

소프트웨어 신뢰성 모델링 기반 소프트웨어 품질 측정

정혜정

평택대학교 데이터정보학과 교수

The software quality measurement based on software reliability model

Hye-Jung Jung

Professor, Department of Data Information & Statistics, Pyeong-Taek University

요 약 본 연구는 소프트웨어 신뢰성을 측정하기 위해 소프트웨어 신뢰도 측정 모형에 따라 소프트웨어 신뢰도를 측정하는 방법을 제시하려 한다. 본 연구에서 제시한 모형의 형태는 비동질적 포아송 과정의 분포를 이용하였으며, 제시된 모형의 소프트웨어 신뢰도를 측정하는 방안을 제시하였다. 제시된 모형에 따라서 적합한 소프트웨어 신뢰도 성장 모형을 선택하는 방법으로는 소프트웨어 고장 데이터에 따라서 신뢰도 함수의 추정 값에 따른 평균제곱오차를 계산하여 적합한 소프트웨어 신뢰도 함수를 제안하는 방법을 연구하였다. 본 연구에서는 소프트웨어 품질을 측정하기 위한 신뢰도 함수를 제안하기 위하여 모델을 제시하고 고장데이터를 적용하여 추정 값의 오차를 최소화하는 관점에서 소프트웨어 신뢰도 함수를 선택할 수 있는 방안을 제시한 연구로 판단된다.

주제어 : 소프트웨어 신뢰도함수, 테스트 데이터, 융합 기술을 통한 소프트웨어 성장 모형 예측, 소프트웨어 신뢰도 측정 융합정책

Abstract This study proposes a method to measure software reliability according to software reliability measurement model to measure software reliability. The model presented in this study uses the distribution of Non - Homogeneous Poisson Process and presents a measure of the software reliability of the presented model. As a method to select a suitable software reliability growth model according to the presented model, we have studied a method of proposing an appropriate software reliability function by calculating the mean square error according to the estimated value of the reliability function according to the software failure data. In this study, we propose a reliability function to measure the software quality and suggest a method to select the software reliability function from the viewpoint of minimizing the error of the estimation value by applying the failure data.

Key Words : Software reliability function, test data, software growth model prediction through convergence technology, software reliability measurement convergence policy

1. 서론

제4차 산업혁명시대가 도래하면서 소프트웨어의 중요성은 더욱 강조되고 있으며 모든 산업분야에 소프트웨어가 큰 역할을 하고 있다. 이렇게 소프트웨어의 중요성에 대한 비중이 높아지면서 소프트웨어 고장으로 인한 산업

분야의 손실도 크게 나타나고 있다. 개발과정에서 발생할 수 있는 다양한 결함들은 소프트웨어 테스트 과정에서 발견되어야 하며, 또한 안전한 소프트웨어 개발을 위해서는 정해진 시간 안에 소프트웨어 테스트가 정확하게 이루어져야 한다. 일정 시간을 투자해서 소프트웨어를 테스트 할 경우 정확하게 모든 결함을 발견할 수 없다

*This research was supported by Pyeong-Taek University

*Corresponding Author : Hye-Jung Jung(jhj@ptu.ac.kr)

