe that the bias in analysts' mind can hopefully be minimized when they conduct variability analysis, if a set of rigorous and practical instructions are available.

On the other hand, FRAM is more like a conceptual method so far than a mature model [5], and it is significant to call on some corresponding approaches or supporting tools based on FRAM so as to contribute to the development of FRAM as such. Since the variability in a single function and between functions is basically hand-picked, there tends to be low efficiency and poor thoroughness in the analysis. Hence, the efficient supporting tools or approaches, based on the function variability and spreading rules defined by FRAM, are necessary to ensure that all the potential states and behaviors of the given system be checked, with the aim of determining whether the paths that variability spreads among functions may lead to an accident. Both the completeness and efficiency can be guaranteed with the aid of appropriate computer tools such as model checking [6], by which all the potential function states and sequences can be automatically searched.

Therefore, in this paper, FRAM was enhanced to explore the paths of hazard evolution by integrating with model checking. The rest of this paper is structured as follows: the background research and related work are reviewed in Section 2, and the method is described in Section 3. An air accident is taken into case study to illustrate the proposed approach in Section 4. Finally, Section 5 sets out conclusions and future work.

2. Literature Review

As a systemic accident model, FRAM was presented first by Hollnagel [5, 7]. It was pointed out that accidents were the resonance and amplification among functions' variability. In FRAM, structural models were used to describe functions and further to analyze aggregations of function variability. FRAM can be used to identify functional or logic deficiencies in system design, in addition to failures in hardware or software. Some comparisons were made between FRAM and other methods to discuss the pros and cons of both. In 2008, Hollnagel [8] elaborated the shortages of traditional safety analysis technologies and the advantages of FRAM, and concluded that FRAM could be more beneficial to facilitate safety analysis of key information system. Based on the research, Herrera and Woltjer [9] compared Sequentially Timed Events Plotting (STEP) with FRAM from aspects of rationale and application, respectively. The results show that FRAM can identify the accident causes that were not found with STEP and justify the advantage of FRAM to analyze the nonlinear and dynamic systems, such as sociotechnical systems. In addition, facing the challenge of current accidents, Hovden et al. [10] suggested that new theories, models, or methods such as FRAM be developed aiming at the "foresight" for accident prevention.

Moreover, FRAM has been used in different fields and demonstrates its significance and contribution to industrial practice, specifically for accident investigation and accident analysis. Based on FRAM, Woltjer [11] discussed the categories of all the contributing factors in aviation accidents and then explained how the resonance happened among human, technical, and organizational causes. Sybert et al. [12] pointed out that FRAM lacked system assessment on interactions between functions and variability in performance during hazard identification in Air Traffic Control (ATC), by analyzing the elastic characteristics of ATC system and confirming the variability existing in the system behaviors [13]. Besides, FRAM was applied to analyzing air accidents, and its effectiveness was verified by Hollnagel et al. [14] and Sawaragi et al. [15]. FRAM has also been adopted in train control, nuclear power, and electric systems; for example, Belmonte et al. [16] analyzed the safety of Automatic Train Monitoring System (ATS) through FRAM, and Macchi et al. [17] applied FRAM on the maintenance in nuclear power plant to explain the principle of the local maintenance activities and the possible influences on system safety.

The applications above indicate that FRAM can facilitate safety analysis and accident investigation and contribute to identification of more details of hazards than the traditional methods. However, in order to further develop FRAM, one of the key points is how to determine function variability and the rules of variability spreading from a function to another, as well as how to conduct efficient and complete search (based on the spreading rules) through all the combinations of the function variability. To make the rules derived and the search realizable, model checking can be adopted. Model checking is a widely used technology with which system's behaviors are described as transition among system states, and system properties are represented with temporal logic formulae, and thus all the possible behaviors can be automatically searched for any unexpected state sequences [6].

At present, there is significant development in model checking. Many logic expressions and rules have been extended to get adapted better to different systems [18–20]. Furthermore, great efforts were made to solve the key problems from which model checking often suffered such as the "explosion" of state space and the time synchronization [21, 22]. Model checking has a very wide range of applications, covering software, network, chemical industry, and other fields [23–25], in which it is taken as a comparatively mature means for justifying whether the system meets a given specification by modeling and simulating a complex system. Particularly, model checking has been used to develop safety analysis and random probability analysis from the perspective of function in hybrid systems [26]. With enlightenment from the application above, it is assumed that model checking can be used to simulate potential function states and sequences based on FRAM. Gao et al. [27] extended continuous stochastic logic to conditional continuous stochastic logic (CCSL) by introducing a conditional probabilistic operator to describe a richer class of properties for continuous-time Markov chains. In Rushby's research [28], model checking was used to analyze the autopilot accident, and it was concluded that the accident was caused by the fact that the cognitive process of human communication had not yet been covered completely. Zhang et al. [29] applied model checking to building information system to evaluate the risks and develop preventive measures, and Lahtinen et al. [30] discussed the sense that model checking has made in the nuclear industry and proposed a systemic method to justify the model when using it for safety-critical systems.

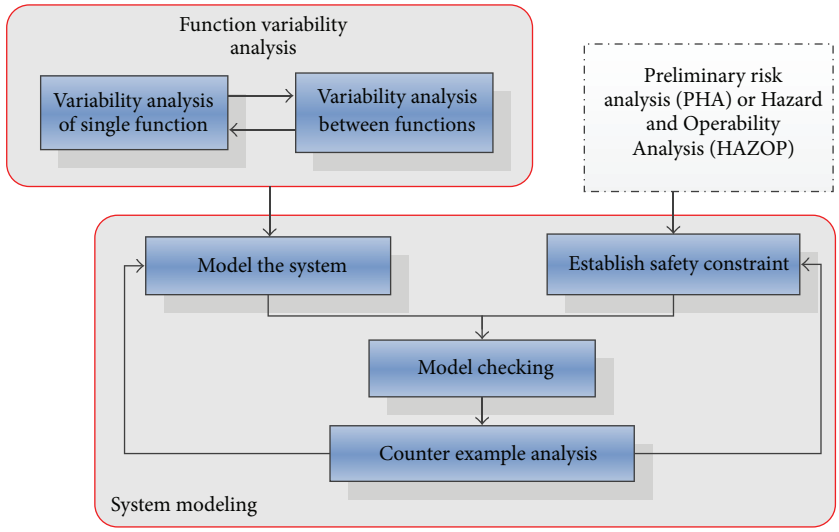

Figure 1: The framework of hazard identification based on the extended FRAM.

