An Approach to Safety Analysis and Verification based upon Formal Functional Model

Azma binti Abdullah and Shaoying Liu

Safety-critical systems are systems whose failure will cause a hazardous disaster, which could lead to loss of human life or damage of properties. To avoid post-development failures, a framework that analyses and verifies the requirement of safety-critical systems before implementing the system development is proposed. In this paper, the framework design is based on these views: (1) how to accurately and completely capture the functional and safety requirements; (2) how to incorporate safety requirements into the functional design specification; (3) how to verify the safety properties implied by the functional design specification; and (4) how to verify and validate the design specification is correctly implemented by a program. In deriving the answers for the problems, this research is designed to fulfill the following objectives: (1) To identify the approach in designing the safety analysis and document safety requirements; (2) To map the derived safety requirements to the form which is related to the formal functional model; (3) To verify the system model satisfies the safety requirement; and (4) To develop a prototype that support the above process to be effectively and efficiently implemented for quality assurance. The proposed framework is to be implemented in three phases: capturing safety properties, analyzing the hazard, and verifying through inspection to answer the first, second and third objectives, respectively. While in achieving the fourth objective, a prototype system that supports the three phases is developed in our framework.

1 INTRODUCTION

Safety is a property of a system frees the system from conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or the environment. Hence, safety-critical systems can be defined as systems whose failure will cause hazardous disaster, leading to the loss of human life or damage of properties. Hence, to ensure that the failure does not happen, safety analysis of the system requirements must be conducted. Safety analysis is the process of assessing the safety of a system by looking at the associated hazards and the methods used by the system to cope with them [4][6]. To assess the safety of the systems, we need to capture the functional and safety requirements of the systems must be captured, after which hazard analysis process is conducted.

In order to have confidence in the final system, it is necessary to verify that the requirement analysis phase has been correctly performed. This verification process is carried out by determining whether the output of a system, or module, meets its specification. Verification can be done through inspection, formal proof or testing. Formal methods are an effective way of improving the quality of documentation by providing formalism that allows requirements specifications to be written precisely and contributes to the design and verification [2]. There are several formal functional models, such as Event-B and SOFL (Structured Object-Oriented Formal Language); in this study, SOFL was chosen. The research goals of this research are divided into: (1) how to accurately and completely capture the functional and safety requirements; (2) how to incorporate safety requirements into the functional design specification; (3) how to verify the safety properties implied by the functional design specification; and (4) how to verify and validate the design specification is correctly implemented by a program. In order to fulfill the goals, the research is based on the following objectives: (1) To identify the approach in designing the safety analysis and document safety; (2) To map the derived
safety requirements to the form which is related to the formal functional model; (3) To verify the system model satisfies the safety requirement; and (4) To develop a prototype that support the above process to be effectively and efficiently implemented for quality assurance. To achieve the objectives, the framework, as illustrated in Fig. 1, consisting of three stages is proposed: (1) Stage 1 - Capturing Safety Properties to achieve Objective 1; (2) Stage 2 - Hazard Analysis to achieve Objective 2; and (3) Stage 3 - Verification Through Inspection to achieve Objective 3. Objective 4 will be discussed in Section IV. Based on the framework, each step that describes an operation is represented by a rounded rectangle box in the figure, and each data item, such as safety-related function or functional scenarios, is represented by a card box. An arrow from a card box to an operation means that the data item of the card box is an input to the operation, and an arrow from an operation to a card box means that the data item of card box is an output of the operation. An arrow from one operation to another operation shows a control flow.

The paper is organised as follows: Section II discusses SOFL, the formal specification model used in this research. Detailed explanation of the operations is discussed in Section III, Section IV presents the development prototype tools, and finally, Section V concludes the study.

2 STRUCTURED OBJECT.ORIENTED FORMAL LANGUAGE (SOFL)

In this research, SOFL is used to write the specification document. SOFL is chosen as it provides a software process model that can be used to systematically develop large-scale and complex systems. Fig. 2 illustrates the SOFL process model, which consists of three main phases: requirements analysis, design and coding.

System development starts with requirements analysis and it is an activity to discover, understand and document the user requirements. This phase consists of informal specification and semi-formal specification. Informal specification is written in a natural language and includes the functions to be implemented, the resources to be used and the necessary constraints on both functions and resources. Semi-formal specification is to clarify and define all the functions, resources and constraints, and to determine the relations among the three parts contained in the informal specification. Evolution is a transformation from informal to semi-formal, and then to formal specifications. The primary purpose of validation is to ensure that the written specification reflects the user’s requirements accurately and completely.

![Fig. 1 The Framework Process](image-url)
Abstract design transforms the semi-formal requirements specification into a formal implicit specification, which will then be further refined into formal explicit specification by detail design. Detail design has two goals: (1) transforming implicit specifications of processes and functions, defined in modules into explicit specifications to serve as a foundation for implementation in a specific programming language, and (2) transforming the structured abstract design specification into an object-oriented detailed design specification in order to achieve good quality of final implementation. Verification of a specification aims to ensure that the specification is internally consistent, acceptable and actually met by their implementation (or program).