Received January 7, 2019

Accepted April 20, 2019

Revised February 13, 2019

Published April 28, 2019

는 소프트웨어의 원칙을 고려하여 본다면 될 수 있는 한 많은 소프트웨어의 결함을 발견할 수 있도록 하여야 할 것이다. 이와 같은 과정을 통해서 소프트웨어의 남아 있는 결함 정도를 정확하게 파악하기 위해서는 무엇보다도 소프트웨어 신뢰도를 정확하게 측정할 수 있는 소프트웨어 신뢰성 성장 모형에 대한 연구가 기반 되어져야 한다.[1-5] 이러한 이유로 인해서 다양한 소프트웨어 신뢰성에 관련된 소프트웨어 신뢰성 성장 모형에 대한 연구가 진행되어져 왔으며, 현장에서 활용하기 위한 방안도 연구가 진행되고 있다. 그러나 소프트웨어 신뢰도를 측정하기 위해서는 소프트웨어 개발 현장에서 소프트웨어 품질에 대한 인식을 정확히 하고, 개발 소프트웨어에 대해서 정확한 테스트가 이루어져야 한다. 본 연구는 다양하게 제시된 소프트웨어 신뢰성 성장 모형 중에서 비동질적 포아송과정(Non-Homogeneous Poisson Process; NHPP)을 적용한 소프트웨어 신뢰도 성장 모형과 일반적인 소프트웨어 신뢰성 성장 모형을 제시하고 제시된 소프트웨어 신뢰성 성장 모형을 중심으로 평균제품오차를 계산하여 고장 관련 자료에 따라 모형을 선택하는 방법을 제시한다. 비동질적 포아송과정에 관한 소프트웨어 신뢰도 성장 모형에 대한 연구는 Goel & Okumoto에 의한 연구와 Ohba & Yamada 등에 의해서 연구된 S 형태의 비동질적 포아송과정 모형 등이 대표적으로 제시되어 있다. 본 연구에서는 Goel & Okumoto에 의해서 제안된 NHPP 소프트웨어 신뢰성 성장 모형을 소개하고, 신뢰도 함수를 구하려고 한다. 또한 모델에 대한 비교를 위해서 일반적으로 가장 많이 활용되어지고 있는 소프트웨어 신뢰도 성장 모형 중에 하나인 Jelinski & Moranda 모델을 다중 오류 발견이 가능하도록 연구된 변형된 모델을 제시하고 이 모델의 모수 추정을 위해서 베이즈추정법에 의한 모수 추정을 소개하고, 이 모델의 신뢰도 함수를 구하는 방법을 제시하였다. 본 연구에 사용된 NHPP 모델은 포아송과정의 성질을 만족하는 소프트웨어 신뢰도 성장 모형을 제시하였다. 일반적인 소프트웨어 신뢰성 성장 모형은 모형의 특성에 따라 다양한 상황을 고려한 연구가 진행되어져 있으며, 본 연구에서는 다중 오류 수정을 위한 소프트웨어 신뢰도 성장 모형을 제시하였다. 본 연구에서 Goel & Okumoto의 NHPP 모델과 Jelinski & Moranda에 의해서 제시된 모델의 변형된 형태로 소프트웨어 신뢰도 성장 모델에 대한 신뢰도 함수를 구해서 비교 검토하였다. 본 연구의 2장에서는 소프트웨어 신뢰도

성장 모델을 제시하고, 3장에서는 2장에서 제시된 모델을 선정하는 방법에 대해서 제시하고, 4장에서는 고장 데이터의 적용 결과를 제시하였다. 본 연구는 현재 소프트웨어의 신뢰성이 중요함을 인식하고 있는 시점에 다양한 산업군에서의 특징을 고려하여 소프트웨어 신뢰도 함수를 적용하고 적용 결과에 대한 효율성을 측정하기 위한 방안을 제시한 연구이다[6-11].

2. 소프트웨어 신뢰도 성장 모형

2.1 NHPP 소프트웨어 신뢰성 성장 모형

소프트웨어 신뢰성을 측정하기 위한 소프트웨어 신뢰도 성장 모델에 있어서 비동질적 포아송과정(Non-Homogeneous Poisson Process : NHPP)의 모델은 아래에 제시된 기본적인 포아송 과정의 성질을 만족해야 한다[4-9].

- (1) $N(0) = 0$
- (2) $\{N(t), t \geq 0\}$ has independent increment.
- (3) $P\{N(t+h) - N(t) = 1\} = \lambda(t) + o(h)$
- (4) $P\{N(t+h) - N(t) \geq 2\} = o(h)$

Goel & Okumoto는 소프트웨어 결함에 대한 신뢰도 함수를 평균값 함수를 가지는 NHPP 소프트웨어 신뢰성 성장 모델을 제안하였다. 여기에서 $N(t)$ 는 t 시간까지 소프트웨어를 시험하면서 발견된 고장수를 의미한다. 또한 $m(t)$ 는 NHPP 소프트웨어 신뢰도 성장 모형의 평균값 함수로 정의한다. 또한 $\lambda(t)$ 는 순간 결함 발생 비율로 정의한다면 비동질적 포아송 과정 (NHPP) 모형의 확률밀도함수에 대한 정의는 아래의 식 (1)과 같다.

