논문 2020-2-8 http://dx.doi.org/10.29056/jsav.2020.12.08

오류 수정 시간을 고려한 소프트웨어 최적 출시 시점 결정 연구

안철훈*†

A Study on Determining the Optimal Time to Launch of Software Considering Error Correction Time

Cheol-Hoon Ahn*†

요 약

본 논문에서는 오류 수정 난이도를 나타내는 지표인 오류 수정 시간을 사용하여 소프트웨어 최적 출시 시점 결정 문제를 연구하였다. 특히 기존의 오류 수정 시간을 고려한 소프트웨어 신뢰도 성장 모델에서 오류 발견 시간과 수정 시간이 독립적이라는 가정을 수정하고, 오류 발견 시간과 수정 시간의 상관관계를 표현할 수 있는 일반적인 프레임워크 모델을 설정하여 소프트웨어 최적 출시 시점을 결정해 보고자 하였다. 그 결과 테스트 초기에 수정 시간이 걸리는 오류를 발견하는 것이 경제적인 관점에서 중요하다는 것을 알 수 있었다. 최적의 소프트웨어 출시 시점을 결정하는 데에 있어서 오류발견시간과 오류수정시간의 상관관계를 분석하는 것이 매우 중요하다는 결론을 얻을 수 있었다.

Abstract

In this paper, the problem of determining the optimal time to market of software was studied using error correction time, an indicator of error correction difficulty. In particular, it was intended to modify the assumption that error detection time and correction time are independent in the software reliability growth model considering the existing error correction time, and to establish a general framework model that expresses the correlation between error detection time and correction time to determine when the software will be released. The results showed that it was important from an economic perspective to detect errors that took time to correct early in the test. It was concluded that it was very important to analyze the correlation between error detection time and error correction time in determining when to release the optimal software.

한글키워드 : 소프트웨어 출시, 오류 수정 시간, 오류 발견 시간, 소프트웨어 신뢰도 성장 모델, 소프트웨어 최적 출시 시점

keywords : Software Release, Error Correction Time, Error Detection Time, Software Reliability Growth Model, Optimal Time to Launch of Software

* 경복대학교 첨단융합학부 소프트웨어융합과
† 교신저자: 안철훈(email: chahn@kbu.ac.kr)
접수일자: 2020.08.31. 심사완료: 2020.11.06.
게재확정: 2020.12.21.

www.kci.go.kr

1. 서 론

소프트웨어 신뢰성은 소프트웨어 품질에 있어서 가장 중요한 지표 중 하나이다. 소프트웨어 신뢰성을 정량적으로 표현한다면 일정 시점에서 오류가 발견되지 않는 확률로써 나타낼 수 있다. 지금까지 소프트웨어 신뢰성을 평가하기 위한 확률 모델로써 소프트웨어 신뢰도 성장모델이 제안되어 소프트웨어 개발 관리에서 다양하게 이용되고 있다.

소프트웨어의 신뢰성을 예측 하는데 가장 대표적인 방법이라고 할 수 있는 소프트웨어 신뢰도 성장 모델 (Software Reliability Growth Model)은 소프트웨어의 고장에 대한 원인을 찾아 개선을 하거나 새로운 디자인, 새로운 기술을 도입하게 되므로 시간이 지남에 따라 신뢰도가 증가하는 것을 기대할 수 있는 모델이다. 이 모델은 소프트웨어 테스팅 단계에서 소프트웨어 고장수와 고장 간격을 근거로 소프트웨어 고장현상을 모델화 하였으며 고장발생간격시간, 소프트웨어 신뢰도 및 고장률 등의 신뢰성 평가측도들 이 추정되어 미래에 어떤 방식으로 고장이 나타날 것인가를 예측 할 수 있다[1][4].

