소프트웨어 시스템의 비용-신뢰도 최적 발행정책의 한계에 관한 연구

論 文
49D-5-6

A Study on the Limitation of Cost-Reliability Optimal Release Policies for Software Systems

崔 圭 植*
(Gyu-Shik Che)

Abstract - I discuss how the existing study results that decide optimum release time are rational and reasonable, considering the cost and reliability simultaneously in this paper. As a study method this paper examines the proposed results and their methodologies centering around the existing related papers for the general cost-reliability optimal release policies, especially their limitation for the derived results.

Key Words : cost-reliability, NHPP, mean value function, optimal release, SRGM, error detection rate

1. 서 론

컴퓨터시스템은 여러 가지 복잡하고 민감한 시스템을 제어하는데 광범위하게 쓰이고 있다. 최근에 와서는 운영시스템, 제어프로그램, 적용프로그램과 같은 여러 가지 소프트웨어 시스템이 더욱더 복잡화 및 대형화되고 있기 때문에 신뢰성이 높은 소프트웨어 시스템을 개발하는 일이 매우 중요하며, 소프트웨어 제품개발에 있어서 소프트웨어의 신뢰도가 핵심사항이다. 1970년대 이후 소프트웨어의 신뢰성을 향상시키기 위한 여러 가지 소프트웨어의 신뢰도 모델이 제시되고 검토되었으며, 특히, 소프트웨어 개발 후 테스트단계에 적용하는 신뢰도를 추정하고 예측하는 모델이 많이 개발되었다. 소프트웨어가 주어진 시간간격동안 고장이 발생하지 않을 확률 즉, 신뢰도는 소프트웨어의 테스트과정을 계속해서 반복 및 수정하면 더욱 더 증가된다. 그러한 결함검출현상을 설명해주는 소프트웨어 신뢰도 모델을 소프트웨어 신뢰도 성장 모델(SRGM)이라 한다.

테스트기간이 길면 길수록 소프트웨어의 신뢰도가 더 높아질 것이나, 개발 소프트웨어를 너무 늦게 발행하면 그에 따른 비용이 증가되고, 사용자에게 불만을 주게 된다. 그러므로, 소프트웨어 개발에서 중요한 문제는 언제 소프트웨어의 테스트를 중단하여 발행하느냐 곧, 지금까지 광범위하게 연구된 "최적 소프트웨어 발행절차서(SRP)문제"[1-4]를 결정하는 것이다. Okumoto와 Goel[1]은 전체평균 소프트웨어비용을 최소화시키는 비용-최적 SRP를 발표하였다. Yamada와 Osaki[4]는 전체 평균 비용을 최소화시키고 소프트웨어 신뢰도를 만족시키는 전체평균비용-신뢰도-최적 SRP를 도입하였다. 이러한 연구결과를 참조하여 Hou, Kuo, Chang[5]은 지수 곡선과 로지스틱 곡선에 적용하는 연구를 수행하였다. 소프트웨어공학에서 신뢰도 분야는 아직까지 큰 비중을 차지하고 있지 못하지만, 앞으로 소프트웨어 산업의 중요성이 더욱 높아진다고 전망해볼 때, 최소의 비용으로 개발 소프트웨어의 신뢰도를 극대화할 수 있는 기법이 계속 연구된다면 소프트웨어의 비용을 크게 줄여 컴퓨터 시스템을 사용하는 여러 분야에 공헌할 것으로 믿는다.

본 논문에서는 지금까지 연구된 비용과 신뢰도에 대한 두 개의 기준을 동시에 고려하여 발행시기를 결정하는 최적발행정책에 대해서 연구해보고, 특히, 이러한 연구결과의 한계에 대해서 연구하고자 한다. 연구방법으로서 소프트웨어 신뢰도를 만족시키는 조건하에서 전체평균 소프트웨어 비용을 최소화시키는 최적 소프트웨어 발행정책을 검토하며, 이러한 최적 비용-신뢰도 발행정책에 대한 연구가 현실적으로 타당한 것인가, 일반적으로 적용될 수 있는 합리성을 가지고 있는가를 연구한다.

2장에서는 NHPP를 기준으로 한 SRGM에 대해서 고장확률 및 신뢰도를 연구하고, 3장에서는 테스트 기간중 및 발행운전 기간중에 검출되는 에러의 수정비용과 테스트비를 연구한다. 4장에서는 비용-신뢰도 최적발행정책을 검토하여 연구하고 이러한 정책의 한계에 대해서 연구하며, 이러한 논의를 바탕으로 5장에서는 비용-신뢰도 최적발행정책의 한계에 대한 결론을 맺는다.