$$P\{N(t) = n\} = \frac{m(t)^n \exp(-m(t))}{n!}, \quad n = 0, 1, 2, \dots, \infty \quad (1)$$

또한 순간 결함 발생 비율 $\lambda(t)$ 는 아래의 식 (2)와 같다.

$$\lambda(t) = \lim_{\delta t \rightarrow 0} \frac{P\{N(t+\delta t) - N(t) > 0\}}{\delta t} \quad (2)$$

순간 결함 발생 밀도 함수 $\lambda(t)$ 를 이용하여 평균값 함수를 구하는 식은 아래 (3)과 같다.

$$m(t) = \int_0^t \lambda(a) da, \quad \lambda(t) = \frac{dm(t)}{dt} \quad (3)$$

일반적으로 평균값 함수는 소프트웨어의 시간 경과에 따른 특성을 고려한다면 시간의 경과에 따라서 비감소함수의 형태를 지니는 함수를 선택하므로, NHPP의 기본적인 형태의 소프트웨어 신뢰성 성장 모형에 적용할 수 있다. 이러한 성질을 고려한 Goel & Okumoto가 제안한 NHPP 소프트웨어 신뢰성 성장 모형의 평균값 함수는 아래 식 (4)와 같다.

$$m(t) = a(1 - e^{-bt}), \quad a > 0, \quad b > 0$$

a : 시험을 통해서 최종적으로 발견될 오류총 수
 b : 소프트웨어 결함 발생 비율

$$\lambda(t) = \frac{dm(t)}{dt} = ae^{-bt} \quad (4)$$

Goel & Okumoto 모델에서 s 시간에 마지막 결함이 발견되었다는 조건하에서 t 시간에 소프트웨어 신뢰성을 측정하기 위한 모형은 아래의 식 (5)와 같이 정의할 수 있다.

$$R(t|s) = e(-a(e^{-bs} - e^{-b(s+t)})) \quad (5)$$

본 연구에서 제안된 NHPP 소프트웨어 신뢰성 성장 모형의 우도함수는 아래의 식 (6)과 같다.

$$L(x_1, x_2, \dots, x_n) = (\prod_{i=1}^n \lambda(x_i)) \exp(-m(x_n)) \quad (6)$$

(6)식의 우도함수에 자연로그를 취하여 구하고자 하는 모수에 대해서 편미분을 취하면 본 연구에서 고려된 소프트웨어 신뢰성 성장 모형의 모수를 아래의 식 (7)과 같이 구할 수 있다.

$$\hat{a} = \frac{n}{1 - e^{-bs_n}}$$

$$\sum_{i=1}^n \frac{n_i (s_i e^{-bs_{i-1}} - s_{i-1} e^{-bs_i})}{e^{-bs_{i-1}} - e^{-bs_i}} - \frac{ns_n e^{-ns_n}}{1 - e^{-bs_n}} = 0 \quad (7)$$

본 연구에서는 비동질적 포아송과정 소프트웨어 신뢰성 성장 모형으로 Goel & Okumoto를 적용하여 실제 소프트웨어 운영 중에 발생한 소프트웨어 고장데이터를 적용하여 신뢰성을 구하고 모델의 적합성을 확인하려 한다 [12,13,14,15].

2.2 일반적 소프트웨어 신뢰성 성장 모형

본 절에서는 일반적으로 가장 많이 활용되어지고 있는 소프트웨어 신뢰성 성장 모형을 이용하여 신뢰도 함수를 구하여 보려 한다. 관찰된 실패 사이의 시간을 $\tilde{t} = (t_1, t_2, \dots, t_n)$ 이라 하면 소프트웨어 신뢰도 함수는 아래 식 (8)과 같이 정의할 수 있다.

$$f(\tilde{t}) = (N_0 - i + 1) \phi e^{-((N_0 - i + 1)\phi t_i)}$$

N_0 : 총 결함 수
 ϕ : 결함 발생 비율

$$(8)$$

제시된 (8)번 식에서 N_0 가 포아송 확률 변수인 경우 아래의 식 (9)와 같은 결함확률밀도함수를 구할 수 있다.

