Failure Mode and Effect Analysis of the Application Software of the Safety-critical I&C System in APR1400

Koheun Kim [a*], Yong geul Kim[a], Woong seok Choi[a], Se do Sohn[a,]
[a] *KEPCO Engineering & Construction, Inc., 989-111 Daeduckdaero, Yuseong-gu, Daejeon, Korea*
[*] *Corresponding author: high07@kepco-enc.com*

1. Introduction

Safety computer system development requires identification of hazards that have the potential for defeating a safety function in nuclear power plant (NPP). Hazards include external events as well as conditions internal to the computer hardware and software [1]. In APR1400, the computer software hazard analysis is performed by hazard and operability analysis (HAZOP) method. Meanwhile, HAZOP has its limitation and cannot be considered better than fault tree analysis (FTA) or failure mode and effect (FMEA) analysis. HAZOP assumes that the system has been carefully studied, and all possible hazards, their effects or consequences and remedies are incorporated in the system. But incorporating every possible event in the design is impossible [2].

In this light, this paper attempts to use FMEA method for evaluating the risk for safety-critical instrumentation and control (I&C) system software for NPP which is more practically than HAZOP. Moreover, the possible common cause failure (CCF) due to software is evaluated from its result, though the FMEA is for single failure analysis. It is possible because the software failures are due to systematic faults that causing simultaneous failure in multiple division when the triggering event happens [3]. This analysis is applied to safety-critical system of Shin-Hanul units 1 and 2 NPP, i.e., APR1400.

2. Methods and Results

In this section, it introduces the software FMEA and an example for the application of the proposed methodology to the plant protection system (PPS) of APR1400.

2.1 Software FMEA

FMEA is a qualitative method of reliability analysis which involves the study of the failure modes which can exist in every sub-item of the item and the determination of the effects of each failure mode on other sub-items of the item and on the required functions of the item. The software FMEA (SFMEA) is a version of the FMEA for software. Sometimes, the SFMEA requires the knowledge of analyzer on the software because only little data may be available to perform SFMEA [4]. To perform the FMEA on software, it should set-up an approach by its own knowledge and take into other studies that have performed it. In terms of its sight, SFMEA includes the following steps:

1) Define the system to be analyzed
2) Construct the functional block diagrams.
3) Identify and set up the list of all potential software failure modes
4) Evaluate each failure mode in terms of the worst potential consequences.

Initially, the scope of the software needs to be defined. Then, the unit of the software component should be defined for the analysis. This study regards the modules of software as the software component. Generally, the application software of safety-critical systems is designed in modules (or tasks). The component unit could be defined in sub-modules or even in the smallest functional block level of the application software, depending on the determination by the safety engineer.

Based on the software unit, the failure mode is investigated. Next, each module is analyzed for its failure mode effect, cause, criticality, and significance of CCF as illustrated in Table 1. Through the process, the failure modes which may lead to CCF are identified and evaluated to determine the associated risk level (e.g., high or intermediate or low) based on the failure effect. The failure modes that are classified as having high criticality and significant CCF are evaluated. As a result, SFMEA identifies the failure modes and their causes that lead software failure and furthermore CCF in the system.

Table I. SFMEA Matrix Column Headings

Heading	Contents
Ref. No	Number to identify in the block diagram
Component	Name of the Component
Failure Mode	Potential component failure mode
Cause	Cause of the Failure mode 1) Programming error: Implementation phase 2) Human error: O&M phase 3) Design error: Concept, Requirement, Design phase 4) System or hardware error: Interface with component designer
Failure effect	Failure effect and the consequence
Criticality	C: high, CM: limited, NC: low
Significant CCF	Y:Yes, N: No

Comment/Value	Comments or value of the output signal that affects to the other task or system

2.2 SFMEA for PPS

This section presents an example for the application of the SFMEA to the PPS application software of Shin-Hanul units 1 and 2, i.e., APR1400. It identifies the possible software failure modes and their causes which lead to significant CCF in PPS. The scope of the software in PPS is limited to the bistable processor (BP) and coincidence processor (CP) protection grade application software. The unit of the software components is defined in task or called as program. Each and every task executes on its own determined cycle time but the data are interdependent on each other. BP application software is composed of four (4) tasks which execute sequentially on its priority;

1) BP_LOGIC 1.0
2) BP_LOGIC 1.1
3) BP_DIAGNOSTIC 2.0
4) BP_SETPOINT 3.0

Likewise, CP application software is composed of two (2) tasks which execute sequentially on its priority;

1) CP_LOGIC 1.0
2) CP_SUPERVISING 2.0.

