DEVELOPMENT OF A BAYESIAN BELIEF NETWORK MODEL FOR THE SOFTWARE RELIABILITY ASSESSMENT OF NUCLEAR DIGITAL I&C SAFETY SYSTEMS

Hyun Gook Kang, and Sang Hun Lee
Department of Mechanical, Aerospace, and Nuclear Engineering
Rensselaer Polytechnic Institute
110 8th St, Troy, NY, USA
kangh6@rpi.edu; lees35@rpi.edu

Seung Jun Lee
School of Mechanical and Nuclear Engineering
Ulsan National Institute of Science and Technology
50, UNIST-gil, Ulsan, Republic of Korea
sjlee420@unist.ac.kr

Tsong-Lun Chu, Athi Varuttamaseni, and Meng Yue
Brookhaven National Laboratory
Brookhaven Avenue, Upton, NY, USA
chu@bnl.gov; avarutta@bnl.gov; yuemeng@bnl.gov

Steve Yang
NUV Technology LLC
1620 Ridge Haven Run, Alpharetta, GA, USA
steve.yang@nuvtechnology.com

Heung Seop Eom, and Jaehyun Cho
Korea Atomic Energy Research Institute
111, Daedeok-daero, Daejeon, Republic of Korea
ehs@kaeri.re.kr; chojh@kaeri.re.kr

Ming Li
U.S. Nuclear Regulatory Commission
Washington, DC, USA
Ming.Li@nrc.gov

ABSTRACT

Since the digital instrumentation and control systems are expected to play an important role for the safety systems in nuclear power plants (NPPs), the need has emerged to not only establish a basis for incorporating software behavior into digital I&C system reliability models, but also to quantify the failure probability of the software used in NPP digital protection systems. In this study, a Bayesian belief network (BBN) model is developed to quantitatively assess software reliability by estimating the number of faults in a software program considering its software development life cycle (SDLC). The model structure and parameters are established based on the information applicable to NPP safety-related systems and the evidence used to construct and quantify the BBN model was collected from three stages of expert elicitation. The software failure probability is estimated from the number of residual defects in a software program at the end of SDLC phase. As a case study, the BBN model was applied to quantify the software reliability of a typical digital protection software having the size of 50 function points and having the Medium development and validation and verification (V&V) qualities. The developed model can be applied to estimate the failure probability for both developing and deployed safety-related NPP software, and such results can be used to evaluate the quality of the digital I&C systems in addition to estimating potential reactor risk due to software failure.

Key Words: Nuclear Power Plant; Probabilistic Risk Assessment; Software Reliability; Bayesian Belief Network

1 INTRODUCTION

The instrumentation and control (I&C) systems in nuclear power plants (NPPs) have recently been replaced with digital-based systems. The reason for this transition lies in the advantages of digital systems compared to conventional analog systems, such as fault-tolerance, self-testing, and system diagnosis. However, digital systems have distinct failure causes and modes compared to analog systems on account of their unique characteristics, such as software, and thus the incorporation of software failure into NPP digital I&C system probabilistic risk assessments (PRA) presents special challenges. Software failure in safety-related or safety-critical systems in digitalized NPPs can be defined as the triggering of a fault in the software, introduced during its software development life cycle (SDLC), that contributes to the host digital system failing to accomplish its intended function, or to initiate an undesired action. For example, failure of reactor protection system (RPS) software may induce the failure of reactor-trip signal generation when a trip condition occurs. As software failure can significantly affect a digital system, especially in the case of the NPP protection system [1, 2], software reliability must be quantified to guarantee NPP safety.

Previous studies on safety-critical software reliability quantification schemes have investigated a spectrum of related methods and identified potential ones that may serve to quantify software failure rates and per-demand failure probabilities of NPP digital systems, such that the system models can be integrated into a PRA [3]. Among the quantification schemes, a Bayesian belief network (BBN) method was considered as one of the potential candidates for the reliability quantification of the NPP digital protection software. Since the BBN method uses conditional probability tables to represent interdependency among disparate events, it presents the advantage of potentially combining qualitative information on the quality of software development activities with quantitative information such as test and operational data in quantifying the NPP safety graded software reliability.