Overall, the existing research provides the evidence that it is feasible to analyze accidents by means of model checking, but model checking has not yet been adopted for safety analysis from the perspective of system functions and their variability, although it was recognized to be able to model and simulate system functions. Accidents would be explained in more details by simulating variability of system's behaviors, if combining model checking and FRAM, to demonstrate the scenarios describing why and how accidents occur.

3. Method

Hazards are defined as the states that may lead to any accident or unexpected event [31]. Based on the rationale of FRAM, the hazard evolution can be viewed as a process that variability is propagated among functions with increasing (or decreasing) magnitude that may violate the safety constraints and lead to accidents. The framework of identification of hazards and their evolution is illustrated in Figure 1. Firstly, FRAM was extended by redefining and clarifying some critical terms and deriving the criterion for function variability, both which were taken as supplementary to the original FRAM. Based on the extended FRAM, an approach of system modeling with detailed steps was provided to show how model checking was used to search all the system states for the potential paths which may lead the system to an accident. The two parts are marked with red blocks in Figure 1 and elaborated in the following subsections, respectively.

3.1. Function Variability Analysis. According to the original FRAM, the characteristics of function are described as the hexagon shown in Figure 2, and the six angles are labeled with the aspects of function: Input, Output, Precondition, Resource, Time, and Control, respectively.

Input is used to start or begin a function; Output represents the product or outcome generated from a function; Precondition is the conditions that need to be ready before

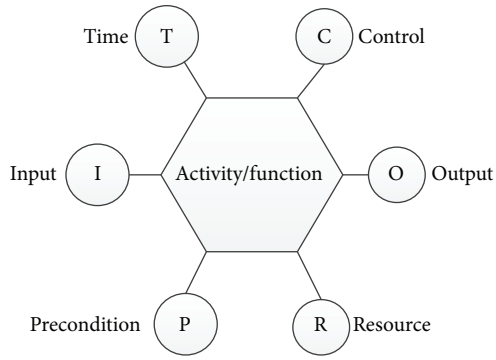

Figure 2: A hexagon representing a function [5].

running a function; Resource refers to the materials that are consumed to run a function or execution condition that is essential for a function; Time is the schedule or the time window that a function needs to follow; and Control is used to supervise and constrain a function, and it can be a set of specific plans, procedures, or guidelines [5]. For convenience, the phrase Five Aspects mentioned in the following text refers specifically to Input, Precondition, Resource, Time, and Control.

The phenomenon that a function does not perform as fully as designed or expected is called function variability [5]. On one hand, due to the instability and variation of the external environment around the systems, the aspects of a function are likely to be affected by the environment, which hence makes the function deviate from its behavior desired by designers. On the other hand, a function may be connected with other function(s) in the same system; for example, the Output of the upstream function can be the Input of the downstream function(s). The variability may possibly spread from the upstream to the downstream function(s) in some forms, once it does exist in the upstream function; therefore, a function's variability may come from

the variability of its upstream function through interactions between the functions. To make it clearer, a pair of definitions is presented as follows.

Definition 1. Variability in a single function that refers to how the Output of a function is influenced by the variability in one or more of the Five Aspects of the same function.

Definition 2. Variability between functions refers to how the aspect(s) of a function (Function A) is influenced by the variability in the Output of its upstream function (Function B), when the Output of Function B is related to one of the Five Aspects of Function A.

Hollnagel defined the term "aggregation" as "functional upstream-downstream coupling" [5], which means that how the upstream Output can have effect on the Five Aspects of the downstream function. It is pointed out that the variability in the upstream Output would correspond to the downstream function, which indicates that the variability would not vary when being passed between the upstream and downstream functions. However, we tend to assume that variability exists not only in a single function but also in the spreading process between functions. For example, in the task "a file is sent from Computer A to Computer B;" the upstream function is "Computer A provides the file," and the downstream function is "Computer B receives the file." The following incident may occur: the file provided by Computer A is normal and correct whereas Computer B receives the file with virus attached due to some unknown attacks during the delivery through the information network. Herein, the Output of upstream function is corresponding to what it is expected to be, but when it becomes to the Input of downstream function, deviation arises which in some sense can be taken as variability. It is shown that the Output variability of upstream function is likely not to correspond to the downstream function, thus in this paper the "aggregation" is classified into two types: variability in a single function and variability between functions, to make safety analysis as complete as possible.

For each function, any of the Five Aspects may deviate from its expected situation, and in most of cases the Output may deviate from its expected states accordingly. However, since the Five Aspects play different roles in a certain function, the variability of them can contribute to the Output variability in different ways. The potential Output variability caused by the Five Aspects of the same function is explained in Table 1. For a specific function, the variability analyzed is likely to be some (not all) of the items shown in Table 1, which actually covers potential variability as complete as possible. With the aid of Table 1, analysts can determine the variability in accordance with the features of each function. It is noted that an aspect (e.g., Input) may have presence which can be accepted by the function and even taken as normal for some reasons even if it is provided improperly, despite the fact that actually it is abnormal from the view of a whole system. In this way, the variability of Output can be relatively predictable once the variability of the other aspects is determined.

Table 1: The Output variability from a single aspect.

