Annals of Nuclear Energy 120 (2018) 62–73

Contents lists available at ScienceDirect

Annals of Nuclear Energy

journal homepage: www.elsevier.com/locate/anucene

Development of a Bayesian belief network model for software reliability quantification of digital protection systems in nuclear power plants

Hyun Gook Kang [a,*], Sang Hun Lee [a], Seung Jun Lee [b], Tsong-Lun Chu [c], Athi Varuttamaseni [c], Meng Yue [c], Steve Yang [d], Heung Seop Eom [e], Jaehyun Cho [e], Ming Li [f]

[a] Rensselaer Polytechnic Institute, Troy, NY 12180, USA
[b] Ulsan National Institute of Science and Technology, Ulsan 44919, Republic of Korea
[c] Brookhaven National Laboratory, Upton, NY 11973, USA
[d] NUV Technology LLC, Alpharetta, GA 30022, USA
[e] Korea Atomic Energy Research Institute, Daejeon 34057, Republic of Korea
[f] U.S. Nuclear Regulatory Commission, Washington, DC 20555, USA

ARTICLE INFO

Article history:
Received 10 January 2018
Received in revised form 27 March 2018
Accepted 29 April 2018

Keywords:
Bayesian belief network
Nuclear power plant
Probabilistic risk assessment
Software reliability

ABSTRACT

As the instrumentation and control (I&C) systems in nuclear power plants (NPPs) have been replaced with digital-based systems, 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 software reliability used in NPP digital protection systems. Therefore, a Bayesian belief network (BBN) model which estimates the number of faults in a software considering its software development life cycle (SDLC) is developed in this study. The model structure and parameters are established based on the information applicable to safety-related systems and expert elicitation. The evidence used in the model was collected from three stages of expert elicitation. To assess the feasibility of using BBN in NPP digital protection software reliability quantification, the BBN model was applied to the Integrated Digital Protection System–Reactor Protection System and estimated the number of defects at each SDLC phase and further assessed the software failure probability. The developed BBN model can be employed to estimate the reliability of 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 the potential reactor risk due to software failure.

1. Introduction

Nuclear power plants (NPPs) have traditionally relied upon analog instrumentation and control (I&C) systems for monitoring, control, and protection functions. Due to the functional advantages of digital systems such as fault-tolerance, self-testing, and system diagnosis, in-operation NPPs have begun replacing analog systems with digital technology, while new plant designs fully incorporate digital I&C systems. However, digital systems may have different failure modes and causes than analog systems on account of their unique characteristics such as software; therefore, their incorporation into NPP probabilistic risk assessments (PRAs) presents a challenge to the reliability assessment of the digital I&C systems.

Although software failure in digital I&C systems has been defined differently in literature (ISO, IEC, IEEE, 2010; Lyu, 1996), 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 development life cycle, that results in, or contributes to, the host digital system failing to accomplish its intended function, or to initiate an undesired action. As software failure can significantly affect to the risk of digital I&C system, especially in the case of the NPP protection system (Kang and Sung, 2002; Kang and Jang, 2008), proper software reliability quantification scheme must be developed to quantify the safety software reliability and guarantee NPP safety.

An important insight from previous research on the reliability of digital systems (Chu et al., 2008, 2009) concerns the need to establish a commonly accepted basis for incorporating software behavior into models of digital I&C system reliability that is compatible with existing NPP PRAs. Since existing PRAs are assumed to be developed using traditional static event tree and fault tree methods, software failures need to be captured in these sequences to

* Corresponding author.
E-mail addresses: kangh6@rpi.edu (H.G. Kang), lees35@rpi.edu (S.H. Lee), sjlee420@unist.ac.kr (S.J. Lee), chu@bnl.gov (T.-L. Chu), avarutta@bnl.gov (A. Varuttamaseni), yuemeng@bnl.gov (M. Yue), ehs@kaeri.re.kr (H.S. Eom), chojh@kaeri.re.kr (J. Cho), Ming.Li@nrc.gov (M. Li).

https://doi.org/10.1016/j.anucene.2018.04.045
0306-4549/

address the failures in current PRA frameworks. In other words, software functions in a digital I&C system need to be modeled as event tree top events or fault tree basic events and the failure probability must be quantified based on the quantitative software reliability methods.

