Software Reliability Estimation by Using Covariates

Gee-Yong Park [a*], Heung-Seop Eom [a], Seung Cheol Jang [a], Se-Woo Cheon [a], and Dong Hoon Kim [a]
[a]*Korea Atomic Energy Research Institute, 1035 Daedeok-daero, Yuseong, Daejeon, Korea*
[*]*Corresponding author: gypark@kaeri.re.kr*

1. Introduction

There is no consensus in using the software reliability in the nuclear power industry [1]. This is partly due to the fact that the software failure mode is not well defined and the failure mechanism of software is different from that of hardware. Nonetheless, the software reliability (SR) has become an increasingly important issue because digital systems are employed more frequently in the nuclear instrumentation and control (I&C) systems. As digital systems are replacing existing analog safety systems, a realistic estimate of the software reliability is necessary to touch all bases in the probabilistic risk analysis of digital safety systems and also to evaluate quantitatively the software quality.

2. Problems of Existing SR Methods

Even though there are over 200 methods proposed for the software reliability in the literature [2], there are some problems in the existing software reliability methods for use in the safety analysis of a nuclear power plant. Most of the existing software reliability methods have been based on the software reliability growth model (SRGM) [3]. The SRGM based on either failure time data or failures during time intervals requires an appropriately sufficient amount of failure data for estimating model's parameters and predicting the software reliability.

For the development of software for use in nuclear safety systems, safety software is usually developed under a rigorous development procedure and a verification and validation (V&V) process following code and standards [4][5]. Thus, it is highly plausible that software failure occurrence/detection phenomena are a so-called rare event system. That is, failures occurred during testing nuclear safety software are very small, being compared to the failures detected during testing commercial software in the non-nuclear fields. As a result, the existing SRGMs cannot find an optimal point in estimating their model parameters and hence, they are unable to make a statistical inference on such a rare event data. Another difficulty in applying the software reliability methods results from the fact that the V&V test team record in detail test inputs, outputs, and test results, but they may not log test data at the right time.

For coping with the problems described above, a software reliability modeling by using covariates such as the V&V activities or the test coverage is proposed. For incorporating either qualitative or quantitative covariate into the model and for estimating stably model parameters, the Bayesian inference is used in this model.

3. Model Descriptions

Even though the failure data may be rare during testing nuclear safety software, the information of the software quality and reliability can be postulated (qualitatively) when referring to the evaluation works in the V&V activities at the requirements and design phases of a software life cycle. Also, efforts for improving the software quality can be obtained by identifying the amount of test cases actually executed or by investigating test coverage.

For the V&V activities on software for use in a reactor protection system (RPS) [6], the V&V works were carried out based on some checklists, each of which was devised carefully for a corresponding software attribute in [4]. This information is used as the covariate in this paper.

For quantifying the qualitative V&V activities, it is assumed that each item in checklist for a particular software attribute is regarded as a block box testing like drawing a ball in an urn with white balls (Success) and black balls (Fail). In addition, the application range of a checklist item over software modules is considered and the range of each item is quantized in the integer-valued horizon. Table I presents some of quantification results of the V&V activities on the software requirements specification (SRS) for the Bistable processor (BP) which is a trip-functioning processor in the RPS.

Table I: One Example of V&V Checklist Quantification

V&V Activity	Property or Viewpoint	Attribute	Check Item	Test Trial	Test Succ.
Licensing Suitability Assessment	Process Characteristics	Completeness	55	275	272
		Consistency	10	50	48
		Correctness	11	55	55
		Style	8	40	38
		Traceability	4	20	20
		Un-ambiguity	2	10	7
		Verifiability	5	25	25

As is presented in Table I, the V&V checklist items both for the software requirements specification of trip-functions of the RPS and for the software design descriptions are all quantified. In the nuclear software V&V process, the anomalies, that is, the software defects, found by the V&V activities are resolved through a rigorous quality assurance program before a next phase begins to start. This fact is also considered

in modeling the software failure probability (SFP) by using the Bayesian inference.