Aspect	Variability of the aspect	Output variability
Input	Earlier	Earlier, normal, later, or omitted
	Later	Later or omitted
	Erroneous	Erroneous or omitted
	Imprecise	Imprecise or omitted
	Omitted	No output
Precondition	Earlier	Normal
	Later	Later or omitted
	Imprecise	Normal, erroneous, or omitted
	Omitted	Normal, omitted, or erroneous
	Erroneous	Omitted or erroneous
Resource	Imprecise	Imprecise, erroneous, or normal
	Later	later, imprecise, or erroneous
	Omitted	Omitted
	Erroneous	Erroneous
Time	Shorter duration	Insufficient or erroneous
	Longer duration	Exceeding or overflow
	Omitted	Omitted, erroneous, insufficient exceeding, or overflow
	Imprecise	Imprecise or erroneous
Control	Imprecise	Imprecise, normal, or erroneous
	Earlier	Imprecise or normal
	Later	Imprecise, erroneous, or normal
	Omitted	Erroneous, imprecise, or normal
	Erroneous	Erroneous

In a function, variability may exist more often in two or more aspects simultaneously rather than in a single aspect, since a function may be connected with two or more other functions. For example, the Input and Precondition of Function A are connected, respectively, with the Outputs of Functions B and C. Furthermore, the Five Aspects may have different influences on different types of function; hence, the aspect weighting higher should be paid more attention to if variability exists in more than one aspect. Essentially functions vary with the characteristics of their Outputs. In this paper, functions are classified into four categories: material handling, energy transfer, information/data processing, and state change, according to which the Output variability can be summarized in Table 2.

In case that the variability of two aspects of the same function works simultaneously, more attention should be paid to the variability of the aspect contributing more to the Output. The parameter q_i $(i = 1, 2, \ldots)$ is used to present the relative importance of aspects' variability, and it is determined according to the ratio of impact of the aspect's variability to that of Input's variability. The parameter value is always an integer mostly depending on features of the specific function, but q_i $(i = 1, 2, \ldots)$ is set as 1 in most cases. It is exemplified with the function "calculate aircraft's weight and

Table 2: The Output variability for different types of functions.

Types of function	Potential variability of Output
Material handling	Earlier, later, imprecise (quantitatively), erroneous (with an incorrect target), and no output
Energy transfer	Earlier, later, imprecise (insufficient quantitatively, or unstable), erroneous (quantitatively, or with an incorrect type of energy), and no output
Information/data processing	Earlier, later, imprecise (quantitatively), erroneous, no output
State change	Earlier, later, imprecise (incomplete change from a state into another), erroneous (with an incorrect target), and no output

gravity balance," which is an instance in the type "information/data processing": given the Resource "passenger's weight or calculation formula," the Output would deviate more or less when something wrong happens to the Resource, for example, an incorrect formula provided. Considering that the aspects affect function in different ways, we assume that the parameters can be used to describe variability. Given that the variability of Precondition is represented with Δf_1, the variability of Input with Δf_2 and the contributing weights of the Precondition and the Input are represented with q_1 and q_2, respectively, the variability of the Output is described as in the following equation:

$$\Delta F = q_1 * \Delta f_1 + q_2 * \Delta f_2. \quad (1)$$

Output of the upstream function can be Input, Time, Control, Resource, or Precondition of the downstream function. When analyzing the variability between each pair of functions related to each other, the spreading from upstream to downstream can also be taken as a certain function. Based on the possible variability of upstream Output (for which the possible variability can be determined as Table 2) as well as interactions between the upstream and downstream functions, the possible variability of the downstream aspect can be analyzed accordingly, which is shown in Table 3.

3.2. System Modeling with Model Checking. Given that the rules of variability and its spreading from upstream to downstream function have been established, the issue "whether the system meets the safety requirements and does not violate the safety constraints" could be interpreted into a model describing "whether the state transitions within the system satisfy the temporal logic formulae derived from the rules of variability propagation." Based on the model, the system behaviors can be simulated with model checking [18] by following the three steps below.

Step 1 (describe state transitions within system). First, the definition about state transitions within system is given below.

Definition 3. In terms of FRAM, function variability roughly indicates two states of function: standard and deviate. The state of a function is taken equivalent to that of its Output, as the function variability is basically reflected with its Output.

To depict the changes in system behaviors, State Transition Diagram (STD) is used, where states are represented with circles, and conditions for state transition are represented with links. By means of model checking, the STDs are beneficial for describing the process that the state change in a function influences that in the other functions related to it, so as to model the functional behaviors of the whole system. The model structure of functional state transition is shown in Figure 3.

Table 3: The variability of the downstream aspect in terms of upstream Output.

Aspect of downstream function	Variability of upstream Output	Variability of downstream aspect
Input	Earlier	Earlier, later, or normal
	Later	Later or omitted
	Omitted	Omitted
	Imprecise	Imprecise, normal, or omitted
	Erroneous	Erroneous
Precondition	Earlier	Earlier, later, normal, or omitted
	Later	Later or omitted
	Omitted	Omitted
	Imprecise	Erroneous or omitted
	Erroneous	Erroneous or omitted
	Insufficient	Insufficient, erroneous, or omitted
Resource	Earlier	Normal
	Later	Omitted or insufficient
	Omitted	Omitted
	Insufficient	Omitted or insufficient
	Erroneous	Erroneous
Time	Earlier	Earlier
	Later	Later
	Higher value	Lasting longer
	Lower value	Lasting shorter
	Erroneous	Erroneous
	Omitted	No requirement about time
Control	Earlier	Earlier or normal
	Later	Later or omission
	Imprecise	Erroneous or imprecise
	Erroneous	Erroneous
	Omitted	Omitted

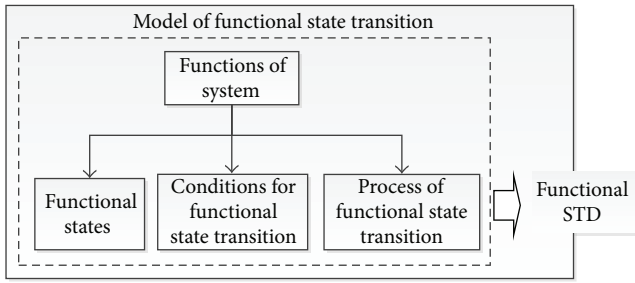

Figure 3: The model structure of a functional state transition.