Regarding the quantification schemes of safety graded software reliability, a previous study (Chu et al., 2013) has 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. Among the various methods investigated in the previous study, a Bayesian belief network (BBN) method was selected as one of the candidates for the software reliability quantification of the digital protection systems. The BBN method is known as a probabilistic graphical model depicting a set of random variables and their conditional independences via a directed acyclic graph, in which nodes represent random variables with the acyclic graphs not forming any loops (Nielsen and Jensen, 2009). Since the BBN method uses conditional probability tables to represent interdependency among disparate events, it can potentially combine qualitative information, such as quality in carrying out software life cycle activities, with quantitative information, such as test and operational data. This illustrates the potential application of the BBN method in NPP PRAs in terms of NPP safety software reliability quantification. In addition, the BBN methodology has been successfully applied to software reliability assessment both in nuclear industry as well as in non-nuclear applications. Previous BBN approaches that have been performed in nuclear safety field includes the European projects SERENE (Marsh, 1999), IMPRESS (IMPRESS, 1999), OECD Halden Reactor project (Gran and Helminen, 2002, 2001) as well as works by Fenton and Neil (1999), Littlewood and Wright (1995), Johnson et al. (2000), Eom et al. (2009), Bouissou et al. (1999). In these studies, various BBN models were proposed by combining disparate sources of reliability evidence in case of the software used in safety critical systems in order to predict the software product quality or reliability.

In this study, a BBN model is developed that estimates the number of faults in a software program considering the software development life cycle (SDLC) characteristics and further derive the probability of software failure on demand which can be incorporated into a NPP PRA model. In the model, the SDLC characteristics include the software development quality and verification and validation (V&V) quality and the software-self characteristics (e.g., size and complexity), and they are represented using a hierarchical structure. A BBN sub-level model specific to each SDLC phase includes the quality of software development and V&V activities, which affect the number of defects inserted and the number of defects detected or removed in each phase, and estimates the number of remaining defects in a software. To collect evidence for the model parameters and estimate the model, three stages of expert elicitation are employed to overcome the lack of available data on nuclear safety-related software.

To evaluate the feasibility of the proposed BBN framework for assessing the software failure probabilities, the model is applied to an example system, namely the Integrated Digital Protection System–Reactor Protection System (IDiPS-RPS) developed by the Korea Nuclear Instrumentation and Control System (KNICS) project. When applied to a specific software program, experts scored the software attributes and the scores were used to obtain a node probability tables (NPTs) for the development and V&V quality nodes of a target software. In addition, software-specific data including the number of function points (FPs) as a measure of software size and complexity were estimated and used to produce a specific result for a reliability assessment of the software program being analyzed.

2. Model development

This study aims to develop a BBN model that estimates the probability of software failure based on observations of the quality of the development and V&V activities throughout the SDLC. Fig. 1 shows the key steps in model development, which consist of three phases of expert elicitation. First phase of expert elicitation was used to identify the attribute that were carried out to accomplish the functions of each SDLC phase and represents qualities of software development and V&V activities. In addition, the causal relationships represented by the BBN structure was verified. The second phase of expert elicitation were used to quantify the model parameters which are applicable to generic safety-related software. In the third elicitation, software-specific observations were applied to the model and were used to provide input values of nodes specific to a target software application system based on the evidence from a particular software.

The following sections describe the detailed structure of the developed BBN model and the framework to quantify the software failure probability based on the observations of the quality of the development and V&V activities throughout the SDLC phases.

2.1. High-level structure of the BBN model

In the BBN model, five SDLC phases, including Requirements, Design, Implementation, Test, and Installation-and-Checkout phases, are considered and the number of faults remaining in the software at the end of each phase is estimated, as shown in Fig. 2. The model starts with estimating the number of defects remaining in the Requirements phase and tracks the number of defects through all five phases of the SDLC. The number of faults at the end of each phase becomes an input to the BBN model of the next phase. In other words, any remaining defects at the end of each phase are passed on to the next phase. The total number of defects remaining in the software at the end of the last phase (i.e. Installation-and-Checkout) is converted into a software failure probability on-demand. In this study, a sequential model for SDLC is used as a simplification of the actual SDLC that is often iterative.

The model assumes the number of defects remaining in each phase depends on two types of software development activities: development quality and V&V quality, as seen in Fig. 2. In general, the development team carries out the development work, and the V&V activities are performed by an organization that is technically and managerially independent of the development organization. Therefore, the number of remaining defects in each phase is defined as a function of the development quality and the V&V quality, where it is assumed that the development process adds defects and the V&V process removes defects, for each SDLC phase. The following sections provide a summary description of each SDLC phase.

2.1.1. SDLC phase I: Requirements

In the BBN model, the software concept phase and software requirement phase in IEEE Std. 1012 (IEEE, 2012) were merged to be considered as the Requirements phase. The concept activity represents the delineation of a specific implementation solution to resolve the user's problem. During the concept activity, system architecture is selected with system requirements allocated to hardware, software, and user-interface components. The key to developing the concept document is the development or selection of the system architectural design and the system requirements. The purpose of formulating the software requirement specification (SRS) is to satisfy the system requirements and user needs, and to ensure the correctness, consistency, completeness, accuracy, testability, and robustness of the SRS.