* 正 會 員 : 建陽大 情報電子通信工學部 副敎授 · 工博
接受日字 : 1999年 5月 27日
最終完了 ; 1999年 12月 9日

기호설명

$N(t)$: 시각 t까지 검출되는 소프트웨어의 누적에러 갯수
a : 초기부터 소프트웨어내에 존재하고 있는 에러의 갯수
b, b_1, b_2 : 에러검출비, 일반적인 경우, 테스트기간중, 발

행후 운전기간중, $b_1 \geq b_2$

$m(t)$: $E[N(t)]$, 평균치 함수

$R(x|t)$: 소프트웨어의 신뢰도, 시각 $t(x \geq 0)$에서 에러가 검출된 후 $(t, t+x)$에서 고장이 일어나지 않을 확률

R_o : 목표신뢰도, $0<R_o<1$

c_1 : 테스트기간중에 검출되는 에러를 수정하는 단위비용, $c_1>0$

c_2 : 운전기간에 검출되는 에러를 수정하는 단위비용, $c_1>c_2$

c_3 : 단위시간당 테스트 비용, $c_3>0$

$C(T)$: 전체 평균 소프트웨어 비용

T_{LC} : 소프트웨어의 수명주기

T : 전체 테스트 시간

T^* : 최적 소프트웨어 테스트 시간

T_1 : $dC(T)/dT=0$을 만족시키는 유일 해 T

T_2 : $R(x|T)=R_o$를 만족시키는 유일 해 T

2. SRGM

소프트웨어 개발단계에서 테스트하는 소프트웨어 시스템에 대해서 고찰해보기로 한다. 소프트웨어 고장은 "시스템 내에 잔존하는 에러에 의해 프로그램 운전이 원치 않게 중지되는 것"으로 정의한다. SRGM 영역에서 아래와 같은 일반적인 가정을 도입한다.[6]

1) 소프트웨어 시스템은 소프트웨어 에러에 의해서 무작위 시간적으로 고장이 발생될 수 있다.
2) 소프트웨어 고장이 발생될 때마다 이것을 일으키는 소프트웨어의 에러를 즉시 제거하며, 새로운 에러는 도입되지 않는다.
3) 테스트기간중 및 운전기간중에 검출되는 에러의 평균 수정비용은 각각 $c_1 m(T)$, $c_2\{m(T_{LC})-m(T)\}$이고, 테스트 비용은 $c_3 T$이다[4,5]
4) T_{LC}는 소프트웨어의 수명기간이므로 $T_{LC}>\max\{T_1, T_2\}$이다.[4,5]

소프트웨어의 테스트에 의해서 검출되는 누적에러의 수를 기술할 때, 일반적으로 에러의 검출시간단위로서 역일시간(calendar time)이나 장비의 실행시간과 같은 테스트시간을 사용한다. $\{N(t), t \geq 0\}$를 시간간격$(0, t)$에서 검출되는 누적 에러(또는 고장)의 수를 나타내는 계수공정이라 하면, NHPP의 평균치함수로 불리는 $N(t)$의 기대치는 $m(t)$로 정의한다. NHPP에 근거한 SRGM을 아래와 같이 정의한다.

$$\Pr\{N(t)=n\} = \frac{\{m(t)\}^n}{n!}\exp[-m(t)] \quad (1)$$

여기서,

$$m(t) = a[1-\exp(-bt)],\ a>0,\ b>0,\ t \geq 0 \quad (2)$$

이다. $a(=m(\infty))$를 최종적으로 검출될 기대누적에러의 수 즉, 추정할 최초의 기대에러라고 정의하면 다음과 같은 식을 유도할 수 있다.

$$\lim_{t\to\infty} \Pr\{N(t)=n\} = \frac{a^n}{n!}\exp(-a) \quad (3)$$

이는 $N(t)$가 오랜기간동안 테스트를 한 후 평균치 a를 가진 Poisson을 따른다는 것을 의미한다.

$m(t)$를 가진 NHPP로부터 유도되는 정량적 신뢰도척도의 하나는 소프트웨어 신뢰도이다. 소프트웨어 신뢰도는 다음과 같이 정의한다.

$$R(x|t) = \exp[m(t+x)-m(t)] \quad (4)$$

여기서, $m(t+x)-m(t) = a(1-e^{-b(t+x)})-a(1-e^{bt})$

$= ae^{-bt}(1-e^{-bx}) = m(x)e^{-bt}$

이므로,

$$R(x|t) = \exp[-m(x)e^{-bt}] \quad (5)$$

이다.