There is assumed to be six functional states: standard, no output, erroneous, earlier, later, and imprecise, all of which are represented with circles and labeled, respectively. Taking Function 7 "ATC confirmed business jet position" as an example (note: please see Section 4 for all the nine functions identified in the case), "output == 0" means Function 7 is in a standard state, whereas "output == 1" means it is in any of deviate states, that is, any of the latter five states above. Then based on the function variability, the conditions for the transition from deterministic state of a function's Input to the different potential Output are analyzed, and Precondition, Resource, Control, and Time are represented with specific arrays or parameters, as well as the transition from upstream to downstream function with logic formulae or IF statements. To continue the example of Function 7, variability arises in the Output when there is variability in the Resource, where the conditions for state transition can be expressed with "output[j] = res[j]" and the j right means Function 7.

Based on the functional states as well as spreading between functions that have been clarified, the system behaviors can be modeled and simulated by means of model checking. The variability of Input and Output is defined as parameters, respectively, through which variability can be thus characterized. For example, the variability of Output is described with (2) if there is any variability in the sense of timing or precision:

$$v = \begin{cases} i, & \left(\dfrac{\text{Time earlier}}{\text{Value larger}}\right), \\ 0, & (\text{No variation}), \\ -j, & \left(\dfrac{\text{Time later}}{\text{Value smaller}}\right). \end{cases} \tag{2}$$

The numerical level of i and j indicates the degree of variability. The larger the value, the more considerable the variability, so the performance of functions and thus the system states can be described by these parameters. As a preliminary principle of variability comparison proposed in this paper, it is assumed that $i = 1$ when a function varies and $i = 0$ when it does not.

According to the rules of variability spreading between the aspects in a function as well as between functions that are determined in FRAM analysis, the conditions for functional state transition can be defined and then the system model established. For example, assuming that the variability of Input is described as i and the variability of Precondition is described as j, and the weight of Input and of Precondition is represented as q_1 and q_2, respectively, the Output is explained as follows:

$$\text{output} = i * q_1 + j * q_2. \tag{3}$$

The conditions for state transitions are expressed as follows:

$$\{\text{output} = i * q_1 + j * q_2\}, \quad \text{if } (\text{input} == i \,\&\&\, \text{precondition} == j \,\&\&\, \text{input\_weight} == q_1 \,\&\&\, \text{precondition\_weight} == q_2). \tag{4}$$

Step 2 (determine safety constraints). Safety constraints are typically developed in terms of the unexpected states or events and mostly converted into safety requirements during the system design. Safety constraints can be taken as the criteria in position, during model checking simulation, for justifying whether the system is safe or not if going along the given path of functional state transition and for further identifying the path(s) leading to accidents after checking all the potential paths. Herein, safety constraints are categorized into two types: (1) critical functional constraints, which means that some deviation of a certain function causes an accident directly and (2) combining functional constraints, which means an accident is caused by the mixture of deviations of more than one function. The second type is more typical in most cases of sociotechnical systems, since in terms of the rationale of resilience engineering, sociotechnical systems are likely to be self-adjustable and keep working normally even though some of their functions deviate from the desired performance, because the deviations can be dampened or mitigated through the interactions among functions.

In order to develop safety constraints, the potential hazards need to be identified beforehand, which is even treated as the indispensable step within system modeling. The hazards, defined here as those that may cause deaths, injuries, damage to equipment, or environmental pollution, can be identified by means of Preliminary Hazard Analysis (PHA) [32] or Hazard and Operability Analysis (HAZOP) [33]. After being determined, safety constraints are interpreted to the descriptions in the text of Linear Temporal Logic (LTL) which is a widely used text form in model checking. The structure Kripke such as $M \mid / = \neg f$ is applied to the first type of constraints, wherein M describes the system state, f is a formula consisting of logic connectors like &&, ‖, !, and so

forth, and ¬ means a negative logic. For the second type of constraints, the form $p = f_1 \wedge f_2$ is adopted, wherein f_1 describes the first potential deviation, and f_2 describes the second one, and so on. For example, equation "output[4] + ouput[9] == 0" to be involved in case study indicates that both outputs of Functions 4 and 9 are zero, which means accident may occur if either Function 4 or 9 is deviate. Furthermore, safety constraints can be described in mathematical ways and algorithms in model checking, provided their values assigned in terms of corresponding parameters.

Step 3 (simulate and analyze results). Simulation is conducted based on the established model, and the simulation result can be explained in accordance with the following principles.

(a) It is firstly checked whether the model is established correctly, if the simulation result does provide counter example(s) which means the safety constraints are violated under certain conditions. Provided that it is a correct model, the counter example(s) evidently indicates that the combinations of functional states do not satisfy the system safety requirements. After the model has been justified to be established correctly and rationally, the practical significance underlying the counter examples needs to be analyzed.

(b) Considering the practical significance of the counter examples given in the result of simulation, measures should be developed to eliminate hazards or to dampen their evolution in the system. The parameters in the model can be reset according to the measures being developed, and even the model structure can be updated, to check the benefit and effectiveness of the measures.

4. Case Study

Taking an air accident as a case, the approach proposed in this paper is used to analyze why and how the accident may occur, and the comparison is made between the conclusions drawn with this approach and those from the official investigation report, to illustrate the merit of this approach.

(1) The Accident Process. On 8 October 2001, an aircraft crashed at the Linate Airport in Milan, Italy. Scandinavian Airlines Flight 686 carrying 110 people collided with a Cessna Citation CJ2 business jet carrying four people. All 114 people on both aircrafts were killed, as well as four people on the ground. The disaster is the deadliest air disaster in Italian aviation history [34]. On the day of the accident, the visibility at the airport is only 50–100 meters due to heavy fog. Flight 686 was allowed to taxi to the runway R6 on 07:54, while the business jet was allowed to taxi to the runway R5 on 8:05. When having parked on taxiway S4, the crew on the business jet reported to the air traffic controllers. It was unnoticed that business jet accidentally broke into runway R6 along the indicator lights and ground markings. Flight 686 was allowed to take off on 8:09, but it crashed with the business jet stopping on the same runway when it took off.

Table 4: Descriptions for Activity 7 based on FRAM.