$$f(\tilde{t}, N_0 = k) = (\prod_{i=1}^r (k - i + 1) \phi e^{-(k - i + 1)\phi t_i}) \times \frac{e^{-\lambda} \lambda^k}{k!} \quad (9)$$

여기에서 k 를 r 에서 ∞ 까지 합하면 \tilde{t} 의 주변확률밀도 함수를 유도할 수 있다. 소프트웨어 성장 모형을 구하기 위한 주변확률밀도 함수는 아래의 식 (10)과 같다.

$$f(\tilde{t}) = \sum_{k=r}^{\infty} (\prod_{i=1}^r (k - i + 1) \phi^r e^{-\sum_{i=1}^r (k - i + 1)\phi t_i}) \times \frac{e^{-\lambda} \lambda^k}{k!} \quad (10)$$

위의 (10)식을 이용하여 베이지 추정 방법을 이용해서 모수를 추정하기 위해 사후분포를 유도하면 식 (11)과 같다.

$$\begin{aligned}
P(N_0 = k|\tilde{t}) &= \frac{f(\tilde{t}, N_0 = k)}{f(\tilde{t})} \\
&= \frac{\prod_{i=1}^r (k-i+1) \phi^r e^{-\sum_{i=1}^r (k-i+1) \phi t_i} \frac{e^{-\lambda} \lambda^k}{k!}}{\sum_{K=r}^{\infty} (\prod_{i=1}^r (K-i+1) \phi^r e^{-\sum_{i=1}^r (K-i+1) \phi t_i} \frac{e^{-\lambda} \lambda^K}{K!})} \\
&= \frac{\prod_{i=1}^r (k-i+1) e^{-\sum_{i=1}^r (k-i+1) \phi t_i} \frac{\lambda^k}{k!}}{\sum_{k=r}^{\infty} (\prod_{i=1}^r (k-i+1) e^{-\sum_{i=1}^r (k-i+1) \phi t_i} \frac{\lambda^k}{k!})} \quad (11)
\end{aligned}$$

위의 식 (11)의 사후분포를 이용하여 소프트웨어에 남아 있는 최종 결함의 수 N_0 값의 베이즈추정치를 구하면 아래의 식 (12)와 같다.

$$\begin{aligned}
\hat{N}_0 &= \sum_{k=r}^{\infty} k p(N_0 = k|\tilde{t}) \\
&= \frac{\sum_{k=r}^{\infty} k R_r(k) e_r(k) \frac{\lambda^k}{k!}}{\sum_{l=r}^{\infty} R_r(l) e_r(l) \frac{\lambda^l}{l!}} \quad (12)
\end{aligned}$$

(12) 식에서 $R_r(k), e_r(k)$ 는 아래의 식 (13)과 같다.

$$\begin{aligned}
R_r(k) &= \prod_{i=1}^r (k-i+1) \\
e_r(k) &= \exp(-M_r(k)) \quad (13)
\end{aligned}$$

위의 N_0 에 대한 추정치를 이용하면 r 개의 오류가 수정된 뒤에 임의의 시간 t 동안에 소프트웨어 시스템의 신뢰도 함수를 계산할 수 있다. 즉, 이 경우 소프트웨어 신뢰도 함수의 신뢰도를 측정하면 아래의 식 (14)와 같다.

$$\begin{aligned}
R(t) &= F(T_{r+1} > t|\tilde{t}) = 1 - F(T_{r+1} < t|\tilde{t}) \\
&= \frac{\sum_{k=r+1}^{\infty} R_r(k) e_r(k) \frac{\lambda^k}{k!} e^{-(k-r)\phi t}}{c} \quad (14)
\end{aligned}$$

여기에서 c 는

$$c = \left(\sum_{l=r}^{\infty} R_r(l) e_r(l) \frac{\lambda^l e^{-\lambda}}{l!} \right)$$

이다.

소프트웨어 신뢰성을 측정하기 위해서 다양한 소프트웨어 신뢰성 성장 모델이 제안되어 있으나 본 연구에서는 두 개의 모델을 이용해서 소프트웨어 신뢰도 함수를 측정하는 방안을 제시한다.