In this study, a BBN model is developed that estimates the number of faults in a software program and the probability of software failure which can be further incorporated into an NPP PRA model. The model captures NPP safety-related SDLC activity quality indicators and software development information, establishes the quantitative causal relationships between the indicators and the number of remaining defects, and further estimates software failure probability. Especially, a practical BBN framework for NPP safety software reliability quantification is demonstrated by (1) identifying software development characteristics; (2) establishing and quantifying the causal relationships between these characteristics; (3) probabilistically aggregating multiple expert inputs; and (4) estimating the number of defects remaining, and the software failure probability using expert opinion.

2 MODEL DEVELOPMENT

The key steps in the model development is shown in Fig. 1 which consist of three phases of model construction and related expert elicitation. First step is to identify the attribute which represents the quality of activities carried out to accomplish the functions of each SDLC phase and to verify causal relationships represented by the BBN structure. The second step to quantify the causal relationship parameters for the model, without any software-specific observation; thus it can be applied to generic NPP safety-related software. Third step is to provide input values of nodes for a particular application software based on the software-specific evidence in order to estimate the software failure probability of the target software program.

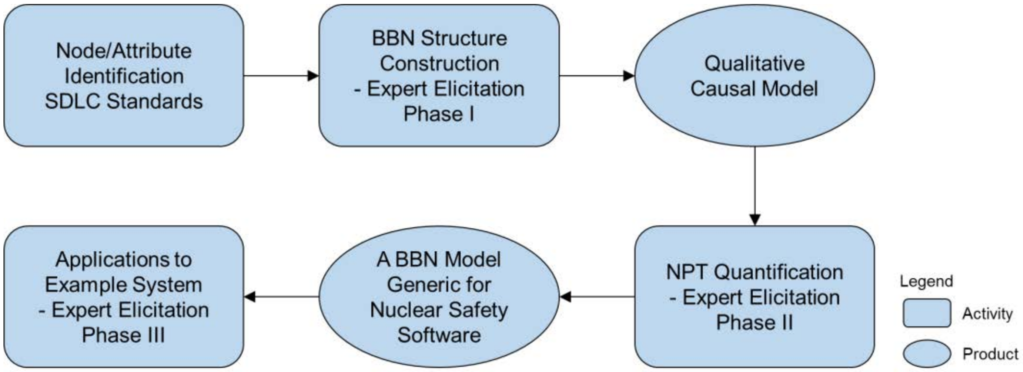

Node/Attribute Identification SDLC Standards
BBN Structure Construction - Expert Elicitation Phase I
Qualitative Causal Model
NPT Quantification - Expert Elicitation Phase II
A BBN Model Generic for Nuclear Safety Software
Applications to Example System - Expert Elicitation Phase III
Legend
Activity
Product

Figure 1. Bayesian Belief Network Model Framework for NPP Safety Software Reliability Quantification

2.1 BBN Model Structure

In this section, the detailed structure of the BBN model and the framework to quantify software failure probability based on the observations on the software development and verification and validation (V&V) quality throughout the SDLC are described. In the BBN model, the SDLC is considered to be consisted of five phases (i.e. Requirements, Design, Implementation, Test, and Installation-and-Checkout phases) and the number of software faults remaining at the end of each phase is estimated, as shown in Fig. 2. The model starts with the defects remaining in the Requirements phase and tracks the number of defects through all five phases of the SDLC. Any remaining defects at the end of one phase are passed on to the next phase and the total number of defects remaining in the software at the end of the last phase is further converted into a software failure probability on-demand.

In the model, the number of defects remaining in each phase is assumed to be dependent on two types of software development activities: development quality and V&V quality. The number of defects remaining in each phase is defined as a function of the development quality and the V&V quality, where the development process adds defects and the V&V process removes defects, at each SDLC phase.