Aspect	Details
Input	Order to taxi to the main runway
Output	Enter R6 runway
Precondition	Available runway(s)
	Fulfill the ATC's taxi instructions
	Complete taxi checklists
Resource	Flight crew who is familiar with the airport
	Signal lights and ground markings on the airport
Control	Alarm system for preventing airplanes from breaking into the wrong runway
Time	The whole process

(2) Function Modules. The plane had to take off with the aid of the instructions given by the Air Traffic Control (ATC) due to the poor visibility at the airport. The processes can be broken down specifically into the following 11 activities/events: (1) the air traffic controller guided flight 686 to R6 runway; (2) Flight 686 taxied to R6 runway; (3) ATC guided business jet to R5 runway; (4) business jet reported to ATC in the S4 taxiway position; (5) ATC confirmed business jet position; (6) ATC guided business jet continue to slide to the main runway; (7) Business jet turned left into the R6 runway along signal lights and ground markings; (8) ATC guided the 686 flight takeoff; (9) Flight 686 took off; (10) alarm system prevented the aircraft break into the runway; (11) the ground radar monitored the positions of the aircrafts and vehicles on the airport. The latter two are involved with safety control and monitoring system. Based on FRAM, the eleven activities/events can be treated as function modules of the accident, with specific descriptions for each of them. The details for the instance of Activity 7 are shown in Table 4, and the standard conduction of the activities/events and interactions among the flight 686, the business jet, the ATC, and the monitoring system are shown in Figure 4.

(3) Potential Variability. According to the rules given in Table 1, the potential conduction variability of each activity/event is analyzed based on the features of the activity/event, and the potential variability of Activity 7 is shown in Table 5 as an example.

(4) Modeling and Simulation with Model Checking

(a) Modeling of System Behaviors. The tool Process Analysis Toolkit (PAT), which is a self-contained framework to support composing, simulating, and analyzing dynamic systems [35], was used here to model and verify the double-plane system (including the planes, the crew aboard, the ATC, and the airport situations) involved in this air accident. For simplicity, it is assumed that the upstream Outputs are consistent with the relevant downstream aspects like Inputs or Preconditions, and the effects of only the Input and

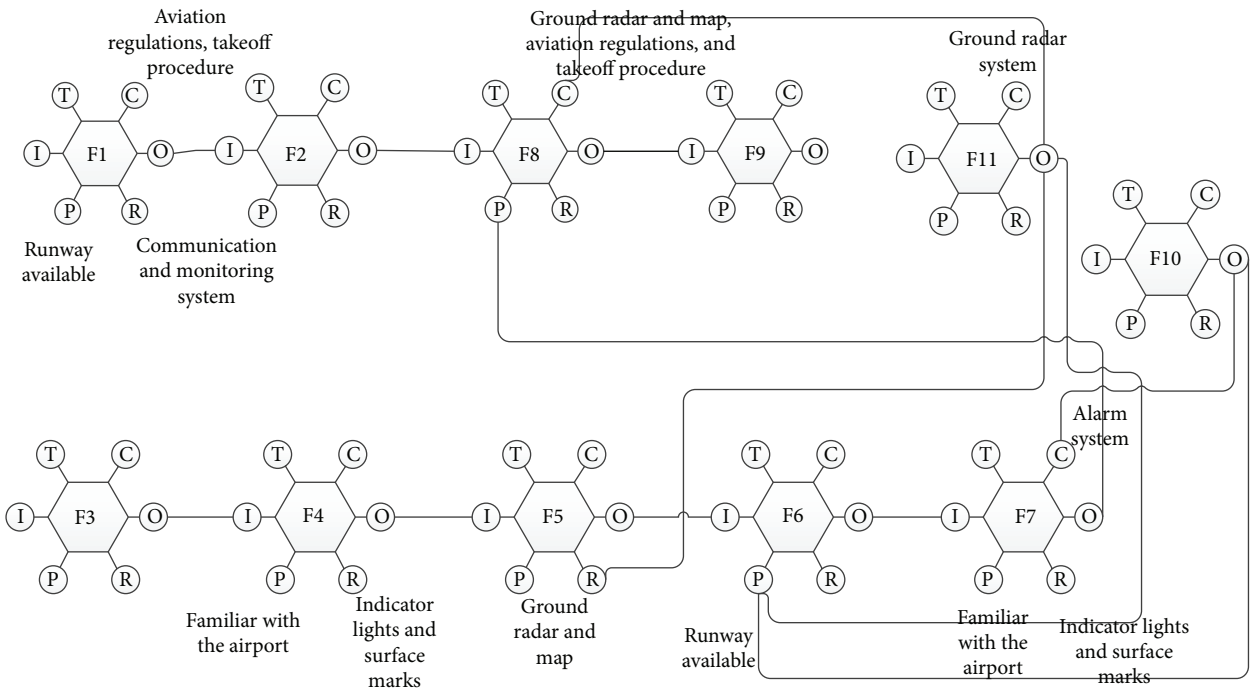

Figure 4: The instantiation of the functional interactions.

Table 5: Potential variability of Activity 7.

Aspects	Variability	Influence on Output
Input	The order by ATC is imprecise	The plane takes the wrong runway
	The order is earlier	The plane takes the wrong runway earlier
	The order is later	The plane takes the wrong runway later
Precondition	The runway is not available	The plane shares the runway with other aircrafts
	The airplane does not fulfill ATC's taxi instructions	The plane takes the wrong runway
Resource	The crew members are not familiar with the airport situations	The plane takes the wrong runway
	The signal lights and the ground markings on airport are misleading	The plane takes the wrong runway
Control	Alarm system does not work properly	The plane is not prevented from breaking into the wrong runway
Time	Too short	The plane takes the wrong runway earlier
	Too long	The plane takes the wrong runway later

Resource on the Output are considered for each function. For the purpose of coding with model checking, the system is described with functions of the two planes, respectively (as shown in Table 6), taking into account the descriptions for the activities identified earlier.