3. 소프트웨어의 수정 비용

Okumoto와 Goel[1]은 소프트웨어 신뢰도 성장모델을 이용하여 비용 및 신뢰도기준에 근거한 최적 소프트웨어 발행정책을 연구하였으며, Yamamoto와 Osaki[4]는 비용과 신뢰도를 동시에 고려하여 소프트웨어 시스템의 최적발행시각을 결정하고자 하였다. Rong-Huei Hou, Sy-Yen Kuo, Yi-Ping Chang[5]은 이러한 연구결과를 지수형 성장곡선과 로지스틱 성장곡선에 적용하는 연구를 하였다.

3.1 테스트 기간중의 에러수정비용

테스트 기간중에 검출되는 총 에러의 수에 에러당 수정비용을 곱한 값이 된다.

$$c_1 m_1(T) = c_1 a(1-e^{-b_1 T}) \quad (6)$$

3.2 운전기간중의 에러수정비용

운전기간중의 에러수정비용을 구하기 위해서 다음과 같이 계산한다. 발행시각 T에서의 누적발생에러수는 $m_1(T) = a(1-e^{-b_1 T})$이므로, 잔여에러의 수 $\bar{a} = a-a(1-e^{-b_1 T})$ $= ae^{-b_1 T}$이다. 이것이 발생후 운전중의 초기 에러수이므로,

$$m_2(t) = \bar{a}(1-e^{-b_2(t-T)}) + m_1(T)$$

$$= ae^{-b_1 T}(1-e^{-b_2(t-T)}) + a(1-e^{-b_1 T})$$

따라서, 수명주기와 이 기간동안 검출되는 에러의 총 수는

$$m_2(T_{LC}) = ae^{-b_1 T}(1-e^{-b_2(T_{LC}-T)}) + a(1-e^{-b_1 T})$$

이고, 이 값에서 테스트기간중에 검출되는 총 에러의 수 $m_1(T) = a(1-e^{-b_1 T})$를 제외한 것이 순수하게 운전중에

검출되는 총에러의 수이므로,

$$m_2(T_{LC})-m_1(T)= ae^{-b_1T}(1-e^{-b_2(T_{LC}-T)}) + a(1-e^{-b_1T})-a(1-e^{-b_1T}) = ae^{-b_1T}(1-e^{-b_2(T_{LC}-T)})$$

가 된다. 그러므로, 운전중에 검출되는 에러의 수정비용은

$$c_2\{m_2(T_{LC})-m_1(T)\}=c_2ae^{-b_1T}(1-e^{-b_2(T_{LC}-T)}) \quad (7)$$

이다.

3.3 테스트기간중의 테스트 비용

전 테스트기간에 단위시간당 테스트비용을 곱한 값이 된다.

$$c_3T \quad (8)$$

3.4 총비용

상기와 같은 논리에 의해서 총 비용은 테스트기간중의 에러수정비용, 운전기간중의 에러수정비용, 테스트기간중의 테스트비용을 합한 값이 된다.

$$C(T)=c_1m_1(T)+c_2\{m_2(T_{LC})-m_1(T)\}+c_3T$$
$$=c_1a(1-e^{-b_1T})+c_2ae^{-b_1T}(1-e^{-b_2(T_{LC}-T)})+c_3T \quad (9a)$$

최적 소프트웨어 발행시각은 전체 평균 소프트웨어 비용을 최소화하는 테스트시간이다.
$C(T)$의 최저값을 구하기 위해 식(9a)를 T로 미분하여 정리한다.

$$(c_1-c_2)ab_1e^{-b_1T}+c_2a(b_1-b_2)e^{-b_2T_{LC}}e^{-(b_1-b_2)T}+c_3=0$$

이 식을 만족시키는 $T>0$인 범위의 T값을 구하면 $C(T)$에 대한 내부최소값이 된다. 그러나, 이 식에서 보통의 해석적인 방법으로는 T값을 구하기가 쉽지 않다. 따라서, 수치해석적인 반복법에 의하여 최적발행시각 $T=T_1$을 구한다. 특별히 소프트웨어 발행 전후의 에러검출비율이 같을 경우에는 $b_1=b_2=b$가 되기 때문에 상기식이 단순화되어 이 때 식(9a)는