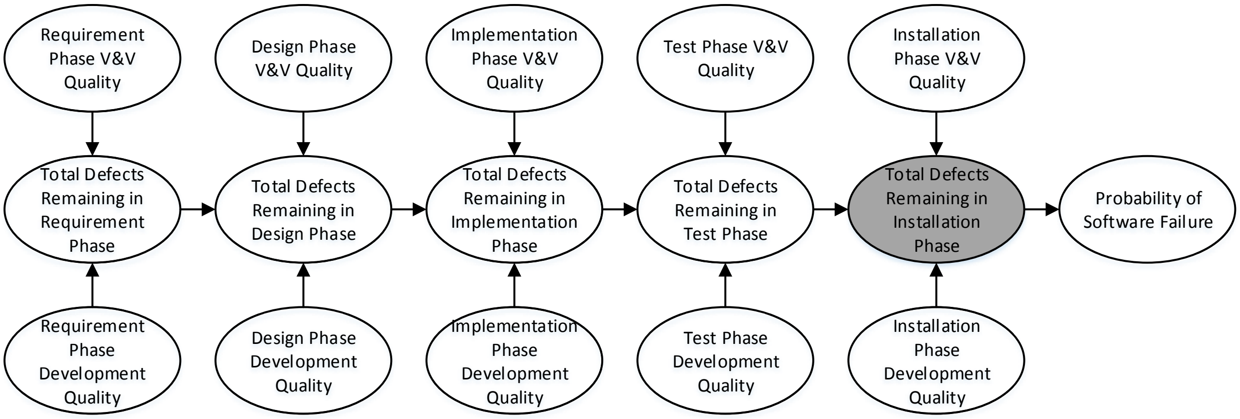
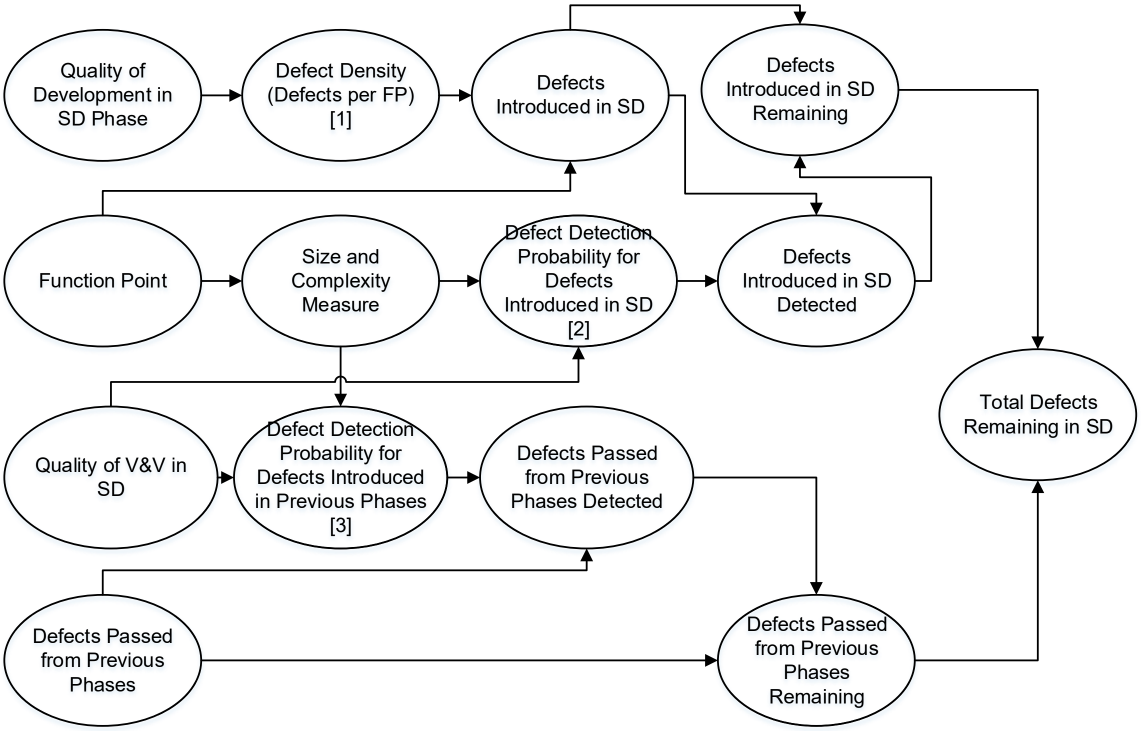