The value for the six aspects of each function is defined as 0 when they are normal and as 1 when there is variability existing. For example, input[1] = 0 indicates that Function 1 has normal Input, and input[1] = 1 means a variant Input to Function 1 from its upstream functions. The similar case applies to res[1] = 0 or output[1] = 0. For the four functions of the flight 686, the upstream Output is equivalent to the downstream Input or some other aspects, which is expressed as follows:

$$\text{input}\,[i+1] = \text{output}\,[i]\,. \quad (5)$$

For the functions performed by ATC, for example, Functions 1, 3, 5, 7, and 9, it is the Resources that have a great influence on the Output. The Output is normal if the Resource and Input are both normal, but there is variability in the Output with the variability in the Resource. The logic correlation is expressed as follows:

$$\text{output}\,[i] = 1 * \text{res}\,[i] + 0 * \text{input}\,[i]\,, \quad \text{if } i\%2 == 1. \quad (6)$$

For Functions 2 and 4, it is the Inputs that have a greater influence on the Output. The Output is incorrect if the Input is wrong; otherwise, the Output is correct if the Input is correct. The logic is expressed as follows:

$$\text{output}\,[i] = 0 * \text{res}\,[i] + 1 * \text{input}\,[i]\,, \quad \text{if } i\%2 == 0. \quad (7)$$

Table 6: The functions of the flight 686 and the business jet.

Airplane	Number	Function
Flight 686	Function 1	The air traffic controller guided flight 686 to R6 runway
	Function 2	Flight 686 taxied to R6 runway
	Function 3	ATC guided the 686 flight take off
	Function 4	Flight 686 took off
Business jet	Function 5	ATC guided business jet to R5 runway
	Function 6	Business jet reported to ATC in the S4 taxiway position
	Function 7	ATC confirmed business jet position
	Function 8	ATC guided business jet continue to slide to the main runway
	Function 9	Business jet turned left into the R6 runway

Similarly, for the six functions of the business jet, the upstream Output is taken equivalent to the downstream Input, which is expressed as follows:

$$\text{input}\left[j+1\right] = \text{output}\left[j\right]. \tag{8}$$

Finally, considering the functions conducted in parallel by the two airplanes, the system state is expressed as follows:

$$\text{System} = \text{flight}() \,|||\, \text{jet}(). \tag{9}$$

(b) Safety Constraints Description. In this case, the accident happened due to the fact that a runway is occupied by the two planes simultaneously, so the safety constraint is described as that no more than one airplane is permitted to be on a runway at a time. According to the parameters' meaning given earlier, output[4] = 0 means the flight 686 is on R6 runway and output[9] = 0 means the business jet is on R5 runway, when there is no function variability, while output[9] = 1 means the business shares R6 runway with the flight 686. Apparently R6 runway cannot be used by two planes at the same time according to air traffic rules; that is, the constraint is interpreted as both the statements output[4] = 0 and output[9] = 0 are true, which is expressed as (10) to indicate that the flight and the business jet take different runways

$$\text{output}\,[4] + \text{output}\,[9] = 0. \tag{10}$$

(c) Simulation and Result Analysis. Given the initial values of parameters preset randomly, how the parameters change is observed for all the potential states by means of model checking. The simulation result shows that there are two scenarios in which the runway may be occupied by the two planes at the same time. The first scenario is exactly the accident process that has truly occurred (and been reported by ANSV) while the second one is a potential accident process which may occur, despite the fact that it has never occurred so far.

Scenario 1. Due to ATC's unclear instructions as well as the misleading ground markings and flight indicator, the business jet entered the wrong taxi way. Moreover, when it even arrived at S4 position, ATC did not correct the direction of the flight because of the map without being updated. Due to the failures of ground radar and alarm device, the business jet was not prevented from breaking into R6 runway. Besides, in spite of the failure to scan the planes' positions with the ground radar, ATC guided flight 686 to take off, which eventually led to the accident. This scenario was explained in ANSV report [34]: the accident was caused basically by the combination of inaccurate order of ATC, the wrong guide lights and ground markings, unfamiliarity with the airport, the map without updated, and the failure of ground radar and alarm system.

Scenario 2. Even if the ATC gives right instructions in some time, the business jet might enter the wrong taxiway due to the error of the markings and flight indicators. When the business jet arrives at S4 position, ATC may not correct the flight direction because of the map without being updated. Furthermore, since the alarm device and ground radar fail to work, the two planes might possibly enter the same runway. This is a potential scenario which could also lead to the accident, and the slight difference from Scenario 1 is whether Function 5 "ATC guided business jet to R5 runway" has normal Output or not. In Scenario 2, all the functions of the planes and ATC are normal before the business jet taxies, but the business jet parks at the wrong location because the crew is unfamiliar with the airport, in addition to the misleading guide lights and ground markings at the airport. When the crew reports to ATC their exact location, ATC does not perceive the deviation of its position due to the fact that neither the report map has been updated nor the ground radar works, so ATC guides the plane to continue taxiing to the main runway R6 instead of the R5 runway that ATC wants. It indicates that the ATC function's Output may deviate from what is desired due to the erroneous Resource, even if the conduction was in accordance with the relevant provisions. Scenario 2 is depicted as in Figure 5 with the abnormal paths being marked as red, to demonstrate how the deviations spread among functions.

5. Conclusions and Future Work

This paper extends FRAM by integrating it with model checking to effectively explore hazard evolution. The extended FRAM refines the understanding of interactions among functions of sociotechnical systems by redefining and categorizing the couplings. It also proposes a process, by means of auto search with a computer tool, for identifying functional deviations as well as their propagation among upstream and downstream functions. The approach is a progress in efforts for the exhaustiveness of heuristic analysis, which in the past depended excessively on the knowledge and experience of analysts, and was hard to traverse all the possible conditions due to limitations of human's recognition. While the exhaustive search across all the scenarios for the one(s) that may lead to accident is conditionally conducted, that is, the search is based on the functions and their variability pre-determined heuristically, more potential couplings among functions can