$$C(T)=c_1m(T)+c_2\{m(T_{LC})-m(T)\}+c_3T \quad (9b)$$

와 같이 된다.
즉, (9b)에서 최소값을 구하기 위한 조건은

$$(c_1-c_2)abe^{-bT}+c_3=0$$

이므로

$$T_1=\frac{1}{b}\ln\frac{ab(c_2-c_1)}{c_3} \quad (10a)$$

이다.[4] 그런데, $T_1>0$이므로, $\ln\frac{ab(c_2-c_1)}{c_3}>0$ 즉, $ab(c_2-c_1)>c_3$일 때에만 (10a)가 성립되며, 그 외에는 구할 수 없는 외부 최소값이 존재하여 그 경계치

$$T_1=0 \quad (10b)$$

에서 최소가 된다.

마찬가지로, 유사한 내부해법이 존재하여 신뢰도를 요건에 최근접시키는 유일한 시각이 존재한다. 최적 소프트웨어 발행시각은 미리 규정된 소프트웨어 신뢰도 $R(x|T)=R_o$를 만족시키는 최근접이 되는 시각이다. 여기서, x는 최근에 검출된 에러를 수정한 후 경과되는 시간이다.
발행시각 T에서의 신뢰도는 (5)에서

$$R(x|T)=\exp[-a(1-e^{-b_2x})e^{-b_1T}]=R_0 \quad (11)$$

$$T_2=\frac{1}{b_1}[\ln m(x)-\ln(\ln\frac{1}{R_0})] \quad (12)$$

이고, 여기서 R_o는 소프트웨어가 추구하는 목표신뢰도이다. 특히, $b_1=b_2=b$인 경우는 단순화되어

$$T_2=\frac{1}{b}\left\{\ln m(x)-\ln[\ln\frac{1}{R_0}]\right\} \quad (13a)$$

이다.[4] 그런데,

$$R(x|0)=\exp[-m(x)]=\exp[-a(1-e^{-bx})]$$

로서, x가 증가함에 따라 $R(x|0)$가 감소한다. 즉, $R(x|0)<R_0$일 때에만 (13a)가 성립되며, 그 외에는 구할 수 없는 외부 해가 존재하며, 경계치

$$T_2=0 \quad (13b)$$

에서 최근접이 된다.

4. 비용-신뢰도 최적 소프트웨어 발행정책

소프트웨어 테스트로부터 구한 소프트웨어의 신뢰도를 어떤 규정치로 유지하는 제한 하에 전체 평균 소프트웨어의 비용을 최소로 하는 최적 소프트웨어 발행정책에 대해서 고려해보기로 한다. 최적 소프트웨어 발행문제는 아래와 같이 공식화할 수 있다.

$c_2>0, c_1>c_3>0, x\geq 0, 0<R_0<1$인 경우에 대해서
$R(x|T)\geq R_0, T\geq 0$인 조건하에 $C(T)$를 최소화 (14)

이러한 방법으로 하여 비용-신뢰도최적 소프트웨어 발행시각에 대한 해를 구할 수 있다.

$$T^*=\max\{T_1, T_2\} \quad (15)$$

여기서, T_1은 (10)에서, T_2는 (13)에서 구한 값이다.

1) $ab>c_3/(c_2-c_1)$이고 $R(x|0)<R_0$이면 (10)과 (13)을 만족시키는 양의 유일한 T_1와 T_2가 각각 존재한다.
2) $ab>c_3/(c_2-c_1)$이고 $R(x|0)\geq R_0$이면 $T^*=T_1$
3) $ab\leq c_3/(c_2-c_1)$이고 $R(x|0)<R_0$이면 $T^*=T_2$
4) $ab\leq c_3/(c_2-c_1)$이고 $R(x|0)\geq R_0$이면 $T^*=0$

여기서, 식(10)과 식(13)을 근거로 하여 도출된 식(15)가 현실적으로 타당한가를 연구해보고자 한다. 상기식이 타당성을 가지려면 현실적으로 적용가능한 식이 되어야 한다. 비용-신뢰도 발행문제에 있어서 가장 이상적인 것은 그림 1에서 보는 바와 같이 목표신뢰도를 만족시키면서 소프트웨어 전 수명기간동안의 수정비용이 최소가 되는 경우 즉, 최적치 연구에서 $T_1 > T_2$가 되어 $T^* = T_1$로 결정되는 경우이다.

