Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 22, 2009

V&V based Reusable Bayesian Nets for a Reliability Assessment of Safety-critical Software

Heung-Seop Eom [a*], Gee-Yong Park [a], Seung-Cheol Jang [a]
[a]*Korea Atomic Energy Research Institute, 1045 Daedeok-daero, Yuseong-gu, Daejeon*
* Corresponding author: ehs@kaeri.re.kr

1. Introduction

Bayesian Net (BN) has been used to evaluate a software reliability by considering its development process, because it allows all the relevant evidence to be taken into account. However one of the serious difficulties in the earlier works was that the user had to build a different BN for each development environment such as a software development lifecycle, which is very laborious and time-consuming work [1, 2]. This limits the practical use of BN in the field. One way to solve this problem is the reuse of generalized BN templates which are not restricted to a particular development environment. We applied this idea to the reliability assessment of the safety-critical software for a nuclear power plant with the help of the recent progress in Bayesian Net research such as Object-Oriented BN (OOBN) and Dynamic BN (DBN). We also carried out a case study for the reactor protection system software which was developed as part of the Korea Nuclear Instrumentation & Control System (KNICS).

2. Reusable BN Model

A software development life-cycle (SDLC) basically consists of a requirement phase, a design phase, an implementation (coding) phase, and an integration phase. Most SDLCs employed for the development of the safety-critical software take form of any number and combination of these 4 phases. Therefore the BN model for a particular SDLC can be constructed by (1) the creation of general phase-BN models, (2) customize general phase-BNs to a particular software development project, and (3) by combining and linking these customized phase-BNs into an entire SDLC BN model.

Common activities exist among the development phases. So when we build BNs for different phases in a SDLC it is inevitable to repeat the construction of similar BN subnets. However the creation of a large and repetitive BN is very time-consuming and laborious work. OOBN simplifies this work by creating predefined BN subnets which are called a 'Class' in Object-Oriented methodology terms [3]. The development activities which are common in all software development phases can be converted into these classes (general BN subnets) which will constitute a phase BN. A SDLC is a set of these phases and it defines an overall software project. This can be expressed by using a DBN [4].

3. Construction of a reusable BN for RPS Software

We applied the generalized BN method to the reliability assessment of the RPS software which was developed as a part of KNICS. The SDLC model employed for the RPS software project was a modified spiral model. This spiral SDLC consists of a requirement phase, a design phase, an implementation (coding) phase, and an integration phase. Some of these phases are repeated in the SDLC and finally end up in the system integration phase. We identified the common development activities in the phases and converted them into BN classes which become components of a phase BN later on. We built a phase BN by connecting these classes according to their causal relationship. And then we customized the SDLC BN for our study which was constructed by linking the phase BNs with the help of the DBN. We also carried out a case study and partial results are presented in this paper.

3.1. Component BN for a Phase BN

Identified component BN classes of a phase BN for the RPS software are as follows.

- Residual defects in a previous phase: This class represents the number of remaining defects in the final product developed in the previous development phase.
- Inspection quality in a current phase (Inspection Quality): This represents the number of defects removed by the inspection activities. Traceability-analysis activity was also included in this class.
- Residual defects in a current phase: This class represents the number of remaining defects in the final product developed in the current development phase.
- Conversion process level in a current phase: This class consists of 7 process characteristics appearing in the acceptance criteria of the licensing suitability evaluation. They are consistence, verifiability, unambiguity, traceability, style, correctness, and completeness.
- Documentation level in a current phase: This class consists of 7 process characteristics appearing in the acceptance criteria of the licensing suitability evaluation. They are accuracy, timing, reliability, robustness, security, safety, functionality.
- Defect prediction class: This class models a causal relationship which predicts defect numbers in the current development phase as shown in Fig. 1.

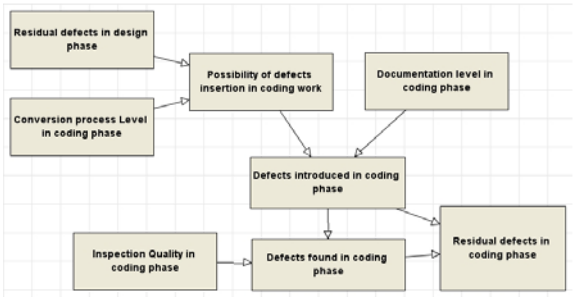
Residual defects in design phase
Possibility of defects insertion in coding work
Documentation level in coding phase
Conversion process Level in coding phase
Defects introduced in coding phase
Residual defects in coding phase
Inspection Quality in coding phase
Defects found in coding phase

Fig. 1. Bayesian Nets for software defect prediction

3.2. The Construction of the phase BN and the SDLC BN

The phase BN represents a single software development phase, for example, a requirement phase or a design phase. In our project, a phase BN consists of 6 component BN classes.