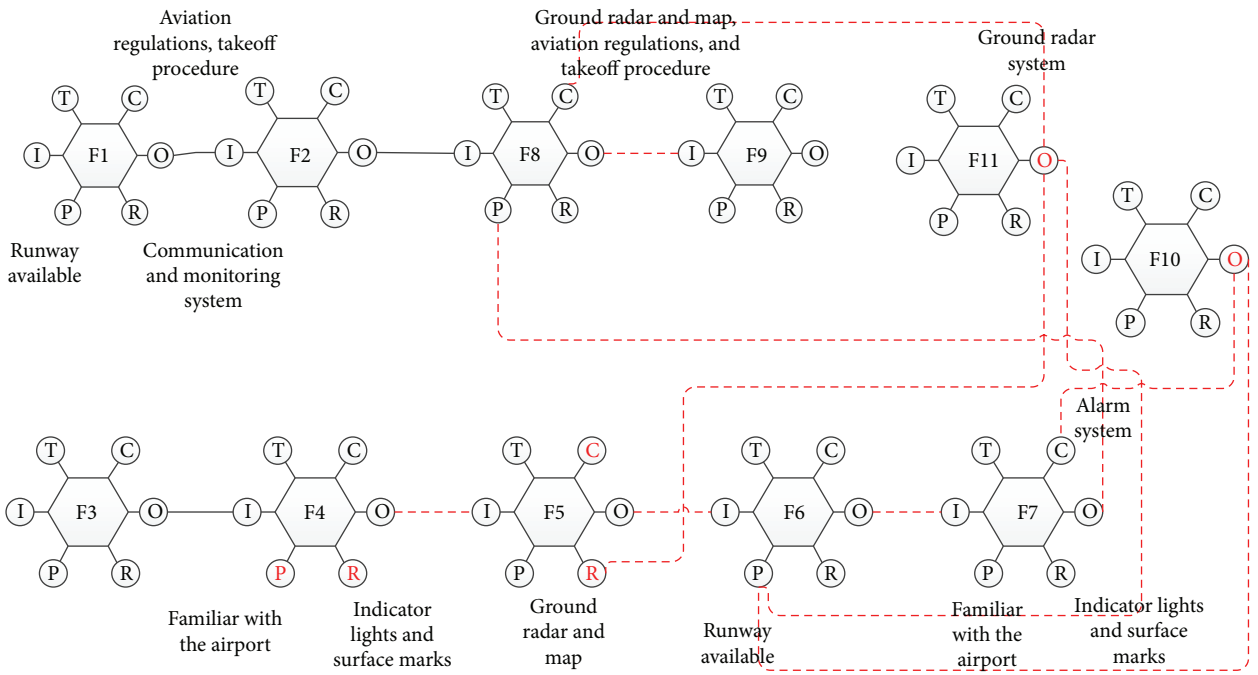

Figure 5: Scenario 2 for the accident.

be identified by relatively rigorous derivation rather than by subjective analysis.

Taking a typical air accident as the case, the plausibility of the approach is illustrated. Additionally, in point of the accident scenarios as such, apparently it is highlighted that an accident was not caused by a simple combination of multiple contributing factors, but by performance variability and its nonlinear propagation among functions. Hence, in order to prevent accidents, it is far more significant to coordinate the functions in the system than put the emphasis on a certain single aspect. For example, to continue the case of the air accident, it is definitely true that the mentoring system should be open, that the map should be updated, and that the indicating lights and ground markings should work regularly. But more efforts need to be concentrated on how to make all of these aspects cooperate and function well simultaneously and continuously, without any aggravation even though an unexpected disturbance happens somewhere and sometime.

It is noted that in this paper we focus more on whether, rather than how, functions vary (in the part of quantitative analysis), as actually the terms of variability in different ways have essential influence on the rationality of a functional model, for the reason that different variability terms of an upstream function may impact its downstream function in different ways. Accordingly, the spreading rules of variability need to be elaborated further based on the specific term of functional variability. In the future work, the approach of functional behavior modeling will be improved and specified based on FRAM, with more deliberation of variability of functions, as well as further development of the rules that describes interactions among functions in the sense of abstraction. Besides, there are some assumptions made to simplify the analysis in the case study; for example, the Time aspect and details of the airplanes' behaviors have not been considered. For the lack of practical illustration of some aspects (e.g., Time and Control), there are not sufficient analysis and discussion with regard to them in functional modeling. Thus, efforts will also be put into case study; that is, the case will be analyzed in detail, with more consideration supplemented involving the aspect of Time, airplanes' states like position and direction, and so on.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by Grants from the National Natural Science Foundation of China (no. NSFC-61403009), from a project of Ministry of Industry and Information Technology of China (no. JSZL2014601B004), and from the Major State Basic Research Development Program of China (973 Program) (no. 2014CB744904). This support is gratefully acknowledged.

References

[1] F. I. Khan and S. A. Abbasi, "Studies on the probabilities and likely impacts of chains of accident (domino effect) in a fertilizer industry," *Process Safety Progress*, vol. 19, no. 1, pp. 40–56, 2000.

[2] C. D. Lorenz and R. M. Ziff, "Precise determination of the critical percolation threshold for the three-dimensional "Swiss cheese" model using a growth algorithm," *The Journal of Chemical Physics*, vol. 114, no. 8, article 3659, 2001.

[3] E. Hollnage, "Barriers and accident prevention," *Ergonomics*, vol. 50, no. 6, pp. 961–962, 2007.

[4] N. G. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*, The MIT Press, London, UK, 2011.

[5] E. Hollnagel, *FRAM: The Functional Resonance Analysis Method, Modelling Complex Socio-Technical Systems*, MPG, London, UK, 2012.

[6] M. C. Browne, E. M. Clarke, and O. Grumberg, "Reasoning about networks with many identical finite state processes," *Information and Computation*, vol. 81, no. 1, pp. 13–31, 1989.

[7] E. Hollnagel, *Barriers and Accident Prevention*, Ashgate, Farnham, UK, 2004.

[8] E. Hollnagel, "Critical information infrastructures: should models represent structures or functions?" in *Computer Safety, Reliability, and Security*, vol. 5219 of *Lecture Notes in Computer Science*, pp. 1–4, Springer, Berlin, Germany, 2008.

[9] I. A. Herrera and R. Woltjer, "Comparing a multi-linear (STEP) and systemic (FRAM) method for accident analysis," *Reliability Engineering & System Safety*, vol. 95, no. 12, pp. 1269–1275, 2010.