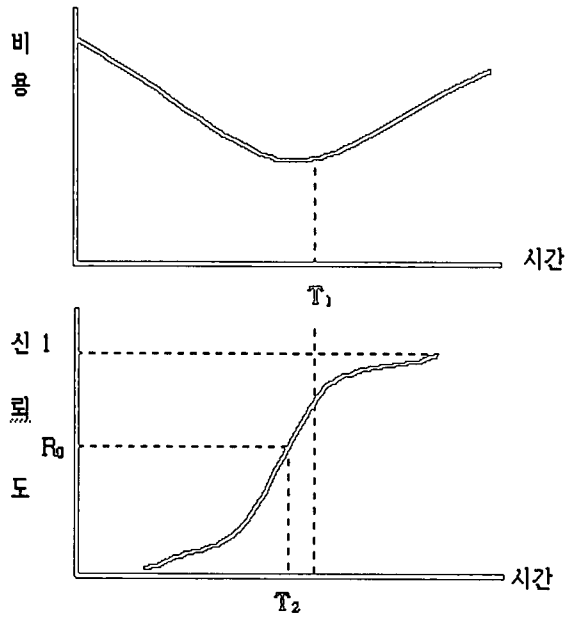

그림 1 비용-신뢰도 최적관계 곡선
Fig. 1 cost-reliability relation curve

문제를 단순화시키기 위해서 특별히 테스트기간중의 에러검출비와 운전기간중의 에러검출비가 같다고 가정하여 $b_1=b_2=b$라고 하고 우선 상기 조건 1)을 고찰해보기로 한다.

$R(x|0) < R_o$ 즉,

$R(x|0) = \exp[-m(x)] = \exp[-a(1-e^{-bx})] < R_o$,

$ab(c_2-c_1) > c_3$

인 조건 하에서 $T_1 > T_2$인 경우를 고찰하는 것이므로, 상기 조건을 만족하려면

$$\frac{1}{b}\ln\frac{ab(c_2-c_1)}{c_3} > \frac{1}{b}\left\{\ln\frac{m(x)}{\ln\frac{1}{R_o}}\right\} \tag{16}$$

로서 $\frac{ab(c_2-c_1)}{c_3} > 1$이므로,

$$\frac{ab(c_2-c_1)}{c_3} > \frac{m(x)}{\ln\frac{1}{R_o}} \tag{17}$$

이고, $R(x|0) < R_o$인 조건을 고려하여 이 식을 정리하면

$$\exp[-m(x)] < R_o < \exp\left[-\frac{c_3}{ab(c_2-c_1)}m(x)\right] \tag{18}$$

이 된다. 즉, 조건 1)을 만족시키면서 $T_1 > T_2$가 되려면 식(18)과 같은 목표신뢰도의 제한을 받게 된다.

그리고, 조건 2)의 경우는 $T_2 = 0$ 즉, 소프트웨어가 개발완료되는 순간의 신뢰도가 목표신뢰도를 만족하기 때문에 테스트할 필요 없이 곧바로 소프트웨어를 발행하는 경우이다. 이 때에는 T_1의 결정이 의미를 가진다고 볼 수 있다. 그러나, 이 때에도 $R(x|0) \geq R_o$로부터

$$R_o \leq \exp[-m(x)] \tag{19}$$

의 목표신뢰도 제한을 받는다. 따라서, 목표신뢰도를 만족시키면서 전 수명기간동안의 소프트웨어 수정비용이 최소가 되는 경우 즉, $T^* = T_1$으로 되는 때는 $T_2 = 0$인 경우뿐이고, 그 신뢰도 또한 식(19)와 같은 제한을 받게 된다. 그리고, 소프트웨어가 개발되자마자 테스트 없이 곧바로 발행된다는 것도 현실에 맞지 않는다.

조건 1)과 2)를 만족시킬 수 있는 목표신뢰도의 한계치를 고찰해보기 위해서 [1]에서 인용한 모델파라미터인 $a=1348$, $b=0.124$, $x=0.1$, $R_0=0.7$, $c_1=1$, $c_2=5$, $c_3=100$, $T_{LC}=100$에 적용해보기로 한다.