기존의 소프트웨어 신뢰도 성장 모델은 테스트에서 발견되는 오류의 갯수에 주목한 모델이다. 한편, 현재의 소프트웨어 개발에서는 오픈 소스 소프트웨어(OSS: Open Source Software)에서 볼 수 있듯이 지속적인 인티그레이션(Integration)에 의해 항상 소프트웨어를 이용할 수 있는 상태를 유지하는 것이 주류가 되고 있다. 이런 개발 형태로는 각 릴리즈(Release)에서 발견한 오류가 모두 수정되어 있다고는 할 수 없다. 즉, 오류 발견뿐만 아니라 오류 수정에 이르기 까지 오류의 라이프사이클을 고려한 신뢰성 평가가 필요하다.

시스템 오류를 수정하기 위한 소프트웨어 신뢰도 모델은 다수 논의된 바 있다. Schneidewind[5], Xie[6], Gokhale[7][8]등은 수정된 오류 수를 누적하여 표현하는 확률 모델을 구축하고 있다. Xie[6]는 오류발견과 오류 수정을 통합하는 모델을 제시하였다. 위의 모델은 오류발견과 오류 수정의 누적 수가 모두 비동질성 포아송 과정을 따른다는 특징이 있다. 또, Okamura와 Dohi[9]는 위에서 설명한 모든 모델을 포함하는 오류 수정 프레임워크 모델을 제안하고 있다. 즉 오류발견시간과 오류수정시간 의 상관관계를 고려한 프레임워크 모델을 제안하고 있다. 여기서 말하는 오류발견시간과 오류수정시간 간의 상관관계는 테스트의 초기 단계에서 발견된 오류일수록 수정시간이 짧다(혹은 길다)라든지 테스트 마지막 단계에서 발견된 오류일수록 수정시간이 짧다(혹은 길다)라는 관계를 정량적으로 나타내며, 이것은 어떤 모듈을 먼저 테스트할 것인지 결정하는 테스트 전략과 밀접하게 관련되어 있다. 여기에서는 발견되지 않은 오류 수뿐만 아니라 발견되었으나 수정되지 않은 버그 수를 고려한 신뢰성 지표를 제안하고 있다.

본 논문에서는 위에서 서술한 오류 발견 시간과 오류 수정 시간의 상관관계를 고려한 소프트웨어 출시 시점 결정 문제를 다룬다. 비즈니스 관점에서 소프트웨어 개발과정에서 언제 테스트를 종료하고 출시할 것인가를 결정하는 것은 매우 중요하다. 그렇기 때문에 오래 전부터 소프트웨어 신뢰성 모델에 기초한 수학적인 접근이 많이 행해지고 있다.

Okumoto와 Goel[4]은 오류 발견 시간만을 고려한 지수형 소프트웨어 신뢰성 모델을 이용하여 테스트 비용, 테스트 기간 중의 오류 수정 비용, 소프트웨어 출시 후의 오류 수정 비용의 기대치를 산출하고, 그 총비용을 최소로 하는 최적의 소프트웨어 출시 시점을 결정했다.

이후에도 여러 요인을 도입한 소프트웨어 최

www.kci.go.kr

적 출시 시점이 논의되었으나, 대부분이 오류 발견 시간에 초점이 맞춰진 것이었다. 즉, 오류가 발견된 후 즉시 수정될 것이라는 가정으로 하고 있다. 그렇지만 실제 소프트웨어 개발에서는 여러 가지 오류가 존재하고, 수정 난이도도 각각 다르다. 오류 중에는 원인을 밝혀내기가 어려운 것도 있다. 오류 발견 시간에만 초점을 맞춘 소프트웨어 출시 시점 결정은 오류 수정 난이도에 관계없이 일률적인 오류 수정 비용을 부과하는 일이 많아 오류 수정이 출시 시점 결정에 주는 영향에 대하여 밝혀지지 않았다. 이에 본 논문에서는 오류 수정 시간을 고려하여 소프트웨어 최적 출시 시점을 알아내고자 한다. 특히, 오류 발견 시간과 오류 수정 시간과의 상관관계가 최적의 소프트웨어 출시 시점 결정에 미치는 영향에 대해 연구하고, 비즈니스적인 관점에서 어떤 테스트 전략이 좋은지에 대해 알아본다.