Requirement Phase V&V Quality
Design Phase V&V Quality
Implementation Phase V&V Quality
Test Phase V&V Quality
Installation Phase V&V Quality
Total Defects Remaining in Requirement Phase
Total Defects Remaining in Design Phase
Total Defects Remaining in Implementation Phase
Total Defects Remaining in Test Phase
Total Defects Remaining in Installation Phase
Probability of Software Failure
Requirement Phase Development Quality
Design Phase Development Quality
Implementation Phase Development Quality
Test Phase Development Quality
Installation Phase Development Quality

Figure 2. High-level Structure of the Bayesian Belief Network model for Software Reliability Quantification

As an example, the process of defect insertion and removal in the BBN model for the Design phase is shown in Fig. 3. The number of residual faults at a specific SDLC phase is determined by (1) the number of defects passed from the previous phase, if it is not the first phase of the SDLC, (2) the number of defects introduced during the current phase, and (3) the number of defects detected and removed by the V&V activities undertaken in the current phase. It is notable that the BBN structures of other SDLC phases can be modeled in a similar manner.

Another assumption in the model is that the quality of the software development activities is directly related to the defect density (defects per function point) in the current phase. Other factors affecting the number of faults inserted are the size and complexity of the software. Similarly, the quality of the V&V activities is directly related to the detection probability for defects introduced in the current phase and defects passed from the previous phase, thus, the number of defects introduced in current or passed from previous phases is reduced accordingly. In addition, the number of function points (FPs), which represents the size and complexity of the software is assumed to affect the defect detection probabilities.

For the rest of the model, (1) the number of defects per FP introduced during development is derived from the quality of development activities; (2) the number of defects detected and removed is modeled by a binomial distribution with parameters equal to the number of existing defects and the probability of detecting defects; (3) the number of defects remaining in the current phase is simply calculated as the sum of the number of remaining defects from the current phase and the number of defects remaining from earlier phases; and (4) the number of defects remaining in the current phase is transferred to the next SDLC phase.



Quality of Development in SD Phase
Defect Density (Defects per FP) [1]
Defects Introduced in SD
Defects Introduced in SD Remaining
Function Point
Size and Complexity Measure
Defect Detection Probability for Defects Introduced in SD [2]
Defects Introduced in SD Detected
Total Defects Remaining in SD
Quality of V&V in SD
Defect Detection Probability for Defects Introduced in Previous Phases [3]
Defects Passed from Previous Phases Detected
Defects Passed from Previous Phases
Defects Passed from Previous Phases Remaining

Figure 3. Sub-level Structure of the Bayesian Belief Network model for the Design phase

2.2 BBN Model Parameters

In the model, key parameters include attributes, software development and V&V, the number of FPs, defect density, and defect detection probability nodes used in the BBN model to quantify the number of residual defects at the last SDLC phase. The following sections discuss the detailed description of such parameters in the BBN model.

2.2.1 Attribute node

An attribute node is a node representing the quality in carrying out one or a collection of associated software development or V&V activities where the attribute nodes are modeled as indicator nodes which are related to their parent node by a specified node probability table (NPT), similar to the indicator nodes of Fenton et al. [4]. The major source used to identify the attributes and their associated activities includes IEEE Std. 1012 on V&V [5] and many other guidance reports and standards [6-10]. Fig. 4 shows the attribute nodes for the quality node representing the overall quality of the development activities in the Design phase. Each attribute node represents the quality in carrying out the associated or required activities, i.e. the attribute quality, and is modeled with three states as defined below.

- High: In addition to satisfactorily carrying out the required activities, additional activities were undertaken that are expected to significantly improve the quality of the work, and enhance the software's reliability.
- Medium: All required (or equivalent) activities were satisfactorily carried out.
- Low: Some of the required activities were not carried out satisfactorily.