이러한 값들을 이용하여 $C(T)$를 최소화시키는 비용최소화 기준과 $R(x|T)=R_o$를 만족시키면서 소프트웨어 전 수명기간동안의 수정비용이 최소가 되는 경우 즉, 최적치 연구에서 $T_1 > T_2$가 되어 $T^* = T_1$로 결정되는 경우의 목표신뢰도 한계를 고찰해보면, 각각에 대해서 $\exp[-m(x)] \to 0$, $\exp\left[-\frac{c_3}{ab(c_2-c_1)}m(x)\right] \to 0$이므로, $R_o \simeq 0$이다. 즉, 비용을 최소로 하면서 원하는 정도의 목표신뢰도를 얻는다는 것은 어렵다. 따라서, 일반적으로 통용되는 목표 신뢰도, 예를 들면 $R_o = 95\%$와 같은 높은 정도의 신뢰도를 얻으려면, 그 최적발행시각이 최저비용에 의해서 결정되는 것이 아니고 목표신뢰도에 의해서 결정되기 때문에, 비용을 최저로 하고자 한 [4]와 [5]의 연구결과는 재검토되어야 한다고 본다.

상기와 같은 알고리즘에 의해 비용-신뢰도를 산출할 경우, 원하는 목표 신뢰도를 얻고 동시에 발행비용을 최소로 하기 위해서는 소프트웨어를 이상적으로 잘 개발하여 소프트웨어 내에 에러의 수가 현저하게 적어지게 하거나, 에러의 검출비가 낮아서 $b \to 0$로 되도록 해야 한다.

5. 결 론

소프트웨어 시스템을 개발하여 발행하기 전에 소프트웨어 시스템 내에 존재하고 있는 에러를 검출하여 수정하기 위한 테스트를 수행할 때, 언제 테스트를 끝내고 발행해야 하는가

하는 문제가 발생하게 된다. 이러한 최적 발행시기를 결정하기 위해 여러 연구자들이 전 수명기간중의 투입비용과 신뢰도를 동시에 고려하여 연구를 수행하였다.

본 논문에서는 이러한 발표논문을 중심으로 하여 소프트웨어의 비용-신뢰도 최적발행시기 결정방법에 관한 한계를 연구하였다. 비용-신뢰도 발행 문제에 있어서 가장 이상적인 것은 목표신뢰도를 만족시키면서 수명기간 동안의 소프트웨어 수정비용이 최소가 되도록 발행시기를 결정하는 것이다. 그러나, 연구 결과 이러한 경우는 목표신뢰도가 비현실적으로 극히 낮은 경우이거나, 소프트웨어가 개발되는 순간의 신뢰도가 목표신뢰도를 만족하여 테스트할 필요 없이 곧바로 소프트웨어를 발행하는 경우뿐인 것으로 나타났다. 이러한 경우는 극히 발생확률이 적은 비현실적인 경우에 해당된다. 그 외에는 전 수명기간동안의 수정비용과 관계 없이 목표신뢰도가 항상 소프트웨어의 발행시기를 결정하게 되므로 비용-신뢰도 최적발행에 대한 그동안의 연구를 재검토해볼 필요가 있다.

그러므로, 새로운 비용-신뢰도 최적발행에 관한 알고리즘이 연구개발되어야 할 것으로 사료된다.

감사의 글

96-03 본 연구는 한국전력공사의 지원에 의하여 기초전력공학 공동연구소 주관으로 수행되었음

참 고 문 헌

[1] K. Okumoto, A.L. Goel, "Optimum release time for software systems based on reliability and cost criteria", J. System Software, vol 1, 1980, pp315-318

[2] H.S. Koch, P. Kubat, "Optimal release time of computer software", IEEE Trans. Software Eng'g, vol SE-9, 1983 May, pp323-327

[3] M. Xie, "On the determination of optimum software release time", Proc. 2nd Int' Symp. Software Reliability Eng'g, 1991, pp218-224

[4] S. Yamada, S. Osaki, "Cost-reliability optimal release policies for software systems", IEEE Trans. Reliability, vol R-34, 1985 Dec., pp422-424

[5] Rong-Huei Hou, Sy-Yen Kuo, Yi-Ping Chang, "Optimal Release Policy for Hyper-Geometric Distribution Software-Reliability Growth Model", IEEE Trans Reliability, vol 45, 1996 Dec., pp646-651

[6] S. Yamada, H. Ohtera, H. Narihisa, " Software Reliability Growth Models with Testing-Effort", IEEE Trans Reliability, vol R-45, 1986 April, pp19-23

저 자 소 개

최 규 식 (崔 圭 植)

1948년 12월 29일생. 1976년 2월 서울대 전기공학과 졸업(학사). 1983년 6월 뉴욕공과대학 전기과 졸업(석사). 1993년 2월 명지대학교 전기공학과 졸업(공박). 1978.2-1993.2 한국전력기술(주) 연구소 책임연구원. 현재 건양대 정보전자통신공학부 교수

Tel : 0461-730-5283

E-mail : che@kytis.konyang.ac.kr