Research Journal of Applied Sciences, Engineering and Technology 16(3): 104-111, 2019
DOI:10.19026/rjaset.16.6006
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: December 13, 2018 Accepted: February 12, 2019 Published: May 15, 2019

Research Article

SysML-Based Integration of System Design and Failure Models and Safety Verification by Simulation

Chang-Won Kim and Jae Lee
Department of Systems Engineering, Ajou University, Suwon, Republic of Korea, Korea

Abstract: The present study aims to develop an improved SysML-based integration model that can be used to perform system design and failure analysis simultaneously and verify safety activities. In recent studies, the safety of a system has been evaluated by modeling the system design and failure analysis. However, because the models developed in there were created using different modeling languages, it was not easy to carry out system design and safety activities efficiently. Furthermore, studies using UML or SysML-based failure models for deriving safety requirements have shown that these models have limited applicability to safety analysis and verification. To solve this problem, we propose to explore an advanced method for failure modeling and verification. First, an improved SysML-based integration model was developed, which can combine system design and safety verification activities interactively. Next, we transformed the integration model for analysis into a simulation model for verification with the safety measures derived from the failure model. A case study of the safety design for an automotive system was then followed with the analysis model and simulation results to verify the safety of the automotive system. Through the case study, the concept of safety design and verification became more explicit and the proposed method proved to be useful.

Keywords: Failure model, modeling and simulation, system safety, Systems Modeling Language (SysML), safety verification

INTRODUCTION

Accidents related to safety-critical systems can lead to loss of life and enormous damage to property. International safety standards have been established to ensure the safety of such systems. One of the issues related to safety design is on how to incorporate safety requirement in system design activities. The integration of system design and safety activities was mentioned without specific methods in the representative safety standards such as MIL-STD-882E, IEC 61508 and ISO 26262 (MIL-STD-882E, 2012; IEC 61508, 2010; ISO 26262, 2011). As such, a model-based approach has been tried to integrate system design and safety activities. The motivations for adopting the approach are as the following:

- Model-Based Systems Engineering (MBSE) can improve the consistency of system design (Paredis, 2008)
- The system design process can improve communication between engineers
- The system performance and constraints can be verified via model simulation, thus enabling system optimization. Therefore, this approach can be utilized to ensure the safety of critical systems (Friedenthal *et al*., 2014; Mhenni *et al*., 2016).

Early studies on the integration of system design and safety activities have proposed a meta-model for communication error reduction between system engineers and safety engineers and for consistent system design support (Beckers *et al*., 2017; Deleuze *et al*., 2014; Helle, 2012; Piriou *et al*., 2014; Hillenbrand *et al*., 2012). However, because the capability of the meta-model is limited to presenting a process or framework for integration of system design and safety activities, a detailed system design methodology that practically incorporates safety activities has not been presented. For this reason, recent studies have focused on the derivation of system safety requirements and on the improvement of safety verification using a failure model that accurately reflects the safety analysis results of the target system in the system architecture.

To achieve integration, a Model-Based Safety Analysis (MBSA) is employed to generate a failure model that accurately incorporates the failure information derived via safety analysis into the system

Corresponding Author: Jae Lee, Department of Systems Engineering, Ajou University, Suwon, Republic of Korea, Korea

architecture and to modify the failure model to make it suitable for safety verification (Sharvia and Papadopoulos, 2015; Jaradat, 2012; Wei *et al*., 2017; Chen *et al*., 2017; Zhao *et al*., 2016; Duan *et al*., 2015; Mehrpouyan, 2011). However, modifying the failure model to facilitate safety verification is a time-consuming process. Thus, to solve this problem, it is necessary to perform safety verification using a failure model that appropriately incorporates failure information into the system architecture. Safety requirements have been derived based on the failure information represented in the failure model (Joshi and Heimdahl, 2007; Papadopoulos *et al*., 2001; Mauborgne *et al*., 2016; Guiochet, 2016). However, it is difficult to perform safety analysis or verification using this methodology because it focuses on using failure models to derive safety requirements.

