Safety Justification and Safety Case for Safety-critical Software in Digital Reactor Protection System

Kee-Choon Kwon[a]*, Jang-Soo Lee[a], Eunkyoung Jee[b]

[a]*Korea Atomic Energy Research Institute, 989-111 Daedeok-daero, Yuseong-gu, Daejeon, Rep. of Korea*

[b]*Korea Advanced Institute of Science and Technology, Republic of Korea*

[*]*Corresponding author: kckwon@kaeri.re.kr*

1. Introduction

The establishment of the reliability and safety is highly important in developing the safety-critical software of the digital reactor protection system in nuclear power plants. These two features are important nature of safety-critical software in safety instrumentation and control system of nuclear power plants. Nuclear safety-critical software is under strict regulatory requirements and these regulatory requirements are essential for ensuring the safety of nuclear power plants. The verification & validation (V&V) and hazard analysis of the safety-critical software are required to follow regulatory requirements through the entire software life cycle. In order to obtain a license from the regulatory body through the development and validation of safety-critical software, it is essential to meet the standards which are required by the regulatory body throughout the software development process. [1].

Generally, large amounts of documents, which demonstrate safety justification including standard compliance, V&V, hazard analysis, and vulnerability assessment activities, are submitted to the regulatory body during the licensing process. It is not easy to accurately read and evaluate the whole documentation for the development activities, implementation technology, and validation activities Therefore, a systematic evaluation technology for the activities provided by nuclear power plant manufacturer is required in order to determine whether the target software has an acceptable level of safety. The safety case methodology has been kwon a promising approach to evaluate the level and depth of the development and validation results.

A safety case is a structured argument, supported by a body of evidence that provides a compelling, comprehensible, and valid case that a system is safe for a given application in a given operating environment [2]. The safety case approach is considered an effective way to argue for and evaluate system safety and it has been contrasted with prescriptive or process-based approaches, which assume that following the process prescribed in safety standards will generate evidence for safety. Since the safety case approach and the safety justification approach each have their own merits, these two approaches could complement each other [3]. The relationship between the dependability strategy and safety case is depicted in Fig. 1.

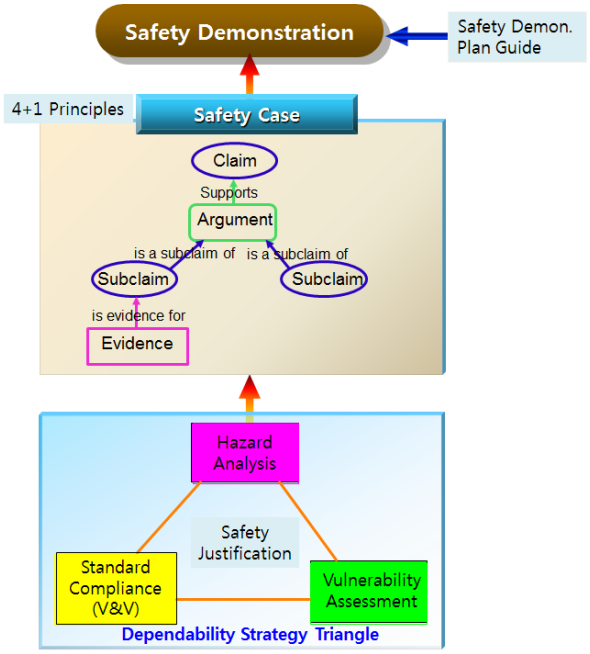

Fig. 1. Safety justification and safety case

2. Safety Justification

The main activities for safety justification in dependability strategy triangle as shown in Fig. 1 are V&V and hazard analysis. To meet the regulatory requirements and design goals, the software V&V and hazard analysis criteria and requirements are based on codes and standards including the BTP HICB-14 of NUREG-0800 SRP, Regulatory Guide 1.152, IEEE Std. 7-4.3.2, IEEE Std. 1012, IEEE Std. 1228, and etc.