Fig. 2 is the example phase BN model (coding phase BN). Triangular arrow heads represent an input node within a class. Rounded arrow tails represents output nodes. The output node of one object replaces the input node of a connected object. Input nodes act effectively as parameters for a BN class. The inner structure of the BN class is not shown in Fig. 2. The input and the output nodes appeared explicitly in Fig. 2 to represent the interfaces among the classes.

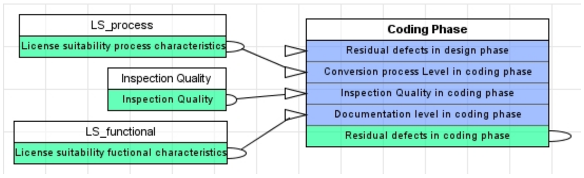
LS_process
License suitability process characteristics
Inspection Quality
Inspection Quality
LS_functional
License suitability fuctional characteristics
Coding Phase
Residual defects in design phase
Conversion process Level in coding phase
Inspection Quality in coding phase
Documentation level in coding phase
Residual defects in coding phase

Fig. 2. Bayesian Nets for Coding Phase

The SDLC BN for the RPS software was modeled by linking a series of phase BNs and their classes. Fig. 3 is the SDLC BN graph for the RPS software in this study. Each phase BN in a SDLC model is connected by a time-indexed variable. Node "Residual defects in a current phase" of each phase BN is connected to "Residual defects in a previous phase" in the following phase BN as shown in Fig. 3.

In Fig. 3, the value in the previous time frame is the input node, and it has no parents in the net. The node representing the value in this time frame is called an output node.

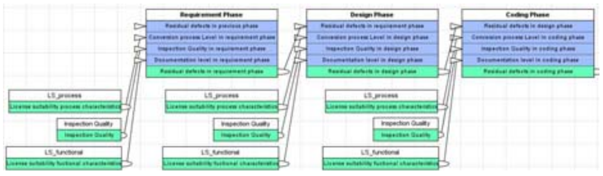

Fig. 3. Software Development Life-Cycle Bayesian Nets for Reactor Protection System software (partial)

3.3 Case study

A case study was carried out for the RPS software. Fig. 4 shows the implementation (coding) phase BN of the Bistable Processor (BP) software and its calculation results. BP software is a part of the RPS system. The inputs for the BN model were mainly derived from the V&V results of the RPS software.

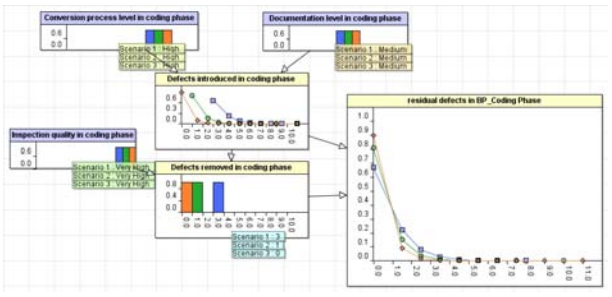

Fig. 4. Bayesian Nets for the coding phase of the BP software and its calculation results

4. Conclusions

In assessing the reliability of a safety critical software by using BNs there has been a serious difficulty that we have to build different BNs for each development environment. To solve this problem we devised a generalized BN construction method which is reusable in many cases. We also carried out a case study for the RPS software which was developed by KNICS. The case study showed that the calculation results of the BNs are useful for a decision making process, and that the method can provide credible information for the applications which require a quantitative reliability of a safety critical software, such as a probabilistic safety assessment (PSA).

REFERENCES

[1] Johnson, G., et al, 2000. Bayesian Belief Network Based Review of Software Design Documents, NIPC & HMIT, 2000.
[2] H.S. Eom, et al, A Study on the Quantitative Reliability Estimation of Safety-Critical Software for Probabilistic Safety Assessment, 4th NIPC & HMIT , 2004.
[3] D, Koller, A. Pfeffer, "Object-Oriented Bayesian Networks", Proceedings of the 13th Conference on Uncertainty in AI, 1997.
[4] O. Bangs, "Top-down construction and repetitive structures representation in Bayesian networks", In Proceedings of the 13th International Florida AI Research Symposium Conference, 2000.