To overcome the problems mentioned above and extend the existing failure model, we developed and verified an improved SysML-based failure model that can be used to perform system design and safety verification activities.

LITERATURE REVIEW

Safety analysis and failure models: MBSA studies present various failure models. References (Joshi and Heimdahl, 2007; Papadopoulos *et al*., 2001) proposed physical architecture-based failure models that focused on possible faults occurring in the physical components of a system. Alternatively, reference (Sharvia and Papadopoulos, 2015) proposed a failure model that incorporates the potential failures of system elements and the effects of the failure of the system elements into the system model. Consequently, the failure model can incorporate accurate failure information into the functional and physical architecture of the target system.

With MBSA, the failure model is typically expressed in formal specification notation using the language of the model checking tool. Thereafter, model checking of the failure model is performed to derive counter-examples that violate safety requirements. Subsequently, system safety is designed with the help of the modified failure model based on the derived counter-examples (Jaradat, 2012).

A failure model that can be used to verify the safety of the system was developed based on the results of Failure Mode and Effect Analysis (FMEA) for the target system; moreover, model checking was performed to verify the safety of the system design (Wei *et al*., 2017). SPIN, the language of the model checking tool, was used in this study to identify possible failure combinations and accidents paths by deriving all counter-examples that violate safety requirements.

Reference (Zhao *et al*., 2016) focused on failure model implementation for safety analysis. A failure model was generated using the system interface model; a minimal cut set was calculated using a failure model to perform the safety analysis automatically. However, because the studies mentioned above did not generate a failure model that implements languages supported by model checking tools, they were unable to perform safety analysis and verification. Moreover, because the method of model development implemented in these studies does not facilitate utilization of the modeling language that was used to create the system model, modifying these types of models requires a substantial amount of time and effort. Therefore, it is necessary to perform safety verification using a model that incorporates relevant failure information into the architecture of an existing system.

Safety requirement derivation: Previous studies have used the failure model for safety requirements derivation and safety verification. As an example, in the study that proposed the methodology for hierarchically performed hazard origin and propagation studies, Fault Tree Analysis (FTA) was performed by utilizing the information on faults expressed in the failure model. The results of the FTA were subsequently used in FMEA to derive safety requirements (Papadopoulos *et al*., 2001).

In the automotive industry, the failure model adopting the systems modeling language (SysML), a defacto standard, has been used to derive safety goals and corresponding safety requirements (Mauborgne *et al*., 2016). A dysfunctional scenario model was developed by considering the hazards of an automobile system when developing the nominal scenario model of the system. Subsequently, the dysfunctional scenario model was used for deriving safety goals. The safety goals were then incorporated into the dysfunctional scenario model. As a result, an avoidance scenario model that incorporates the safety goals into the dysfunctional scenario model was created. In the study that derived the safety requirements using the Unified Modeling Language (UML), the behavior model of the system was represented as a UML sequence diagram and state machine diagram. Guiochet derived safety requirements by carrying out a hazard and operability study using this model and a particular guideword (Guiochet, 2016).

In the studies mentioned in this section, failure and system models have been used to derive safety requirements. However, it should be noted that, although failure models are frequently used to derive safety requirements, the ability of the failure models in facilitating safety analysis or safety verification is limited.

Research objective: Existing failure models have been used to perform safety analysis and verification, or to derive safety requirements. However, safety analysis and verification performed using the failure models

turned out to be time-consuming and computationally expensive because the failure model needs to be changed for compatibility with the model checking tool. To reduce the costs, a method of safety verification that does not require modifying the failure model must be developed. Alternatively, studies that derived safety requirements based on the failure model demonstrated that there was no need to change the failure model to derive safety requirements; however, these studies were limited regarding their ability to derive specific safety requirements. Therefore, it is necessary to extend the capabilities of existing failure models. For this reason, in this study, an improved SysML-based failure model was created. The proposed model was validated via simulation and subsequent the safety design of the system was evaluated.