Modeling the SFP is implemented by a computer tool, WinBUGS [7]. Fig.1 shows the posterior distribution of the SFP based on the quantified data of the V&V activities. (The WinBUGS program for the V&V activities is similar to that shown in Fig.2.)

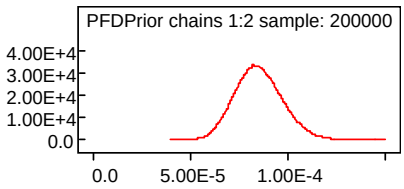


Fig. 1. Software failure probability from the V&V activities.

Based on the above result, another Bayesian modeling for the software failure data is constructed, where the posterior distribution from the V&V activities is used as a prior distribution.

By investigating all test reports, the number of test cases only related with the trip-functions is 1040 and, among these test cases, 6 failures were identified. The Bayesian modeling by WinBUGS for this failure data is shown in Fig.2.

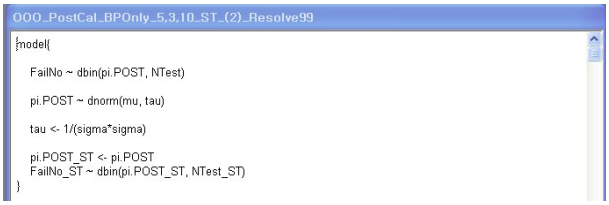
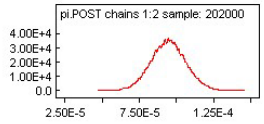


Fig. 2. WinBUGS program for failure data.

The posterior distribution from the model as shown in Fig.2 is shown in Fig.3 and, in this case, this posterior distribution might be regarded as the probability of failure on demand (PFD) of the RPS software.



node	mean	sd	MC error	2.5%	median	97.5%	start	sample
pi.POST	9.46E-5	1.152E-5	5.813E-8	7.215E-5	9.457E-5	1.175E-4	100001	202000

Fig. 3. Probability of failure on demand from the test data.

Below the graph in Fig.3 is the statistics for this graph (this graph and the graph in Fig.1 are obtained by the sampling based on the Markov-chain Monte Carlo simulation provided by the WinBUGS). The expected value of the software PFD is 9.465×10^{-5} as shown in the "mean" column in this statistics.

3. Conclusions

This paper describes the Bayesian modeling of software reliability for incorporating qualitative information about software quality and also for coping with the problem of estimating parameters in case of the rare-event-type failure data. The software V&V activities are quantified and incorporated into the model to generate the software failure probability. The software PFD is obtained from the Bayesian modeling, whose prior distribution is the SFP, for the failure data. As a covariate, test cases or other metrics such as the test coverage can be involved into a Bayesian software reliability modeling.

REFERENCES

[1] Reg. Guide-1.168, Verification, Validation, Reviews, and Audit for Digital Computer S/W used in Safety Systems of Nuclear Power Plants, U.S. NRC, 2004.
[2] K. Rinkasa, K. Shibata, T. Dohi, Proportional Intensity-Based Software Reliability Modeling with Time-Dependent Metrics, COMPSAC 2006, Chicago, USA, 2006.
[3] IEEE Std. 1633-2008, IEEE Recommended Practice on Software Reliability, Institute of Electrical and Electronics Engineers, 2008.
[4] NUREG-0800, Standard Review Plan: BTP HICB–14, Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems, U.S. Nuclear Regulatory Commission, 2007.
[5] IEEE Std. 1012, Software Verification and Validation Plans, 2004.
[6] G. Y. Park, et al., Software Qualification Activities for Safety Critical Software, NPIC&HMIT 2009, Knoxville, Tennessee, April 5-9, 2009.
[7] D. Spiegelhalter, A. Thomas, N. Best, and D. Lunn, WinBUGS User Manual, http://www.mrc-bsu.cam.ac.kr/bugs, 2003.