2.1 V&V for Safety-Critical Software

The V&V activities in the software requirement phase is a licensing suitability evaluation, the Fagan inspection, a traceability analysis, the formal verification for formal specification, and test preparation. The V&V activities in the software design phase are almost the same as those in the software requirement phase. Formal verification is a little

different if one uses a different formal method in design phase. In the implementation phase, we also perform a licensing suitability evaluation, a Fagan inspection, and a traceability analysis. However, in this phase, the main activity is testing, specifically component testing. The integration phase can be divided into a software integration phase and a system integration phase. The main V&V activity in the software integration phase is integration testing. In the system integration phase, the main V&V activities are system testing and/or acceptance testing [4].

2.2 Software Hazard Analysis

The HAZOP method has been suggested for a hazard analysis in the software requirement and design phases. At the design phase, the software HAZOP was performed first, and software FTA was then applied. The software FTA was applied to some critical modules selected from the software HAZOP analysis. The software FTA can obtain some valuable results that have not been identified through a rigorous V&V procedure. The hazard analysis for safety-critical software is specifically the differences compared to other non-safety software qualification In the planning phase, we need to create a safety plan and shall perform a hazard analysis accordingly [4].

3. Safety Demonstration Through Safety Case

Existing safety demonstration is to build documents in each software life cycle for software development and validation and to provide them to the regulatory body. More than 300 documents were produced for the reactor protection system and more than 50 documents among them were for software of bistable processors. The regulators or reviewers have reviewed all the documents and it took a long time to have the conviction, "this software is safe to acceptable levels." It is difficult to visualize the flow to preach to the assurance life cycle stages.

In this paper, in order to take advantage of the GSN (Goal Structuring Notation) technique [5,6], UK's Adelard ASCE (Assurance and Safety Case Environment) 4.2.7 was used as a software tool. In general, safety guarantees safety demonstration through safety case which consists of safety claim, safety evidence and safety argument for systematic and schematic safety demonstration [7]. We implemented a safety case for bistable processor (BP) and Coincidence processor (CP) in digital reactor protection system. Fig. 2 shows a bird's eye view for safety case of BP/CP software. The top goal, "Safety-critical graded SW of the RPS is acceptably safe to operate on the PLC," for the BP/CP software is shown in Fig. 3. C1 and C2 are context in which the claim should be interpreted. C1 stated that BP is the part of the reactor protection system and C2 desribes the platform manufacturer and the type of PLC products. A1 is the assumption against the claim that the PLC on which the BP program runs is reliable.