3. 모형의 선정기준

소프트웨어의 신뢰성을 측정하기 위해서 다양한 소프트웨어 신뢰성 성장 모델이 제시되고 연구되어져 있다. 소프트웨어 테스트 동안에 발견된 고장 데이터를 이용해서 소프트웨어 신뢰성을 추정 할 경우 가장 효율성이 좋은 모델을 선정하는 기준을 마련하는 것은 쉽지 않다. 본 연구에서는 추정된 소프트웨어 모델에 대해서 평균제곱오차를 계산하고 계산된 결과를 이용해서 모델을 선택할 수 있는 방안을 제시하려 한다. 소프트웨어 테스트 과정에서 수집된 고장데이터를 이용해서 모수를 추정하고, 추정된 모수에 대해서 신뢰성을 평가할 수 있도록 하기 위해서는 모형의 선정기준을 정확히 설정해야 한다. 본 연구에서는 제시된 모델에 대해서 평균제곱오차를 이용해서 모델을 선정하도록 한다. 본 연구에서 제시하는 평균제곱오차(Mean Square Error :MSE)는 아래의 식 (15)와 같다.

$$MSE = \frac{\sum_{i=1}^n (m(x_i) - \widehat{m}(x_i))^2}{(n-k)} \quad (15)$$

평균제곱오차(MSE)의 값이 작으면 모델이 정확성이 높은 것으로 판정할 수 있다. 소프트웨어에 있어 신뢰성을 측정하는 데에는 데이터의 특징을 고려하여 모델을 선정해야 하는 어려움이 있다.

4. 제안된 모델에 대한 신뢰도

본 연구에서 제시한 소프트웨어 신뢰성 성장 모델을 이용하여 신뢰성을 검정하기 위해서 고장시간 자료를 이용하여 비교 분석 하였다. 본 연구를 위해서 사용된 자료는 아래의 Table 1과 같다.

Table 1. Number of Errors Each Test Step

TestStep	Fault	Test Step	Fault	Test Step	Fault
1	6	11	5	21	3
2	7	12	4	22	4
3	7	13	6	23	3
4	5	14	5	24	3
5	5	15	4	25	4
6	4	16	3	26	3
7	6	17	4	27	2
8	7	18	3	28	2
9	5	19	4	29	1
10	6	20	4	30	1

Table 2. MSE of Test Step

Test Step	MSE of 2.1	Test Step	MSE of 2.2
5	0.45	5	0.52
10	0.34	10	0.45
15	0.29	15	0.34
20	0.25	20	0.30

위의 모의시험 결과 단계가 높아질수록 평균제곱오차가 감소하고 있는 것을 알 수 있다. 또한 본 자료는 오류가 단계가 증가하면서 단조 감소하는 함수의 형태이므로 2.1절에서 제안한 NHPP 모형이 2.2절에서 제안한 Jelinski & Moranda의 소프트웨어 신뢰성 성장 모형에 비해서 평균제곱오차(MSE)가 적다는 것을 알 수 있다. 소프트웨어 신뢰도 성장 모형의 선택은 오류의 형태에 따라서 정의할 수 있으며 발견된 오류의 수가 단조 감소하는 경우 선형적 소프트웨어 신뢰성 성장 모형보다는 NHPP 모형이 적합하다는 것을 알 수 있었다.

5. 결론

본 연구에서 소프트웨어 신뢰성 성장 모형을 적용하여 신뢰성을 정확히 추정하고 모수를 추정하기 위해서 어떤 모델을 적용하는 것이 가장 적합한지를 평균오차제곱을 이용하여 제시하였다.

소프트웨어 품질을 측정하는데 있어 국제 표준에서 제시하고 있는 품질 특성을 고려하여 본다면 신뢰성에 대한 평가가 항상 중요한 요인으로 작용하고 있다. 모수를 추정하고 추정된 모수를 이용하여 정확한 모델을 적용했는지에 대한 기준을 마련해서 소프트웨어 신뢰성에 대한 정확한 평가가 이루어질 수 있도록 하여야 할 것이다.