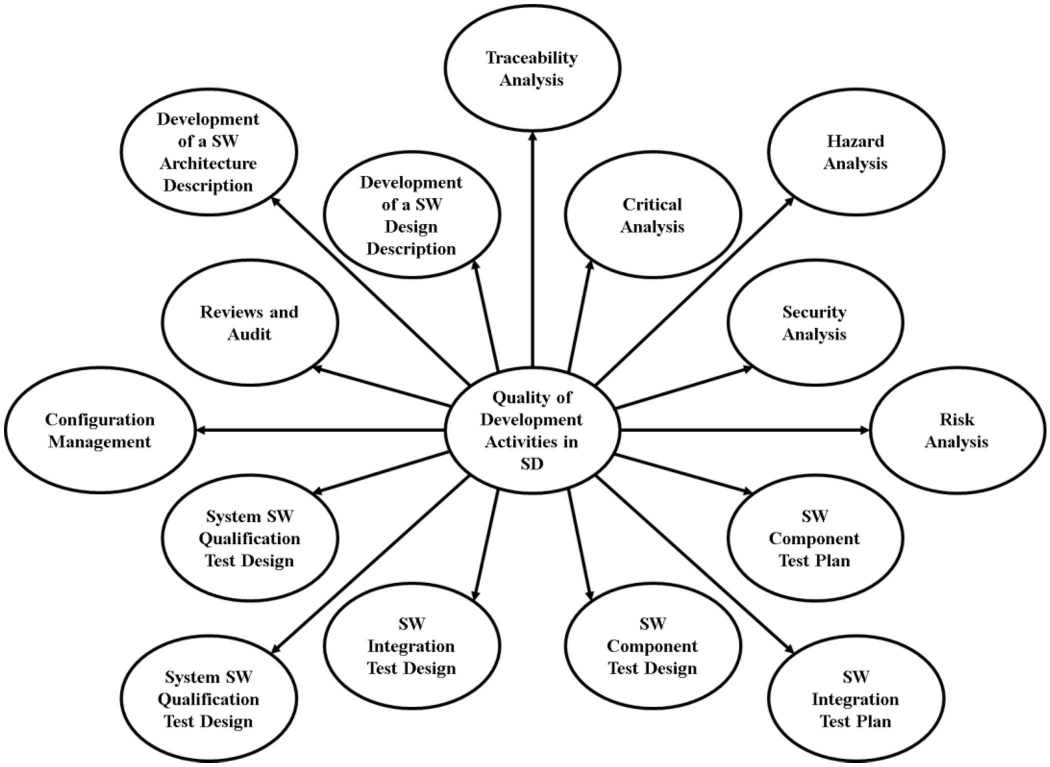

Traceability Analysis
Development of a SW Architecture Description
Hazard Analysis
Development of a SW Design Description
Critical Analysis
Reviews and Audit
Security Analysis
Configuration Management
Quality of Development Activities in SD
Risk Analysis
System SW Qualification Test Design
SW Component Test Plan
SW Integration Test Design
SW Component Test Design
System SW Qualification Test Design
SW Integration Test Plan

Figure 4. Attributes nodes for development quality at the software Design phase

2.2.2 Quality of development and V&V node

The quality of development and V&V nodes, which are the parent nodes of the attribute nodes as shown in Fig. 4, are connected to their respective attribute nodes in a diverging configuration. The states of the development and V&V quality nodes is considered to be representing the overall quality of the related activities. Similar to the attribute nodes, these nodes are also modeled as root nodes having three states, defined as below. Since the BBN model should be applicable to safety-related software; the probability of a quality node as a parent node (the prior distributions for the development and V&V quality nodes) should be estimated using data from the population of safety-related software.

- High: Software development by a high-maturity company rigorously following established standards, and implementing additional measures to significantly improve the quality of the software.
- Medium: Software development in which all required activities for safety-related systems are completed.
- Low: Software development in which some required activities for safety-related systems are not completed.