MATERIALS AND METHODS

The improved SysML-based failure model proposed in this study is a model that reflects information about failures in the system architecture. A failure means that the function assigned to a component of the system does not function as intended; thus, a failure can be reflected in the system architecture model by accounting for the malfunction in the functional architecture of the system. Since SysML is a language for modeling system functions and physical architecture, it is appropriate to use SysML to incorporate the failure information into the system architecture as an element of the system model. Therefore, in this study, we use SysML to create a failure model that accurately reflects the necessary failure information in the system model.

How to create SysML-based system architecture model: System failure information is derived by performing functional FMEA using system design information obtained via the system architecture. The failure information must be derived via functional FMEA and mapped to the SysML model element as input. A failure model that reflects the FMEA results in the system architecture is developed using this methodology. Figure 1 shows the procedure for developing the proposed SysML-based failure model.

As shown in Fig. 1, the first step in failure model development involves modeling the system architecture under normal conditions in which no failure exists; this means that there is no failure information present within the system. Since the detailed physical structure of a system cannot be accurately modeled at the initial stages of system design, the functional architecture of a system should be designed and system behavior should be analyzed. To analyze the system behavior, an activity diagram, such as that shown in Fig. 2, was designed and implemented because it can express the

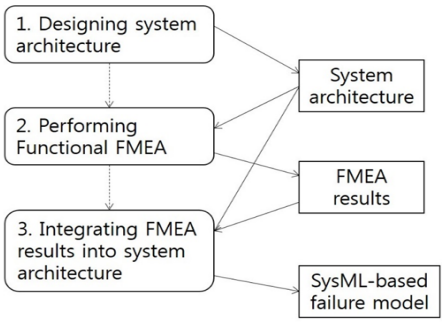

Fig. 1: General procedure for SysML-based failure model development

data exchange and functional flow between the functions of the system. The functional architecture, shown in Fig. 2, does not consider safety measures to prevent failure. Thus, it is only capable of analyzing how the system behaves in a normal state.

The second step in failure model development involves using functional architecture, as shown in Fig. 2, in functional FMEA to derive failure and failure effects at the initial stage of system design. Because functional FMEA analyzes the failure modes concerning the functions of the target system and evaluates their effects, the results provide failure information on the failure mode of each system function, its effect on the system and safety measures that should be taken. Therefore, by integrating such failure information into the system model, a SysML-based failure model can be generated. Moreover, the safety measures derived via functional FMEA can be reflected in the functional architecture, which can be simulated to verify that the safety measures are sufficiently useful.

How to utilize failure information: Since the output of the functional FMEA exists in the form of worksheets, it is difficult to input the results into the system model directly. As such, it is necessary to map the FMEA results to the model elements of the SysML state machine diagram. The system model refers to a model of the normal state of the system before the safety analysis. However, the failure model should show system behavior under abnormal conditions, i.e., the occurrence of a failure that does not affect the entire system, by incorporating failure information into the normal state model of the target system. The state machine diagram illustrates and describes the state of a system, system behavior in each state and state transitions of the system. In other words, the state machine diagram yields information on the normal, abnormal and failure states of a system, in addition to the system behavior in each of these states. Additionally, the state machine diagram can characterize state transitions between normal and abnormal states and the transition from an abnormal