BP and CP application software architecture for Shin-Hanul units 1 and 2 is depicted in detail in the reference Kim, C.H. et al. (2011) [5].

The definition of realistic failure modes for CCF is an important input to the SFMEA, therefore, the task failure modes are derived from the definition of its correct behaviors, which means the failure modes are the opposite of the correct behaviors:

1) Task execution time is bounded.
2) Task performs intended actions and does not perform unintended action.
    A. Provides expected outputs.
    B. Variables keeps their value until they are modified.
    C. Interacts as expected with I/O board.
    D. Interacts as expected with CPU resources.
    E. Interacts as expected with shared memory (or retention memory).
    F. Does not modify code memory and constants.

In addition, the PLC manual supplements the defined failure modes by defining the failure mode precisely. Before the evaluation, the assumptions of the system should be set up. The assumptions of the PPS application software are as follows;

1) Test logic does not prevent the system from performing its protective function
2) Communication between processors and the validity signals are provided with signals. The validity signals are for checking the integrity of the signals in the receiving processor.
3) Though a task has an error which is detected and the task stops, the invalid data from the task will not affect to the other tasks.
4) Fail-safe status is defined by the designer. Meanwhile, the failures which might be detected by system software and led to a fail-safe state are excluded for high criticality.
5) Input read and output read are controlled by system task, i.e., operating system.

Table II. SFMEA of example

Heading	Contents
Ref. No	1.0
Component	BP_LOGIC
Failure Mode	Provides erroneous values with valid hardware status
Cause	Programming error, Hardware read error or Design error
Failure effect	1) Erroneous values are transmitted to the coincidence processer (CP) with valid status 2) Risk of unsafe output
Criticality	C
Significant CCF	Y
Comment/Value	Value: Trip Validity bit: Trip quality

As a result, the SFMEA for the PPS identified 11 failure cases that are critical and can lead to significant CCF. One of the cases is illustrated in Table II. Table II shows the result of SFMEA for the selected failure mode, i.e., “Provides erroneous trip values when Hardware is ok”. This failure mode prevents the PPS from initiating the trip signal in the BP due to erroneous process values, when the trip is required. This failure mode could be caused by “programming error” or “hardware read error”, or “design error”. The reactor trip is related to the safety function in the system. In this regard, failure mode of the trip value is considered as critical and significant.

This model have acquired some advantages from using SFMEA compare to HAZOP method which also carry out risk analysis: 1) SFMEA makes a systematical and analytical approach to figure out the possible failure modes and the causes in the software; 2) SFMEA entails the analysis of software architecture and categorizes the level of criticality to the modules.

These results can helps later on with the developing the software.

3. Conclusions

Through SFMEA, the critical software failure modes and tasks that could result in CCF are identified and also evaluated to determine the associated risk level (e.g. high or intermediate or low) based on the failure effect. Biggest benefit from this analysis comparing with HAZOP is it can reveal the possible weak points and provide the guidance to the V&V team by helping to generate the test cases. Furthermore, the list of causes could be organized and the relationship among the elements that contributes to CCF could be substantiated.

REFERENCES

[1] IEEE, IEEE 7-4.3.2, IEEE Standard Criteria for Digital Computer in Safety Systems of Nuclear Power Generating Station ,2003.
[2] A.A. Haider and A. Nadeem, “A Survey of Safety Analysis Techniques for Safety Critical Systems”, International Journal of Future Computer and Communication, Vol.2, No.2, April 2013.
[3] E. R. S, J.-B. Mariana, A. Yousef and B. Herve, "Modeling of digital I&C and software common cause failures: lessons learned from PSA of TELEPERM XS based Protection System," in PSAM 12, 2014.
[4] L. Ristord and C. Esmenjaud, “FMEA Performed on the SPINLINE3 Operational System Software as part of the TIHANGE 1 NIS Refurbishment Safety Case”, NEA/CSNI/R (2002)1/VOL2 (2001).
[5] C. H. Kim, D. Y. Oh, K. Kim, S. D. Sohn, J. H. Kim, H. B. Kim and W.S. Choi, "Development of Safety Critical Software for Nuclear Power Plant using a CASE Tool," in ICI2011 (ISOFIC, CSEPC, ISSNP 2011), Daejeon, Korea, 2011.