2.2.3 Number of function points

The number of FPs is a measure of software size and complexity, designated as "Size and Complexity Measure" in Fig. 3, and it is modeled as a root node in the BBN model. The definition of FP and its counting rules are governed by the International Function Point Users Group (IFPUG) [11]. Unlike the line of code (LOC) metric, FPs can be used to measure the size of the software in any phase of its life cycle.

2.2.4 Size and complexity

The size and complexity node affects the defect detection probabilities and is assumed to have three states *High*, *Medium*, and *Low* that are defined in terms of its parent node, i.e. the number of FPs. In this study, it is assumed that numbers of FPs less than 100 and greater than 1000 correspond to *Low* and *High*, respectively; otherwise, the size and complexity is *Medium*.

2.2.5 Defect density

The number of defects per function point, designated as "Defect Density (Defects per FP)" in Fig. 3, is assumed to be determined by the quality of the developmental activities, represented as the "Quality of Development" node, and the number of FPs.

2.2.6 Defect detection probability

There are two different defect detection probability nodes representing the probabilities that V&V activities detect and remove the faults in the software introduced in the current phase, and the faults passed from the previous phases. Each defect detection probability node is a child node of the V&V quality node and the software complexity node. In this study, it was assumed that no new defects are introduced during the removal process for the sake of simplicity, and the faults from all precedent phases were assumed to be detected at the same defect detection probability at each SDLC phase.

3 BBN MODEL EVALUTION

3.1 Expert Elicitation for the BBN Model Evaluation

As a part of the BBN model construction and quantification, expert elicitations were performed to quantify the NPTs of the defined BBN nodes. They include elicitations on both the theoretical aspects of the model such as verifying the causal relationship and the adequacy of the BBN model and the quantification of the BBN NPTs. In this study, experts from different organizations with hands-on experience in development and V&V of the NPP software were used to estimate the BBN parameters. After receiving the answers from the experts for each node, distributed node probabilities were used in the NPT modeling instead of using discretized estimates, originally given by the individual experts, to account for the variability among NPP safety-related software and the BBN model parameter uncertainties. The variety of the answers from the experts were aggregated and used to derive the probability distributions which represents the overall data points given by experts for each BBN node.

In this study, in order to reduce the uncertainty from a diversity in experts' opinions for some BBN nodes, handbook data on U.S. software developments and V&V experience [12] as well as testing result from the development experience of two trial systems [13, 14] were used as observations that enable a Bayesian update of the expert inputs. Based on the Bayesian update method considering the conjugate prior family of probability distributions, the NPTs for "Defect density" node and "Defect detection probability at current phase" nodes were updated using the above reference data.

3.2 Fault Size Distribution Estimation

In this study, a fault size distribution (FSD) method [15] is used to convert the number of defects estimated from the BBN model to the software reliability. The operating experience of safety-related protection system software is used in a hierarchical Bayesian analysis [16] to estimate a distribution for the probability of software failure that captures the variability present among the population of safety-related software and used in estimating an FSD for safety-related protection system software. In the analysis, it was assumed that the population variability distribution is lognormally distributed, and uses the operating experience of various safety-related protection system software [17-22].

In order to quantify FSD based on the global operational data on safety-related software, an average software size in FP is required which is not available in the above operating experience study. A typical digital protection software contains an input module that reads sensor measurements, an internal processing logic such as comparison logic to trigger the actuation unit when the measurements exceed their setpoints, and an output unit which produces a trip signal. By following the FP counting rules, one low-level external input, one internal logic file, and one external output are counted. As a result, an estimate of 50 FPs is considered as the typical size of the digital protection software in this study. Table I shows the estimated FSD distribution given the size of 50 FPs and assuming *Medium* quality regarding all attributes for generic safety-related protection system software.