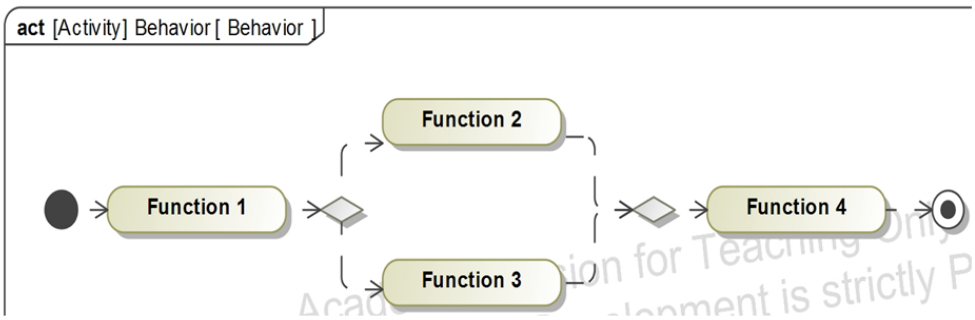

Fig. 2: Activity diagram: General system behavior model

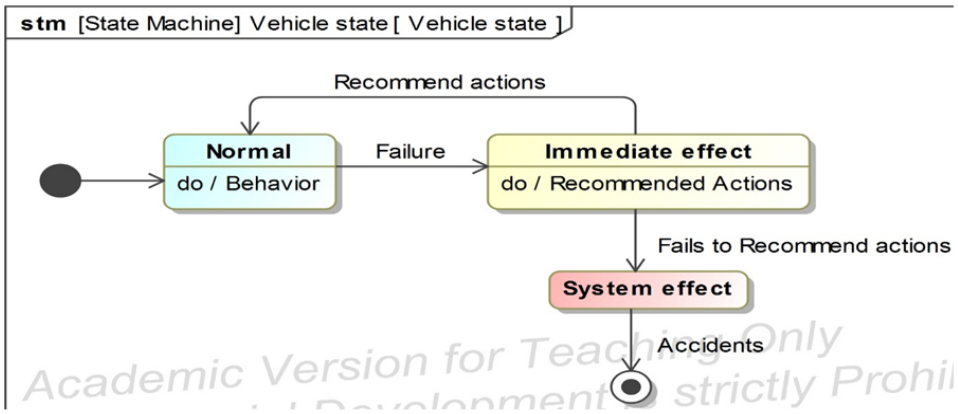

Fig. 3: Generalized SysML-based failure model

Table 1: Mapping functional FMEA elements to state machine diagram elements

Functional FMEA elements	State machine diagram elements
Function	Action node of Do activity
Failure mode	Transition path
Immediate effect	State
System effect	State
Recommended action	Action node of Do activity

state to a failure state. Table 1 describes how the elements of the FMEA are mapped to the model elements of the state machine diagram.

The last step in developing a SysML-based failure model is to integrate the failure mentioned above information into the system model. As listed in Table 1, the normal and abnormal states of a system are represented by a state node. The state node describes the failure state that has resulted from ineffective or insufficient safety measures that were performed during the abnormal state. Also, the proposed model can describe how the system behaves in each state (i.e., a normal state, an abnormal state and a failure state) by representing the functions of the system and the recommended actions (safety measures) as Do action nodes (Table 1). Finally, in the event of a failure, the transition path refers to the transition period beginning from the initiation of safety measures until the termination of safety measure. Figure 3 shows the generalized form of the proposed SysML-based failure model generated by performing the steps outlined in this section.

Case study of an automotive braking system: The aim of this study was not only to generate a SysML-based model that incorporates failure information into the system architecture but also to implement the generated SysML-based failure model for safety verification. To demonstrate that the SysML-based failure model can be used for safety verification, the proposed methodology was applied to an automotive braking system.

To create a failure model for an automotive braking system, the architecture of the target system must first be designed. Subsequently, a functional FMEA should be performed based on the generated system architecture to obtain failure information for the braking system. In this study, we performed functional FMEA for the *Control caliper*, which is a function that regulates braking force application to automobile wheels. Table 2 summarizes the FMEA results for this function.

Following the methodology proposed in this study, the mapping procedure, presented in Table 1, should be used to integrate the function/failure status information results, listed in Table 2, into the system architecture model. Figure 4 shows the failure model developed for an automotive braking system.

Table 2: Functional FMEA results of an automotive braking system

Function	Failure mode	Immediate effect	System effect	Recommended actions
Control caliper	Ineffective control	Unable to control caliper	Unable to decelerate automobile	1. Diagnose the cause of failure. 2. Incorporate an auxiliary caliper regulatory function.