Program is an implementation of the detail design in a specific programming language. It is essential to ensure that a program transformed from a detailed design (an explicit specification) satisfies the specification. There are four levels of transformations: (1) transformation of the abstract data types, (2) transformation of explicit specifications of process, methods and functions, (3) transformation of modules, and (4) transformation of classes. The major features of the process model include (1) informal and semi-formal specifications for user requirements and formal specification for abstract design, (2) specification for evolution process through informal, semi-formal and formal stages, and refinement for detailed design and implementation, and (3) rigorous review and testing to verify and validate specifications and programs [2].

3 RESEARCH WORK

This study is divided into three stages, which are capturing safety properties, hazard analysis and verification through inspection. For the requirement specification, it is written in SOFL as it provides a formal and comprehensible language for the requirements and design specification as well as a practical method for developing the software systems. Detail explanation of every stage is represented in this subsection.

3.1 Stage 1 - Capturing Safety Properties

Functional requirements documents describe what the system should do, and it is conceivable that documents may be incomplete and contain mistakes [4]. Safety properties are one of the outputs of functional requirements for safety-critical systems. Fig. 3 explains the process of capturing safety properties from functional requirements, and it consists of three steps: (1) Capturing the desired safety-related functions, necessary data resources and constraints, (2) Deriving functional scenario from safety-related function, and (3) Deriving safety properties from functional scenario using five keys as guidelines. These processes and five keys will assist the developer in finding and identifying the appropriate safety properties required to ensure that the related functions are free from failure. The five keys for capturing safety properties are functional constraints, domain knowledge, developer experience, real-time constraints for functions and the input/output device.
### 3.2 Stage 2- Hazard Analysis

Hazard analysis is a process that is conducted after the safety properties of a system have been captured. The aim of the hazard analysis is to deliver a system which does not pose an unacceptable danger to its end-user or to the environment in which the system is installed [3].

The process of hazard analysis, as illustrated in Fig. 4, consists of three steps: (1) Deriving hazards from safety properties, (2) Using Fault Tree Analysis (FTA) to analyse the possible cause for each hazard, and (3) Converting each minimal cut-set of FTA into a formal property in terms of variables used in the formal specification. The events are converted to formal language as each result needs to be verified and the conversion is related to the formal functional model.

### 3.3 Stage 3- Verification Through Inspection

Software inspection is a static analysis technique that is widely used for detection in document specification [5]. The main idea of the inspection is to have another person look at and examine the work you have made. The approach of verification through inspection, shown in Fig. 5, consists of three steps: (1) Linking functional scenario to events, (2) Analyzing linking to detect defect, and (3) Produce inspection report. The aim of the method is to utilise inspection to determine whether every functional scenario defined in the safety-related function is correctly implemented and does not have any relation with the events from fault tree analysis.
4 DEVELOPMENT

The objective of the development phase is to develop a prototype tool to support the process, from stage one until stage 3, to be effectively and efficiently performed for quality assurance. The main form for prototype is shown in Fig. 6, and it will assist the user to automatically perform the process.

The tool offers the following functions.

For stage 1, the user is able to choose to capture safety properties either for a single process or for CDFD of the systems. The tool will list all the functions, and when the user click on any particular function, all the data resources, constraints, semiformal specification, formal specification and the CDFD for the function will occur. This would make the selection of the safety-related function easier for the user. Also, the tools can automatically generate functional scenario from CDFD and suggest functional scenario for safety-related function. The user can then edit the suggestion functional scenario if they think it is not correct.

For stage 2, the user can develop fault tree analysis (FTA) for hazard analysis using the same tool. The tool also helps the user to obtain the suitable variable from the formal functional model for minimal cut-set in FTA.

For Stage 3, the tool will give the suggestion for linking the path from functional scenarios to events. The tools also guide the user in filling-up the checklist form and produce an inspection report. A print-out of the form and the report can be made available.

5 CONCLUSION

This paper illustrates an approach to safety analysis and verification based on formal functional model. The objectives for this research are: (1) To identify the approach in designing the safety analysis and document safety requirements, (2) To map the derived safety requirements to the form which is related to the formal functional model, (3) To verify the system model satisfies the safety requirement, and (4) To develop a prototype that support the above process to be effectively and efficiently implemented for quality assurance. The proposed framework for this research is implemented in three phases: capturing safety properties, analyzing the hazard, and verifying through inspection to answer the first, second and third objectives, respectively. While in achieving the fourth objective, a prototype system that supports the three phases in our framework is developed. This research focuses only on safety-critical systems and uses the formal functional model SOFL to write the requirement specification. This research can also be implemented to the other formal functional model using the same three stages as already suggested but some modifications to the process may be required to make it more suitable for use in the other model.

REFERENCES