2. 소프트웨어 신뢰도 성장 모델

2.1 오류 발견 시간에 따른 소프트웨어 신뢰도 성장 모델

이전의 소프트웨어 신뢰도 성장 모델은 개발 과정에서의 오류 발전 누적 갯수를 확률로 표현하여 현시점에서의 잔존한 오류 갯수나 장래의 오류 발견에 관한 평가 및 예측을 하는 모델이다.

비동질성 포아송과정(NHPP:Non-Homogeneous Poisson Process)에 기반한 소프트웨어 신뢰도 성장 모델은 다음과 같은 가설로부터 구축된다 [2][3].

- *가설 1 : 테스트 전에 소프트웨어 내에 잠재하는 오류 수 M은 평균 ω의 포아송 분포를 따른다.*
- *가설 2 : M개의 오류 발견 시간은 독립적이며 동일하게 분포하는 확률 변수 $\{X_1,\ldots,X_m\}$로 나 타나며, 그 누적 분포 함수를 F(t)로 한다.*

테스트 시작 시간을 t=0으로 하고, 시간 t까지의 오류 발견 누적 갯수를 $\{N(t),t\geq 0\}$이라고 하면, 상기의 가정으로부터 아래 식1과 같은 확률 함수를 얻을 수 있다.

$$P(N(t)=n)=\sum_{m=n}^{\infty}P(N(t)=n|M=m)P(M=m)$$
$$=\sum_{m=n}^{\infty}\binom{m}{n}F(t)^n(1-F(t))^{m-n}\frac{\omega^m}{m!}e^{-\omega}$$
$$=\frac{(\omega F(t))^n}{n!}e^{-\omega F(t)}. \quad \text{– 식1}$$

위의 식에서 오류의 발견 누적 개수는 E[N(t)] = ωF(t)가 된다.

소프트웨어 신뢰도 성장 모델에서 테스트 시간 t부터 t+u까지 오류가 발견될 확률은 아래 식2와 같다.

$$R(u|t)=\exp\left(-\omega(F(t+u)-F(t))\right) \quad \text{– 식2}$$

2.2 오류 발견 시간과 수정 시간을 고려한 모델

Okamura와 Dohi[9]는 오류 발견 시간 뿐만 아니라 오류 수정 시간을 고려한 모델을 제안하고 있다. 먼저, 오류 수정 시간을 다루기 위해 다음과 같은 가설을 설정한다.

- *가설 1 : 테스트 전 소프트웨어 내에 잠재하는 오류 개수 M은 평균 ω의 포아송 분포를 따른다.*
- *가설 2 : M개의 오류 발견 시간 X 및 수정 시간(발견에서 수정 완료까지의 시간)*

www.kci.go.kr

S는 독립적이며 동일하게 분포하는 이변량 확률 변수 $\{(X_1, S_1) \cdots, (X_M, S_M)\}$로 나타나고, 그 동시 분포 함수를 F(x,s)로 한다.

이전 모델과의 본질적인 차이는 오류 발견 시간과 수정 시간이 이변량 분포로 표현되었다는 점에 있다. 즉 오류수정시간이 오류가 발견되는 시간에 따라 달라질 수 있다는 것이다. 이것은 테스트의 초기 단계에서 발견된 오류일수록 수정시간이 짧다(혹은 길다), 또는 테스트 마지막 단계에서 발견되는 오류일수록 수정시간이 짧다(혹은 길다)라는 상관관계를 나타내게 된다.

이변량 분포 F(x,s)에서 다음의 식을 도출할 수 있다.