[10] J. Hovden, E. Albrechtsen, and I. A. Herrera, "Is there a need for new theories, models and approaches to occupational accident prevention?" *Safety Science*, vol. 48, no. 8, pp. 950–956, 2010.

[11] R. Woltjer, *A Systemic Functional Resonance Analysis of the Alaska Airlines Flight 261 Accident*, Human Fators and Economic Aspects on Safety, 2006.

[12] H. Sybert, H. C. Stroeve Mariken, and A. P. Henk, "Studying hazards for resilience modelling," in *Proceedings of the 1st SESAR Innovation Days*, pp. 1–8, Toulouse, France, November-December 2011.

[13] P. V. R. de Carvalho, "The use of Functional Resonance Analysis Method (FRAM) in a mid-air collision to understand some characteristics of the air traffic management system resilience," *Reliability Engineering & System Safety*, vol. 96, no. 11, pp. 1482–1498, 2011.

[14] E. Hollnagel, S. Pruchnicki, R. Woltjer, and S. Etcher, "Analysis of Comair flight 5191 with the functional resonance accident model," in *Proceedings of the 8th International Symposium of the Australian Aviation Psychology Association*, Sydney, Australia, 2008.

[15] T. Sawaragi, Y. Horiguchi, and A. Hina, "Safety analysis of systemic accidents triggered by performance deviation," in *Proceedings of the SICE-ICASE International Joint Conference*, pp. 1778–1781, Busan, South Korea, October 2006.

[16] F. Belmonte, W. Schön, L. Heurley, and R. Capel, "Interdisciplinary safety analysis of complex socio-technological systems based on the functional resonance accident model: an application to railway traffic supervision," *Reliability Engineering & System Safety*, vol. 96, no. 2, pp. 237–249, 2011.

[17] L. Macchi, P. Oedewald, M. H. Rø Eitrheim, and C. Axelsson, "Understanding maintenance activities in a macrocognitive work system," in *Proceedings of the 30th European Conference on Cognitive Ergonomics (ECCE '12)*, pp. 52–57, Edinburgh, Scotland, August 2012.

[18] J. Bentahar, M. El-Menshawy, H. Qu, and R. Dssouli, "Communicative commitments: model checking and complexity analysis," *Knowledge-Based Systems*, vol. 35, pp. 21–34, 2012.

[19] F. Wang, "Efficient model-checking of dense-time systems with time-convexity analysis," *Theoretical Computer Science*, vol. 467, pp. 89–108, 2013.

[20] V. Chapurlat, "UPSL-SE: a model verification framework for systems engineering," *Computers in Industry*, vol. 64, no. 5, pp. 581–597, 2013.

[21] C. Tian, Z. Duan, and N. Zhang, "An efficient approach for abstraction-refinement in model checking," *Theoretical Computer Science*, vol. 461, pp. 76–85, 2012.

[22] R. Gómez, "Model-checking timed automata with deadlines with Uppaal," *Formal Aspects of Computing*, vol. 25, no. 2, pp. 289–318, 2013.

[23] E. Onem, A. B. Gürdağ, and M. U. Çağlayan, "Formal security analysis of ariadne secure routing protocol using model checking," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 9, no. 1, pp. 12–24, 2012.

[24] M. H. Ter Beek, A. Fantechi, S. Gnesi, and F. Mazzanti, "A state/event-based model-checking approach for the analysis of abstract system properties," *Science of Computer Programming*, vol. 76, no. 2, pp. 119–135, 2011.

[25] S. Konur, C. Dixon, and M. Fisher, "Analysing robot swarm behaviour via probabilistic model checking," *Robotics and Autonomous Systems*, vol. 60, no. 2, pp. 199–213, 2012.

[26] M. Gribaudo, A. Horváth, A. Bobbio, E. Tronci, E. Ciancamerla, and M. Minichino, "Fluid petri nets and hybrid model-checking: a comparative case study," *Reliability Engineering and System Safety*, vol. 81, no. 3, pp. 239–257, 2003.

[27] Y. Gao, M. Xu, N. Zhan, and L. Zhang, "Model checking conditional CSL for continuous-time Markov chains," *Information Processing Letters*, vol. 113, no. 1-2, pp. 44–50, 2013.

[28] J. Rushby, "Using model checking to help discover mode confusions and other automation surprises," *Reliability Engineering & System Safety*, vol. 75, no. 2, pp. 167–177, 2002.

[29] S. Zhang, J. Teizer, J.-K. Lee, C. M. Eastman, and M. Venugopal, "Building Information Modeling (BIM) and Safety: automatic safety checking of construction models and schedules," *Automation in Construction*, vol. 29, pp. 183–195, 2013.

[30] J. Lahtinen, J. Valkonen, K. Björkman, J. Frits, I. Niemelä, and K. Heljanko, "Model checking of safety-critical software in the nuclear engineering domain," *Reliability Engineering & System Safety*, vol. 105, pp. 104–113, 2012.

[31] Department of Defense Standard Practice for System Safety, MIL-STD-882D, 1993.

[32] C. Zhao, M. Bhushan, and V. Venkatasubramanian, "Phasuite: an automated HAZOP analysis tool for chemical processes: part I: knowledge engineering framework," *Process Safety and Environmental Protection*, vol. 83, no. 6, pp. 509–532, 2005.

[33] S. Dowlatshahi, "The role of product safety and liability in concurrent engineering," *Computers & Industrial Engineering*, vol. 41, no. 2, pp. 187–209, 2001.

[34] Agenzia Nazionale per la Sicurezza del Volo (ANSV), *Accident Involved Aircraft Boeing MD-87, Registration SE-DMA and Cessna 525-A, Registration D-IEVX*, Milano Linate Airport, 2001.

[35] J. Zhang, Y. Liu, J. Sun, J. S. Dong, and J. Sun, "Model checking software architecture design," in *Proceedings of the IEEE 14th International Symposium on High-Assurance Systems Engineering (HASE '12)*, pp. 193–200, Omaha, Neb, USA, October 2012.