Reliability modeling of digital RPS with consideration of undetected software faults

M. Khalaquzzaman[a*], Seung Jun Lee[a], Man Cheol Kim[b], Wondea Jung[a]
[a]*Integrated Nuclear Safety Assessment Division, Korea Atomic Energy Research Institute*
1045 Daedeok-daero, Yuseong-gu, Daejeon, 305-353, Korea
[b]*School of Energy Systems Engineering, Chung-Ang University, 84 Heukseok-ro, Dongjak-gu, Seoul 156-756, Korea*
[*]*Corresponding author: kzaman74@gmail.com*

1. Introduction

Digitalizing the reactor protection system of nuclear power plant has been initiated several decades ago and now full digitalization has been adopted in the new generation of NPPs around the world because digital I&C systems have many better technical features like-easier configurability and maintainability over analog I&C systems [1]. Digital I&C systems are also drift-free and incorporation of new features is much easier [2]. Rules and regulation for safe operation of NPPs are established and has been being practiced by the operators as well as regulators of NPPs to ensure safety. The failure mechanism of hardware and analog systems well understood and the risk analysis methods for these components and systems are well established. However, digitalization of I&C system in NPP introduces some crisis and uncertainty in reliability analysis methods of the digital systems/components because software failure mechanisms are still unclear. This paper provides overview of different software reliability methodologies and proposes a technic for estimating the reliability of RPS with consideration of undetected software faults.

2. Methodology

In this section the techniques are used to model for estimating the reliability of RPS has been briefly described. Traditionally, probabilistic safety assessment (PSA) of analog system is performed by fault-tree and event tress analysis based on hardware failure rates. However, traditional PSA based on only hardware failure would under estimate because digital system fails for both hardware and software failure. Software failures are observed to be dominant over hardware failures in a digital system [4]. The reliability modeling techniques for random hardware failure and software failure have been described in the following sub-sections:

2.1 Hardware Reliability

Exponential distribution is most widely used for hardware reliability assessment. If λ is the constant random failure rate of a hardware component and *t* is the mission time then the reliability of the component is estimated based on the probability density function:

$$f(t) = \lambda e^{-\lambda t} \quad \text{.....(1)}$$

Probability of failure for the period of *t*,

$$Q(t) = 1 - e^{-\lambda t} \quad \text{.....(1)}$$

2.2 Software Reliability

Over the decades, a good numbers of statistical methods have been developed for the software reliability estimation. However there is no consensus on a method of software reliability analysis [3]. The software reliability growth model (SRGM) is the most matured technique for a software dependability assessment. In this technique, the increment of reliability is measured due to the removal of detected faults from the software. However, the SRGM technic is believed to be inappropriate for reliability assessment of safety critical software because of the argument on effectiveness of fault removal and chance of introducing new faults during fixing of discovered faults [5].

An alternative approach to estimate number of remaining faults after verification and validation (V&V) of software is evaluating the software development life cycle (SDLC), which has been described by H. S. Eom et al. [6]. The method is expected to overcome the problems of conventional methods. However utilizing the SDLC modeling with BBN and random testing results, we propose to incorporate BBN based fault estimation and random testing of software for estimation of defect density in the software. In other words, we propose, the initial estimation of faults by assessing SDLC process and then reliability estimation trough assessing the estimated faults from the software test results.

2.2.1 Estimating remaining faults after V&V

Software development model called *Waterfall Model* has been considered as standard for estimation of remaining faults. Faults are estimated by Bayesian Belief Network (BBN) model in which the inputs are the outcomes of the qualitative assessment SDLC activities with respect to a fixed criterion.

2.2.2 Re-evaluation of remaining faults by testing

Number of faults estimated by BBN could be used to estimate probable fault density, ϕ (i.e. no of faults per functions) in the software. This prior fault/defect density can be used to assess how these faults could affect the

demand failure criteria applying beta distribution for a series of random black-box testing, and probability of demand failure can be estimated.

Case-1: with evidences of test failures

When φ follows Beta(α,β) distribution and α>0 and β>0, the mean fault density can be revised as posterior on the random black-box test results based on the following formula for *f* failures out of total *n* tests:

$$\phi = \frac{\alpha + f}{n + \alpha + \beta} \text{.....} (iii)$$

Case-2: when no evidence of test failures

When there is no evidence of failure during *n* tests, posterior mean fault density can be estimated based on the following formula as shown in [7]

$$\phi = \frac{\alpha}{n + \alpha + \beta} \text{.....} (iv)$$

2.3 System Reliability

Based on the posterior probability of fault density using the test results, the expected failure probability of software is estimated and applied to estimate the system failure, and can be applied fault-tree analysis.

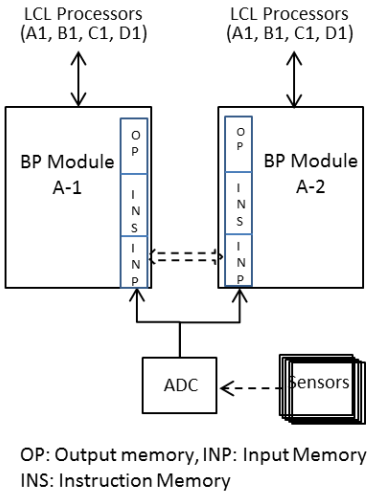

Fig.1. Simplified block diagram of BSP module.

Probability of the failure of bistable processor (BP) output is estimated as follows:

$$p(BPO) = p(BP) + p(BPI) \text{.....} (v)$$

$$p(BP) = p(hw) + p(sw) \text{.....} (vi)$$

Where, *p(hw):* probability of BP hardware failure, *p(sw):* probability of BP software failure, *p(BPI):* probability of bistable input failure

3. Conclusions

Software reliability analysis of safety critical software has been challenging despite spending a huge effort for developing large number of software reliability models, and no consensus yet to attain on an appropriate modeling methodology. However, it is realized that the combined application of BBN based SDLC fault prediction method and random black-box testing of software would provide better ground for reliability estimation of safety critical software.

REFERENCES

[1] JNES, Digital Instrumentation and Control System for Safety System and Main Control Room Design, JNESeL-TT-005, December 2011.

[2] H. K. Shin, Sang-Ku Nam, Se-do Sohn , Hang-Bae Kim, Digital System Reliability Test for the Evaluation of Safety Critical Software of Digital Reactor Protection System, Journal of Systemic, Cybernetic and Informatics; Vol. 4, No. 4, 2006.

[3] Y. Yaguang, S. Russell, Reliability Estimation for a Digital Instrument and Control System, Nuclear Engineering and Technology, Vol. 44, No.4, May 2012.

[4] J. H. Hayes, Building Requirement Fault Taxonomy: Experiences from a NASA Verification and Validation Research Project, Proceedings of the 14th International Symposium on Software Reliability Engineering (ISSRE'03), IEEE 2003.

[5] H. G. Kang, H. G. Lim, H. J. Lee, M. C. Kim, S. C. Jang, Input-profile-based software failure probability quantification for safety signal generation systems, Reliability Engineering and System Safety, 94,1542–1546, 2009.

[6] H. S. Eom, G. Y. Park, S. C. Jang, H. S. Son, H. G. Kang, V&V-based remaining fault estimation model for safety–critical software of a nuclear power plant, Annals of Nuclear Energy 51, 38–49, 2013.

[7] K. W. Miller, L. J. Morell, R. E. Noonan, S. K. Park, D. M. Nicol, B. W. Murrill, J. M. Voas, Estimating the Probability of Failure When Testing Reveals No Failures, IEEE Transactions on Software Engineering, Vol. 18, No. 1, 1992.