- $F_{UC}(t)$: 시간 t까지 오류가 발견되어 수정될 확률
- $F_C(t)$: 시간 t까지 오류가 발견되었으나 수정되지 않았을 확률
- $F_D(t) = F_{UC}(t)+F_C(t)$: 시간t까지 오류가 발견될 확률

위의 것을 동시 밀도 함수 $f(x,s) = \partial^2 F(x,s)/\partial x \partial s$ 를 사용하면 아래 식3,4,5를 구할 수 있다.

$$F_{UC}(t) = \int_0^t \int_0^{t-x} f(x,s)dsdx, \quad - \text{식3}$$

$$F_C(t) = \int_0^t \int_{t-x}^{\infty} f(x,s)dsdx, \quad - \text{식4}$$

$$F_D(t) = \int_0^t \int_0^{\infty} f(x,s)dsdx \quad - \text{식5}$$

$N_D(t)$를 t까지 발견된 오류의 누적 갯수라고 하고, $N_C(t)$를 t까지 수정된 오류의 누적 갯수라고 하면, 버그 수 M이 포아송 분포에 따르기 때문에 $N_D(t)$와 $N_C(t)$의 동시확률은 아래 식6과 같다.

$$P(N_D(t) = d, N_C(t) = c) = \frac{(\omega F_{UC}(t))^{d-c}(\omega F_C(t))^c}{(d-c)!c!} e^{-\omega F_D(t)} \quad - \text{식6}$$

위에서 살펴본 바와 같이 어떤 구간에서 오류가 발견될 확률 뿐만 아니라 특정 시간까지 발견된 오류가 모두 수정되었을 확률을 산출하고 있다. 이것은 OSS의 기반을 둔 소프트웨어 개발에서 기존의 소프트웨어 신뢰도보다도 유익한 척도가 되고 있다.

3. 소프트웨어 최적 출시 시점

3.1 기존의 소프트웨어 최적 출시 시점

Okumoto와 Goel[4]은 테스트 비용을 고려한 소프트웨어 최적 출시 시점을 다음과 같이 정의하고 있다.

- c_1 : 단위시간당 테스트 비용
- c_2 : 테스트 과정에서 발견된 오류를 수정하는 데 필요한 단위 갯수당 비용
- c_3 : 출시 후 운용 과정에서 발견된 오류를 수정하기 위한 단위 갯수당 비용

일반적으로 오류 수정비용은 테스트 과정 보다 출시 후의 운용 과정에서 높기 때문에 $c_3 > c_2$라고 가정할 수 있다. 현재 시간 t까지 발견된 오류 갯수를 N(T)라고 하고, 소프트웨어의 공급 중단 시간을 T_{LC}로 한다.

www.kci.go.kr

이 때, 시점 T에서 소프트웨어를 발매했을 때의 총 기대비용 C(T)는 아래 식7과 같다.

$$C(T) = c_1 T + c_2 E[N(T)] + c_3 (E[N(T_{LC}) - E[N(T)]). \quad - \text{식7}$$

E[N(T)]은 시간 T까지 발견된 오류의 총갯수 기대치이며, 오류 발견 시간 분포 F(T) 및 초기 오류 수의 기대치 ω를 이용하면 아래 식8을 도출할 수 있다.

$$C(T) = c_1 T + c_2 \omega F(T) + c_3 \omega \{F(T_{LC}) - F_D(T)\} \quad - \text{식8}$$

이 때 총 기대비용 C(T)를 최소화하는 최적 출시 시점 T를 구하는 것은 아래 식9로 정리할 수 있다.

$$\min_{0 \le T \le T_{LC}} C(T). \quad - \text{식9}$$

3.2 오류 수정 시간을 고려한 소프트웨어 최적 출시 시점

이번에는 오류 수정 시간을 고려한 모델을 근거로 하여 소프트웨어 최적 출시 시점을 구하려고 한다. 위의 출시 시점을 구하는 것과 마찬가지로 아래의 비용을 가정한다. 단, 테스트 과정 및 운용 단계에서의 오류 수정 비용이 단일 오류당 비용이 아니라 수정 시간에 비례한 비용으로 변경하였다.