Table I. Estimated FSD distribution for safety-related protection system software

Mean	σ	5th percentile	Median	95th percentile
1.02×10^{-4}	1.34×10^{-3}	0.24×10^{-7}	6.12×10^{-6}	3.03×10^{-4}

3.3 Software Defect Estimation for Typical Digital Protection System

To account for uncertainty in the parameters representing the variability among generic safety-related software, probability distributions were used in modeling the elements of the NPTs instead of using constant point values. The distributions were generated by fitting a probability distribution model over the expert elicitation on the BBN parameter values. The model was evaluated using WinBUGS [23], which uses Markov chain Monte Carlo (MCMC) to solve the Bayesian inference problem posed in the model. Tables II and III show the evaluation results of the BBN parameters in case of the development and V&V quality in all phases being *Medium* quality given the number of FPs of 50.

Table II. BBN model parameter for a typical digital protection system

Phase	Defects introduced in current phase		Detection probability for defects passed from previous phase		Detection probability for defects introduced in current phase	
	Mean	SD	Mean	SD	Mean	SD
Requirement	19.71	35.9	-	-	0.79	0.16
Design	42.61	52.56	0.46	0.26	0.79	0.17
Implementation	49.45	56.96	0.48	0.25	0.84	0.15
Test	19.88	35.25	0.70	0.16	0.73	0.14
Installation/ Checkout	12.63	29.35	0.70	0.19	0.80	0.14

Table III. BBN model evaluation for a typical digital protection system

Phase	Detected defects passed from previous phase		Detected defects introduced in current phase		Defect density		Defects remaining	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Requirement	-	-	15.64	29.28	0.39	0.72	4.07	10.15
Design	1.86	5.40	33.82	43.23	0.85	1.05	11.00	17.05
Implementation	5.25	9.66	41.49	49.03	0.99	1.14	13.71	17.99
Test	9.61	13.16	14.54	26.42	0.40	0.70	9.45	13.08
Installation/ Checkout	6.64	9.77	10.12	23.88	0.25	0.59	5.32	9.12

4 CONCLUSION

In this study, a BBN model is developed that estimates the number of faults in a software program and further converts to the software failure probability, which can be further incorporated into an NPP PRA model. In the model, the SDLC characteristics such as the software development quality and V&V quality, and software-self characteristics using a hierarchical structure. The BBN model at each SDLC phase considers the quality of software development and V&V activities, which affect the number of defects inserted and the number of defects detected in that specific phase, and estimates the number of software defects remaining. The attributes, which are modeled as indicator nodes to the software development and V&V quality, were developed from guidance on safety-related systems.

Since the model structure (e.g. the nodes and their interconnections and parameters) were based on information applicable to safety-related systems, it can be applied to generic safety-related software as well as the software in a specific application. When applied to a target software program, the quality of software development and V&V activities is evaluated against the attributes, software-specific data on the number of faults detected and the software size are estimated and used to Bayesian-update the BBN model to tailor it to the program being analyzed. The quantitative parameters in the BBN model were estimated by effectively utilizing expert opinions from NPP safety software experts and integrating with other available sources of evidence. The NPTs were quantified with the expert opinions expressed as probability distributions for the effective accommodation of the BBN parameter uncertainty. Then, through literature and operating experience, some key parameters in the NPTs were updated in a Bayesian manner.

This study is expected to provide an insight on quantification of the NPP safety graded software reliability based on the proposed framework which can effectively and systematically integrate different kinds of available qualitative and quantitative information regarding safety-related software. As more evidence and observations become available in the future, key parameters in the NPTs can be regression-updated to further reduce the uncertainties related to BBN NPTs and the estimated software reliability.

5 ACKNOWLEDGMENTS

This work was supported by Nuclear Research & Development Program of the National Research Foundation of Korea grant, funded by the Korean government, Ministry of Science, ICT & Future Planning (Grant Code: 2016M2B2A9909815, 2015M2A8A4021648).

6 DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the U.S. Government. Neither the U.S. Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use, or the results of such use, of any information, apparatus, product, or process disclosed in this report, or represents that its use by such third party would not infringe privately owned rights. The views expressed in this paper are not necessarily those of the U.S. Nuclear Regulatory Commission.