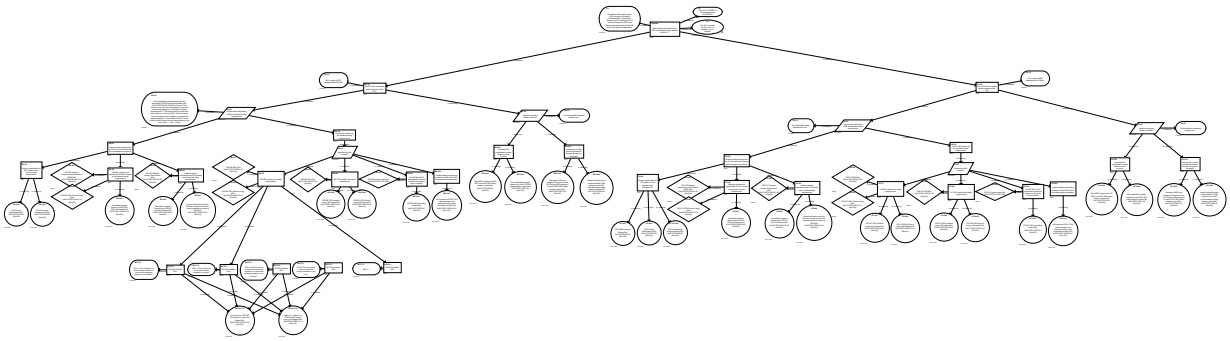

Fig. 2. The BP/CP software safety case: Bird's eye view

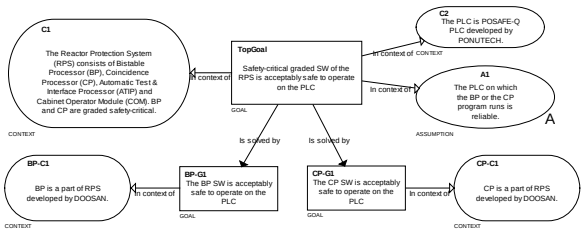

Fig. 3. The BP/CP software safety case: Top goal

BP-G1, ""The BP software is acceptably safe to operate on PLC.", as shown in Fig. 4, is the top goal for the safety of BP. In order to support the claim of BP-G1 in this sample, the two strategies (BP-S1, BP-S2) are established: argument by satisfaction of all the identified safety requirements (BP-S1) and argument by safety analysis activities such as hazard analysis (BP-S2).

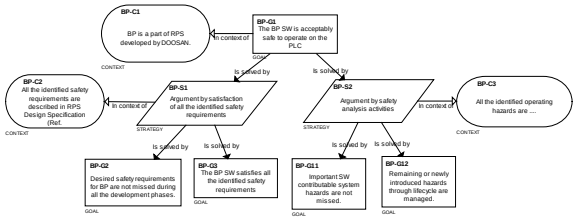

Fig. 4. The BP software safety case: BP-G1 goal

Fig. 4 shows that the strategy BP-S1 is solved by BP-G2 and BP-G3. It can be claimed that the BP software is acceptably safe to operate on PLC if the desired safety requirements for BP are not missed during all the development phases (BP-G2) and BP software satisfies all the identified safety requirements (BP-G3). BP-S2 is solved by BP-G11, "Important SW contributable system hazards are not missed," and BP-G12, "Remaining or newly introduced hazards through lifecycle are managed."

As shown in Fig. 5, the BP-G2 goal claiming that "the desired safety requirements for BP are not missed during all the development phases" can be split into three sub goals: "the design specification for BP includes all the desired safety requirements" (BP-G4); "the software requirement specification for BP includes all the desired safety requirements" (BP-G5); and "the software design specification for BP includes all the desired safety requirements" (BP-G6).

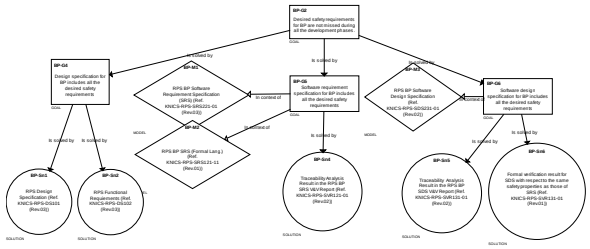

Fig. 5. The BP software safety case: BP-G2

Fig. 6 shows four subgoals under BP-G3: the claim that "BP software requirements specification meets the safety requirements." (BP-G7), the claim that "BP software design specifications satisfy the safety requirements." (BP-G8), the claim that "the BP SW implemented in the PLC generates the desired outputs for the given input scenarios." (BP-G9), and the claim that "implementation and testing results for the BP SW on PLC are independently evaluated" (BP-G10).

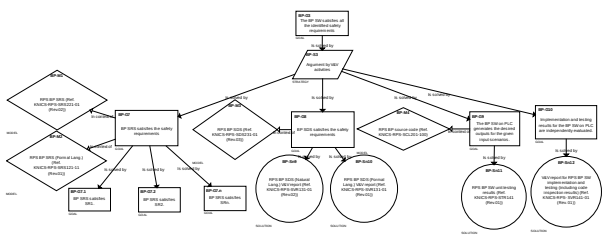

Fig. 6. The BP software safety case: BP-G3

BP-G7 claiming that "BP SRS satisfies the safety requirements.", as shown in Fig. 7, can be split into *n* subgoals where *n* is the number of safety requirements. For example, BP-G7.1 is "BP SRS satisfies SR1 (the first safety requirement) and BP-G7.2 is "BP SRS satisfies SR2." BP-G7.1~n can be supported by the evidence such as SRS natural language V&V report and formal SRS V&V report. For example, BP-G7.1~n are supported by the solution BP-Sn7.1.1 (6.1.5 section in RPS BP SRS (Natural Lang.) V&V Report) and the solution BP-Sn7.1.2 (Table 6.1 ~ Table 6.7 in the formal BP SRS V&V report).