- c_1 : 테스트 과정에서 발생하는 단위 시간당 비용
- c'_2 : 테스트 과정에서 발견된 오류를 수정하는 데 필요한 단위 시간당 비용
- c'_3 : 운영 단계에서 발견된 오류를 수정하는 데 필요한 단위 시간당 비용

소프트웨어의 이용기간을 T_{LC}라고 하고 출시 시점을 T라고 하면 총 기대비용 C(T)는 아래 식10과 같이 구할 수 있다.

$$C(T) = c_1 T + c'_2 W(T) + c'_3 (W(T_L) - W(T)). \quad - \text{식10}$$

이제 $\{(X_1, S_1) \ldots, (X_{N_D(T)}, S_{N_D(T)})\}$를 시간 T까지 발견된 오류 발견 시간과 수정 시간의 열로 하여 T까지 발견된 N_D(T)의 갯수가 오류 수정 시간 $S_1, \ldots, S_{N_D(T)}$의 합이 되어지기 때문에 아래의 식11을 구할 수 있다.

$$\begin{aligned} W(T) &= \mathrm{E}\left[\sum_{i=1}^{N_D(T)} S_i\right] \\ &= \sum_{k=0}^{\infty} P(N_D(T) = k) k \int_0^T \mathrm{E}[S|X = x] f_D(x) dx \\ &= \omega \int_0^T \int_0^{\infty} s f(x,s) ds dx. \end{aligned} \quad - \text{식11}$$

여기에서 f_D(X)는 오류 발견 시간에 대한 주변 확률밀도함수이며 아래 식12로 나타낼 수 있다.

$$\mathrm{E}[S|X = x] = \frac{\int_0^{\infty} s f(x,s) ds}{f_D(x)} \quad - \text{식12}$$

오류 발견 시간 X와 오류 수정 시간 S가 독립적인 경우, 즉 테스트 초기와 종반에서 발견된 오류 수정 시간이 같은 경우는 아래 식13과 같다.

$$W(T) = \mathrm{E}[N_D(T)]\mathrm{E}[S] = \omega F_D(T)\mathrm{E}[S] \quad - \text{식13}$$

오류 수정 시간을 고려한 소프트웨어 최적 출시 시점을 결정하는 문제는 총 기대비용 C(T)를 최소화하는 출시 시점 T를 구하는 문제이고 아래 식14로 나타낼 수 있다.

www.kci.go.kr

$$\min_{0\le T\le T_{LC}} C(T). \quad - 식14$$

4. 오류발견시간과 수정시간 간의 상관관계에 따른 소프트웨어 최적 출시 시점

이 장에서는 오류발견시간과 수정시간 간에 상관관계를 나타내기 위하여 아래 식15와 같이 FGM Copula를 사용한다[10].

$$F(x,s) = F_D(x)F_C(x)\left(1+\alpha\overline{F}_D(x)\overline{F}_C(s)\right) \quad - 식15$$

여기에서, $F_D(x) = P(X \le x), F_C(x) = P(S \le s)$는 오류 발견 시간 및 수정 시간의 주변 분포 함수이며, $F(x,s) = P(X \le x, S \le s)$는 동시 분포 함수이다.

또한 $\overline{F}_D(t) = 1-F_D(t), \overline{F}_C(t) = 1-F_C(t)$이다. 상관관계의 강도를 나타내는 상관계수 α는 $-1 \le \alpha \le 1$의 범위를 가진다. α<1의 경우는 음의 상관관계로 초기단계에서 발견된 오류일수록 수정 시간이 길다라는 의미이며, α=0의 경우는 상관관계가 없는 것으로 테스트 초기와 종반에서 발견된 오류의 수정 시간이 동일하다라는 의미이다. 또 α>0의 경우는 양의 상관관계로 초기 단계에서 발견된 오류일수록 수정 시간이 짧다라는 의미가 된다.