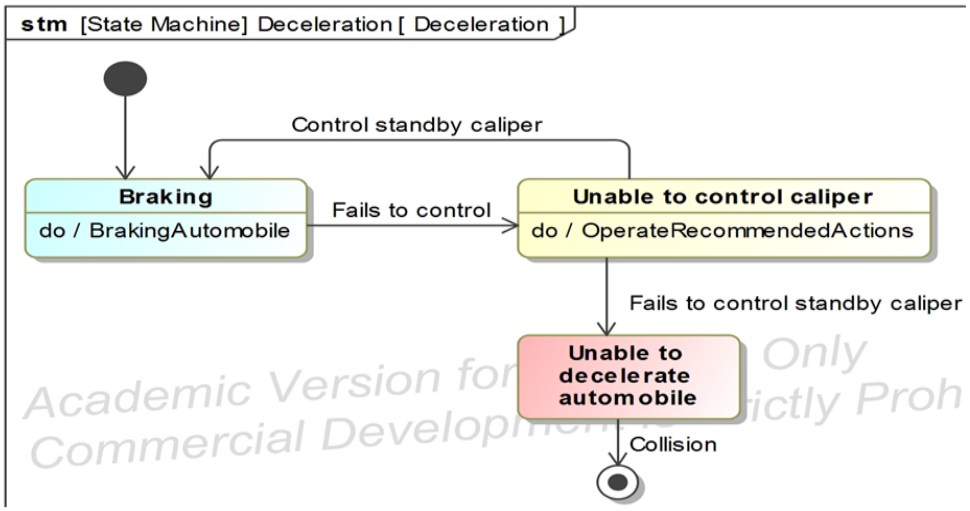

Fig. 4: SysML-based failure model of an automotive braking system

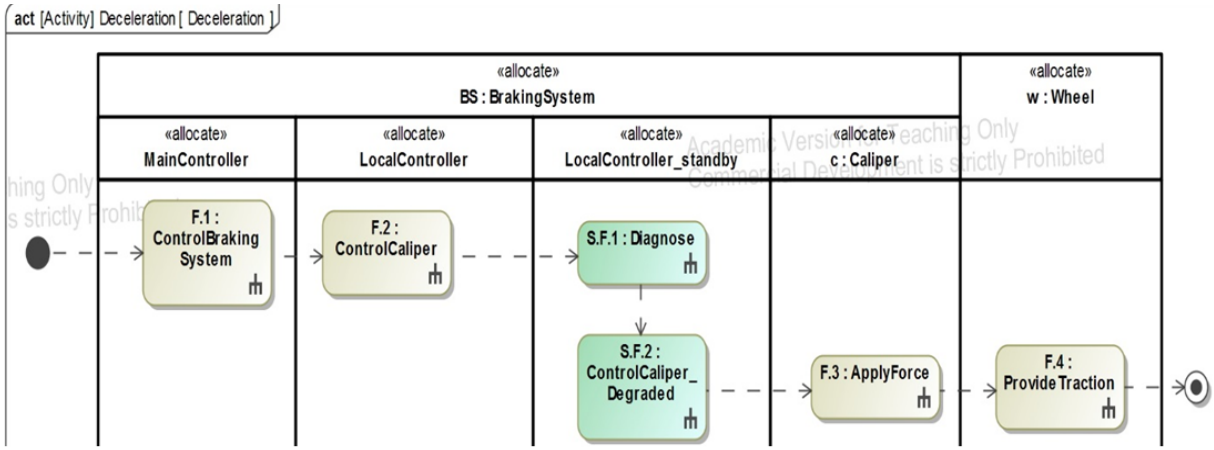

Fig. 5: Behavioral model of automotive braking system with safety measures

Verification of safety measures: The failure model in Fig. 4 shows the behavior of the automotive braking system in the normal state and the safety measures that would be performed when the system enters an abnormal state due to failure. Thus, the case study failure model, shown in Fig. 4, demonstrates that the proposed model can identify failures in an automotive braking system and incorporate safety measures into the system architecture to prevent system failure. Figure 5 shows the activity diagram of the functional architecture of the automotive braking system.

The behavioral simulation was performed to verify that the safety measures were appropriately incorporated into the functional architecture of the system. The functional architecture for normal-state and failure-state braking was simulated. The results are shown in Fig. 6 and 7.