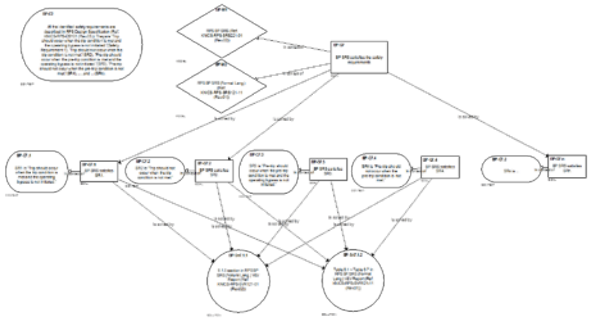

Fig. 7. The BP software safety case: BP-G7

Fig. 8 describes how the top claim (the safety of the BP SW) can be argued by safety analysis activities, in addition to satisfaction of safety requirements. As illustrated in Figure 8, if important SW contributable system hazards are not missed (BP-G11) and the remaining or newly introduced hazards through lifecycle are managed (BP-G12), the BP SW can be claimed to be acceptably safe to operate on the PLC. The software HAZOP was performed in the software hazard analysis during the requirements phase of the BP development, and software HAZOP and software FTA techniques were used in the design and implementation phase. Thus, software HAZOP result for the BP SRS in the RPS SRS hazard analysis report (BP-Sn13) and the software contributable system hazard list in the RPS SDS hazard analysis report (BP-Sn14) can both serve as evidence supporting claim BP-G11.

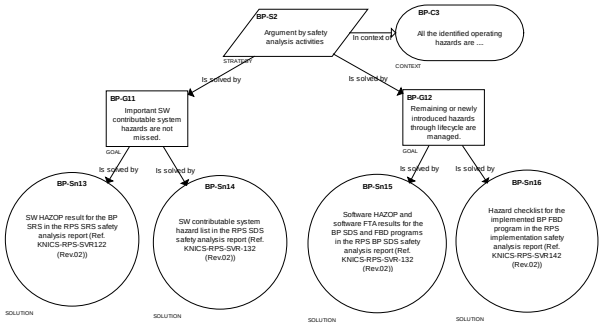

Fig. 8. The BP software safety case: Hazard analysis

4. Conclusions

It is suggested to evaluate the level and depth of the results of development and validation by applying safety case methodology to achieve software safety demonstration. A lot of documents provided as evidence are connected to claim that corresponds to the topic for safety demonstration. We demonstrated a case study in which more systematic safety demonstration for the target system software is performd via safety case construction than simply listing the documents. It is necessary to develop techniques to assess the appropriateness of demonstration which connects suggested claim and evidence and to determine if one has enough argument to some extent in the future.

5. Acknowledgement

This work was supported by the Nuclear Safety Research Program through the Korea Foundation of Nuclear Safety (KOFONS), granted financial resource from the Nuclear Safety and Security Commission (NSSC), Republic of Korea (No.1403008).

REFERENCES

[1] K. Kwon, J. Lee, G. Park, E. Jee, Paradigm of Software Safety Assurance for Digital Reactor Protection System in NPPs, KCC2016, Jeju, June 29-31, 2016.
[2] MoD, Defence Standard 00-56 Issue 4 (Part 1): Safety Management Requirements for Defence Systems, UK Ministry of Defence (MoD).
[3] E. Jee, G. Park, J. Lee, K. Kwon, A Safety Case for Reactor Protection System Software Developed with a Prescriptive Approach, EHPG 2016, Oslo, pp. , May 10, 2016.
[4] K. Kwon, et al., Qualification of safety-critical software for digital reactor safety system in nuclear power plants, Nuclear Safety and Simulation, Vol.4, No.3, pp. 226-233, 2013.
[5] Tim P. Kelly, “Arguing safety: A Systematic Approach to Managing Safety Cases,” University of York, 1999.
[6] John Spriggs, GSN – The Goal Structuring Notation : A Structured Approach to Presenting Arguments, Springer, 2012.
[7] Terje Siversten, Software Safety Demonstration, Halden Reactor Project, HWR-1056, 2013.