FGM Copula에서 오류발견시간과 수정시간 간의 상관관계를 나타낼 때 때 기대누적수정시간은 아래 식16과 같다.

$$W(T) = \omega F_D(T)\mathrm{E}[S] \times\left(1-\frac{\alpha\overline{F}_D(T)}{\mathrm{E}[S]}\int_0^\infty F_C(s)\overline{F}_C(s)ds\right) \quad - 식16$$

오류발견시간과 수정시간을 매개변수 β_D,β_C의 지수 분포로 표현하면 아래 식17과 같다.

$$W(T) = \frac{\omega}{\beta_C}(1-e^{-\beta_D T})\left(1-\frac{\alpha}{2}e^{-\beta_D T}\right) \quad - 식17$$

표 1. 최적 출시 시점과 최소 총 기대 비용.
Table 1. Optimal Time to Launch and Minimum Total Expected Cost

α (상관계수)	T^*(최적 출시 시점)	C(T^*) (최소 총 기대비용)
1.0	495.4	1095.6
0.5	477.3	1077.4
0.0	455.3	1055.3
-0.5	427.5	1027.0
-1.0	389.9	988.0

표1은 c_1=1.0, c'_2=0.01, c'_3=0.2, ω=100.0, β_D=0.01, β_C=0.002, T_{LC}=720.0로 하고, 상관계수 α를 -1부터 1까지 변동시켰을 때 최적 출시 시점 T^*와 최소 총 기대비용 C(T^*)를 나타내고 있다.

매개변수 c_1,c'_2,c'_3의 설정은 기본적인 테스트 비용 c_1대한 비율이 중요하기 때문에 c_1=1로 설정하고 c'_2,c'_3은 기본적인 테스트 비용의 1%, 20%로 상대적으로 설정하였다. 이 표에서 오류발견시간과 수정시간이 양의 상관관계일 때, 독립적인 경우와 비교하여 최적의 소프트웨어 출시 시점이 늦어져 총비용도 비싸짐을 알 수 있다. 반대로 음의 상관관계일때는 최적의 소프트웨어 출시 시점이 빨라져 총비용이 낮아지게 된다.

결론적으로 말하면 오류발견시간과 수정시간이 양의 상관관계일 경우 테스트 마지막 시점에

www.kci.go.kr

수정시간이 긴 오류가 발견되는 경향이 있기 때문에 출시 이후의 오류 수정비용을 적게하기 위하여 출시 시점의 연장을 고려할 필요가 있다. 또한 전체 테스트 비용을 적게 하기 위해서는 테스트 초기에 수정에 시간이 많이 걸리는 오류를 발견하는 것이 좋다는 것도 알 수 있다.

여기에서는 c_1, c'_2, c'_3에 상대적인 비용을 설정하였기 때문에, α=1과 α=-1의 경우 비용의 차이는 1095.6 -988.0 = 107.6 이며 전체 비용의 10% 정도이나, 프로젝트의 규모가 클수록 기본적인 테스트 비용으로 설정한 단위시간당 테스트 비용 c_1이 커지기 때문에 음의 상관관계의 테스트 전략을 실시함으로써 상당히 많은 비용 효과를 볼 수 있다.

5. 결 론

본 논문에서는 오류발견시간과 오류수정시간과의 상관관계에 따라 소프트웨어 개발 비용과 출시 시점 결정에 미치는 영향에 대하여 분석하였다. 분석 결과, 테스트 초기에 수정 시간이 걸리는 오류를 발견하는 것이 경제적인 관점에서 중요하다는 것을 알 수 있었다. 결론적으로 최적의 소프트웨어 출시 시점을 결정하는 데에 있어서 오류발견시간과 오류수정시간의 상관관계를 분석하는 것이 매우 중요하다는 것을 알 수 있었다.