RESULTS AND DISCUSSION

If the functional FMEA results, listed in Table 2, show that the *Control caliper* function has failed, the status of the function becomes *Unable to control caliper*. When safety measures are activated in this state, brake function returns to normal. However, if the *Unable to control caliper* state persists, the status of the function becomes *Unable to decelerate automobile*. To regulate the function in the abnormal state, we

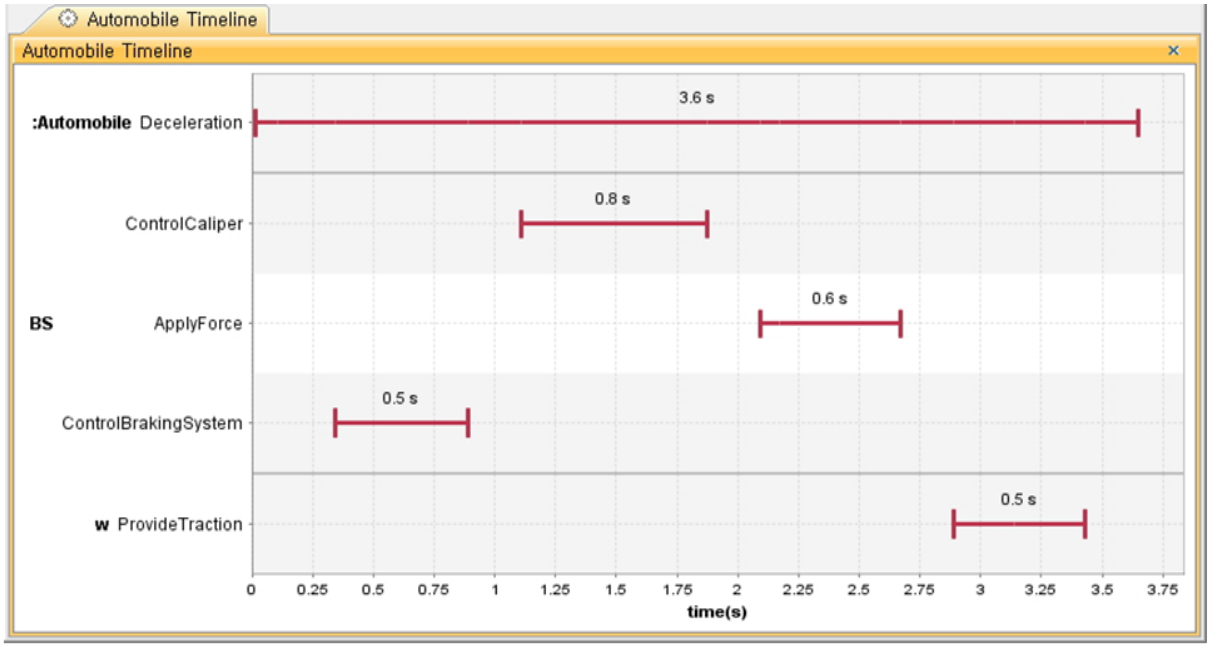

Fig. 6: Simulation of the behavior of the automotive braking system without safety measures