7 REFERENCES

1. H. G. Kang, and T. Y. Sung, "An analysis of safety-critical digital systems for risk-informed design," *Reliability Engineering & System Safety,* **78.3**, pp. 307-314 (2002).
2. H. G. Kang, and S. C. Jang, "A quantitative study on risk issues in safety feature control system design in digitalized nuclear power plant," *Journal of nuclear science and technology* **45.8**, pp. 850-858 (2008).
3. T. L. Chu, M. Yue, G. Martinez-Guridi, and J. Lehner, *Development of Quantitative Software Reliability Models for Digital Protection Systems of Nuclear Power Plants*, US Nuclear Regulatory Commission, Washington DC (2013).
4. N. E. Fenton, M. Neil, and J. G. Caballero, "Using Ranked Nodes to Model Qualitative Judgments in Bayesian Networks," *IEEE Transactions on Knowledge and Data Engineering*, **19.10**, (2007).
5. IEEE, *IEEE Standard for System and Software Verification and Validation*, IEEE Std 1012™-2012, IEEE Computer Society (2012).
6. US NRC, (Draft was issued as DG-1267, dated August 2012), *Verification, Validation, Reviews, and Audits for Digital Computer Software Used in Safety Systems of Nuclear Power Plants*
7. IEC, *Nuclear power plants Instrumentation and control systems important to safety Software aspects for computer-based systems performing category A functions*, IEC 60880, International Electrotechnical Commission (2006).
8. Radio Technical Commission for Aeronautics, *Software Considerations in Airborne Systems and Equipment Certification*, RTCA/DO-178C, Prepared by Special Committee 205 (2012).
9. J. D. Lawrence, *Software Reliability and Safety in Nuclear Reactor Protection Systems*, NUREG/CR-6101, US Nuclear Regulatory Commission, Washington DC (1993).

10. US NRC, *Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems*, NUREG-800, Standard Review Plan, Branch Technical Position 7-14, Revision 5 (2007).

11. International Function Point Users Group, http://www.ifpug.org

12. C. Jones, Applied Software Measurement: Global Analysis of Productivity and Quality, Third edition, McGraw-Hill Education (2008).

13. Idaho National Laboratory, *Verification and Validation (V&V) Report for 2A Loop Instrumentation and Operating Control System*, PLN-4681 (2014).

14. H. S. Eom, et al., *Reliability Assessment Method of Reactor Protection System Software by Using V&V based Bayesian Nets*, Korea Atomic Energy Research Institute (2010).

15. B. Littlewood, "Theories of Software Reliability: How good are they and how can they be improved?" *IEEE Transactions on Software Engineering*, **5**, pp. 489-500 (1980).

16. C. L. Atwood, *Handbook of Parameter Estimation for Probabilistic Risk Assessment*, NUREG/CR-6823, US Nuclear Regulatory Commission, Washington DC (2002).

17. W. P. Poore III, et al., *Initial Inventory of Digital I&C Systems Used in Domestic Nuclear Power Plants* (Proprietary), LTR/NRC/RES/2012-002, Oak Ridge National Laboratory (2012).

18. J. H. Bickel, "Risk implications of digital reactor protection system operating experience," *Reliability Engineering and System Safety*, **93**, pp.107–124 (2008).

19. S. A. Eide, et al., *Reliability Study: Westinghouse Reactor Protection System, 1984-1995 - Vol. 2*, NUREG/CR-5500, US Nuclear Regulatory Commission, Washington DC (1999).

20. EPRI, *Estimating Failure Rates in Highly Reliable Digital Systems*, EPRI-1021077, Electric Power Research Institute (2010).

21. KINS, Operational Performance Information System for Nuclear Power Plant (OPIS), http://opis.kins.re.kr/opis?act=KEOBA1100R

22. M. Jockenhövel-Barttfeld, et al., "Modelling Software Failures of Digital I&C in Probabilistic Safety Analyses based on the TELEPERM® XS Operating Experience," *International Journal for Nuclear Power*, **60.3**, pp. 151-158 (2015).

23. D. Spiegelhalter, et al. *WinBUGS user manual* (2003).