오류발견시간과 수정시간이 양의 상관관계일 경우 테스트 마지막 시점에 수정시간이 긴 오류가 발견되는 경향이 있기 때문에 출시 이후의 오류 수정비용을 적게하기 위하여 출시 시점의 연장을 고려할 필요가 있다. 또한 전체 테스트 비용을 적게 하기 위해서는 테스트 초기에 수정 시간이 많이 걸리는 오류를 발견하는 것이 훨씬 유리하다는 결론을 얻게 되었다.

참 고 문 헌

[1] 장인홍, 정덕환, 이승우, 송광윤, “NHPP소프트웨어 신뢰도 성장모형에서 베이지안 모수 추정과 예측”, 한국데이터정보과학회지, v.24 no.4 755 - 762, 2013년 7월, URL : https://www.koreascience.or.kr/article/JAKO201323263075000.pdf

[2] 김희철 ,“지수화 지수 분포에 의존한 NHPP 소프트 웨어 신뢰성장 모형에 관한 연구”, 한국 컴퓨터 정보학회 논문지, 제11권 5호 9-18. 2006년 11월, URL : http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE06536391

[3] 이상식, 김희철, 송영재. “비동질적 포아송과정을 사용한 소프트웨어 신뢰 성장모형에 대한 베이지안 신뢰성 분석에 관한 연구”, 한국정보처리학회 논문지D 10권5호 805-812, 2003년 8월, URL : http://kiss.kstudy.com/thesis/thesis-view.asp?key=2084472

[4] K. Okumoto & L. Goel, “Optimum release timefor software systems based on reliability and cost criteria”, Journal of Systems and Software, Vol. 1 315-318, 1980, URL: https://www.sciencedirect.com/science/article/abs/pii/0164121279900335

[5] N. F. Schneidewind, “Modelling the fault correction process”, Software Reliability Engineering, 2001. ISSRE 2001. Proceedings. 12th International Symposium on 185-190, December 2001, DOI: https://doi.org/10.1109/ISSRE.2001.989472

[6] M. Xie & Q. P. Hu & Y. P.Wu & S. H. Ng., “A study of the modeling and analysis of software fault-detection and fault-correction processes”, Quality and Reliability Engineering International, Vol. 23 459-470, June 2007, DOI: https://doi.org/10.1002/qre.827

[7] S. S. Gokhale & M. R. Lyu & K. S. Trivedi. “Analysis of software fault removal policies using a nonhomogeneous

www.kci.go.kr

continuous time markov chain", Software Quality Journal Vol. 12 No. 3 211-230, September 2004 DOI: https://doi.org/10.1023/B:SQJO.0000034709.63615.8b
[8] S. S. Gokhale, M. R. Lyu, and K. S. Trivedi., "Incorporating fault debugging activities into software reliability models: A simulation approach", IEEE Transactions on Reliability, Vol. 55 281-292, June 2006, URL : http://www.cse.cuhk.edu.hk/~lyu/paper_pdf/tr-revised.pdf
[9] H. Okamura & T. Dohi. "A generalized bivariate modeling framework of fault detection and correction processes", In Proceedings of the 26th International Symposium on Software Reliability Engineering 35-45, October 2017, DOI: https://doi.org/10.1109/ISSRE.2017.22
[10] H. Joe. "Dependence Modeling with Copulas", CRC Press, Published September 20 2016, ISBN 9781466583221, URL : https://www.taylorfrancis.com/books/9780429103186

저 자 소 개

안철훈(Cheol-Hoon Ahn)

1995.2 홍익대학교 경영학과 졸업
2001.2 홍익대학교 정보공학 석사
2002.3 - 현재 : 경복대학교 소프트웨어융합과 교수
<주관심분야> 소프트웨어공학, 프로그래밍언어

www.kci.go.kr