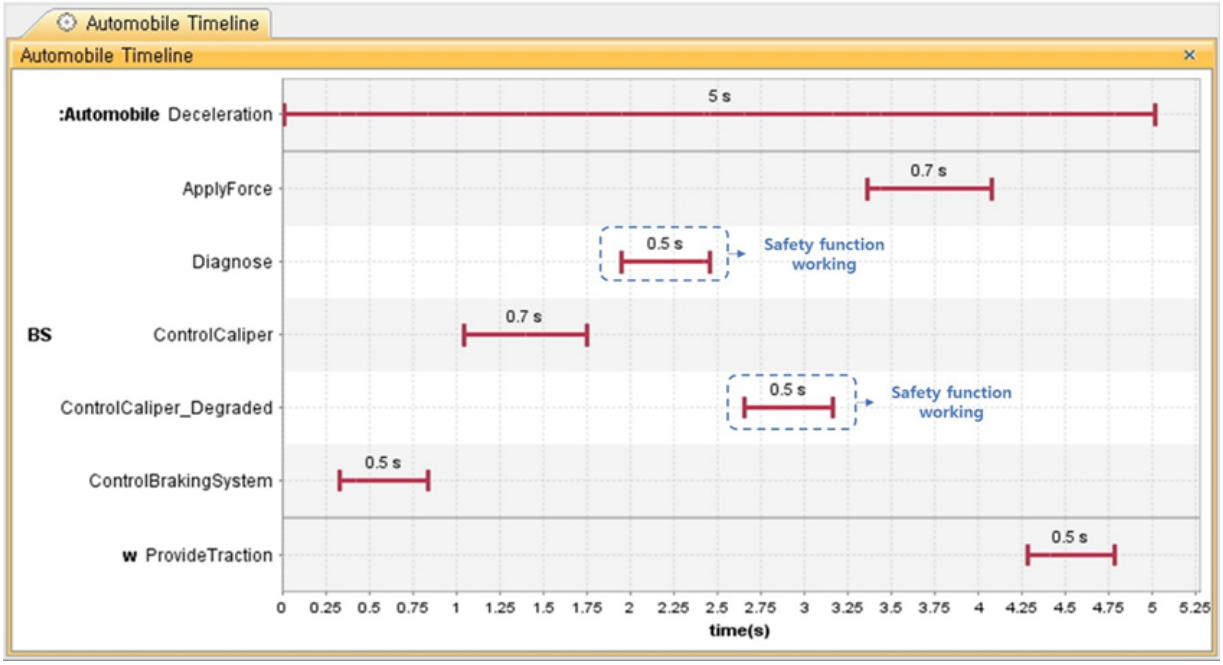

Fig. 7: Simulation of the behavior of the automotive braking system with safety measures

developed safety measures to diagnose the *Control caliper* function and to incorporate an auxiliary caliper regulatory function to prevent failure of the *Control caliper* function.

Figure 4 shows the actions performed in the normal braking state and the non-decelerating automobile state resulting from ineffective caliper control and safety measures. Furthermore, the action performed in each state is represented as a Do activity so that the type of initiated action can be confirmed when the system is in a normal state and an abnormal state. Additionally, state transitions are determined by the occurrence of failure and the effects of safety measure operations.

Figure 5 describes the actions and safety measures incorporated into the functions of an automotive braking system. The physical elements of the system to which each function is assigned are given in the swim lanes. The rectangles with rounded gray borders (F.1 to

F.4) are functions related to normal-state braking. Conversely, the rectangles with rounded blue borders (S.F.1 and S.F.2) are safety measures to prevent failure of the control caliper. This arrangement demonstrates how the system architecture has been modified to ensure that the braking system remains intact even if the control caliper malfunctions.

Figure 6 shows that when braking is initiated, the braking functions are performed in the following order: Control Braking System, Control Caliper, Apply Force and Provide Traction. Figure 7 shows the results of simulating the functional architecture for braking with safety measures. In this simulation, the Control Caliper function malfunctions and therefore, the corresponding safety measures are initiated. Consequently, the Control Braking System function is initiated; subsequently, the Control Caliper_Degraded function is initiated after the diagnose function detects the failure of Control Caliper. Next, Apply Force and Provide Traction functions are performed in order.

In summary, the results of the simulation as shown in Fig. 7, demonstrate that the proposed failure model for the automotive braking system can be used to verify the system safety design by determining whether the safety measures are appropriately integrated into the system architecture.

CONCLUSION

In this study, to overcome the problems with the existing failure models, we proposed an improved SysML-based integration model. First, a system model and a failure model were integrated to form a single model by generating a SysML-based failure model that incorporates failure information into the system architecture. Next, the behavior of the SysML-based failure model with the incorporated safety measures was incorporated into the simulation model and the safety design for the system architecture was verified using it. Since SysML-based failure model does not require constant switching between the system model and the failure model for model checking, it is more efficient and less time-consuming than existing failure models. Additionally, the proposed model allows the incorporation of safety requirements, which are functions that are not present in the existing failure models. In the future, the proposed SysML-based failure model should be further extended by employing it to perform the safety analysis of a system.

ACKNOWLEDGMENT

The authors wish to thank No Magic, Inc. for providing us with the academic site licenses for the M&S tool Cameo Systems Modeler®.