본 연구에서는 소프트웨어 고장데이터에 대해서 신뢰성 모델에 적용하여 모델의 정확성을 평가할 수 있도록 제시한 연구로 앞으로 다양한 경우를 고려하여 소프트웨어 신뢰성을 평가할 수 있는 방안을 제시할 수 있도록 할 것이다.

또한 이와 같은 결과는 국제 표준 ISO/IEC 25023에서 제시하고 있는 8가지 품질 특성 중 신뢰성을 측정할 수 있는 기준 연구로 활용되어질 수 있을 것이다.

신뢰성에 대한 측정은 남아 있는 잔존 결함에 대한 정보와 양도 시간에 대한 예측이 가능하므로 다양한 신뢰도 함수를 고려한 신뢰도 측정의 방안 연구가 중요한 것으로 보여진다.

REFERENCES

- [1] ISO/IEC 9126-2. (2003). *Software Engineering - Product Quality -Part 2: External metrics.*
- [2] ISO/IEC 25010. (2011). *System and software engineering-System and software Quality Requirements and Evaluation(SQuaRE) -System and software quality model.*
- [3] ISO/IEC 25023. (2015). *System and software engineering-System and software Quality Requirements and Evaluation(SQuaRE) - Measurement of system and software product quality.*
- [4] ISO/IEC 25000, (2005). *System and software engineering: System and software Quality Requirements and Evaluation (SQuaRE) -Guide to SQuaRE.*
- [5] ISO/IEC 25010, (2011). *System and software engineering: System and software Quality Requirements and Evaluation(SQuaRE)-System and software quality models.*
- [6] Z. Jelinski & P. B. Moranda. (1972). *Software reliability research. In Statistical Computer Performance Evaluation*, ED.W. Freiburger, Academic Press, New York, 465-497.
- [7] A. L. Goel & K. Okumoto. (1979). Time dependent error-detection rate model for software reliability and other performance measures. *IEEE Trans. Reliability*, 28(3), 206-211.
- [8] H. J. Jung. (2014). The Quantity Data Estimation for Software Quality Testing, *The Journal of Digital Police & Management*, Vol. 12. No 1, 12(1).
- [9] H. J. Jung. (2003). Performance Evaluation of Software Reliability Growth Model Using Plot of Fault Data,

- Korea Information Processing Society, 10(5), 829-836*
- [10] H. J. Jung. (2018) , Reliability measurement applied to software quality assessment metrics, *The Journal of Multimedia, 21(1), 373-376.*
- [11] H. J. Jung. (2017). The Survey for Software Quality Testing, *The Journal of Multimedia, 20(2), 292-295.*
- [12] J. E. Sin(2012), 'Applied SPSS Statistics Analysis', Kyony Moon.
- [13] H. J. Jung.(2016). Software Quality Testing on the basis of the International Standard ISO/IEC 25023, *Journal of the Korea Convergence Society, 7(6), 35-41.*
- [14] Y. W. Kim. (2014). A study on Convergent & Adaptive Quality Analysis using DQnA model. *Journal of the Korea Convergence Society, 5(4), 21-25.*
- [15] W. I. Keon(2012), *Practical Software Testing Foundation*, STA.

정혜정(Hye-Jung Jung)

[정회원]



- 1988년 : 경북대학교통계학과 조기 졸업(이학사)
- 1991년 : 경북대학교대학원통계학과 졸업(이학석사)
- 2004년 : 경북대학교 대학원 통계학과 졸업(이학박사)
- 2008년7월 ~ 2009년 6월 : UNLV 교환교수
- 1995년 ~ 현재 : 평택대학교 데이터정보학과 교수
- 2001년 ~ 현재 : 금융결제원자문위원, 평택시 노사협의회자문위원, ISO/IEC JTC1/ SC7위원, ISO/IEC SC 34 위원, 소프트 품질인증위원, 국가기술표준원 정보기술위원, 국가기술표준원 제품안전자문위원, 멀티미디어학회이사, 경기정보산업협회 편집위원장, 한국융합학회 이사 등
- 관심분야 : 소프트웨어신뢰성공학, 소프트웨어품질평가, 소프트웨어 품질평가에 대한 표준화, 소프트웨어 용어표준화, 소프트웨어테스팅 등
- E-Mail : jhj@ptu.ac.kr