CONFLICT OF INTEREST

It should be noted that there is no financial support and there is no competitive interest in this area.

REFERENCES

Beckers, K., I. Côté, T. Frese, D. Hatebur and M. Heisel, 2017. A structured and systematic model-based development method for automotive systems, considering the OEM/supplier interface. Reliab. Eng. Syst. Safety, 158: 172-184.

Chen, L., J. Jiao, Q. Wei and T. Zhao, 2017. An improved formal failure analysis approach for safety-critical system based on MBSA. Eng. Failure Anal., 82: 713-725.

Deleuze, G., A. Leger, P.Y. Piriou and C. Sylvain, 2014. Interoperability between a dynamic reliability modeling and a systems engineering process: Principles and case study. Proceeding of the Embedded Real Time Software and Systems 2014 (ERTS²2014), Feb 2013, Toulouse, France, pp: 4B.2.

Duan, G., J. Tian and J. Wu, 2015. Extended FRAM by integrating with model checking to effectively explore hazard evolution. Mathe. Prob. Eng., 2015: 11.

Friedenthal, S., A. Moore and R. Steiner, 2014. A Practical Guide To SysML: The Systems Modeling Language. 3rd Ed., Morgan Kaufmann, Waltham.

Guiochet, J., 2016. Hazard analysis of human-robot interactions with HAZOP-UML. Safety Sci., 84: 225-237.

Helle, P., 2012. Automatic SysML-based safety analysis. Proceeding of the 5th International Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB '12), pp: 19-24.

Hillenbrand, M., M. Heinz, J. Matheis and K.D. Müller-Glaser, 2012. Development of electric/electronic architectures for safety-related vehicle functions. Software Practice Exp., 42(7): 817-851.

IEC 61508, 2010. Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems. International Electrotechnical Commission, Geneva.

ISO 26262, 2011. Road Vehicles-Functional Safety. International Organization for Standardization, Geneva.

Jaradat, O., 2012. Automated architecture-based verification of safety-critical systems. M.S. Thesis, School of Innovation, Design and Engineering, Malardalen University, Vasteras, Sweden.

Joshi, A. and M.P.E. Heimdahl, 2007. Behavioral fault modeling for model-based safety analysis. Proceeding of the 10th IEEE High Assurance Systems Engineering Symposium, pp: 199-208.

Mauborgne, P., S. Deniaud, E. Levrat, E. Bonjour, J.P. Micaëlli and D. Loise, 2016. Operational and system hazard analysis in a safe systems requirement engineering process-application to automotive industry. Safety Sci., 87: 256-268.

Mehrpouyan, H., 2011. Model-Based hazard analysis of undesirable environmental and components interaction. M.S. Thesis, Department of Computer and Information Science, Linköping University, Linköping, Sweden.

Mhenni, F., N. Nguyen and J.Y. Choley, 2016. SafeSysE: A safety analysis integration in systems engineering approach. IEEE Syst. J., 12(1): 161-172.

MIL-STD-882E, 2012. Department of Defense Practice: System Safety. Department of Defense, Arlington.

Papadopoulos, Y., J. McDermid, R. Sasse and G. Heiner, 2001. Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. Reliab. Eng. Syst. Safety, 71(3): 229-247.

Paredis, C., 2008. Model-Based Systems Engineering: A Roadmap for Academic Research. In Lecture, Atlanta, Georgia.

Piriou, P.Y., J.M. Faure and G. Deleuze, 2014. A meta-model to support the integration of dependability concerns into systems engineering processes: An example from power production. IEEE Syst. J., 10(1): 15-24.

Sharvia, S. and Y. Papadopoulos, 2015. Integrating model checking with Hip-HOPS in model-based safety analysis. Reliab. Eng. Syst. Safety, 135: 64-80.

Wei, Q., J. Jiao and T. Zhao, 2017. Flight control system failure modeling and verification based on SPIN. Eng. Failure Anal., 82: 501-513.

Zhao, L., K. Thulasiraman, X. Ge and R. Niu, 2016. Failure propagation modeling and analysis via system interfaces. Mathe. Prob. Eng., 2016: 11.