극값분포 특성에 근거한 소프트웨어 최적 방출시기에 관한 비교

The Comparative Study of Software Optimal Release Time Based on Extreme Distribution Property

김 희 철*
Kim-Hee Cheul*

Abstract

Decision problem called an optimal release policies, after testing a software system in development phase and transfer it to the user, is studied. The infinite failure non-homogeneous Poisson process models presented and propose an optimal release policies of the life distribution applied extreme distribution which used to find the minimum (or the maximum) of a number of samples of various distributions. In this paper, discuss optimal software release policies which minimize a total average software cost of development and maintenance under the constraint of satisfying a software reliability requirement. In a numerical example, extreme value distribution as another alternative of existing the Poisson execution time model and the log power model can be verified using inter-failure time data

요 약

본 연구에서는 소프트웨어 제품을 개발하여 테스팅을 거친 후 사용자에게 인도하는 시기를 결정하는 방출문제에 대하여 연구되었다. 무한고장수를 가진 비동질적인 포아송 과정에 기초하고 수명분포는 최소 및 최대값을 적합 시키는데 효율성을 가진 극값 분포를 이용한 최적 방출시기에 관한 문제를 제시하여 소프트웨어 요구 신뢰도를 만족시키고 소프트웨어 개발 및 유지 총비용을 최소화 시키는 최적 소프트웨어 방출 정책에 대하여 논의 되었다. 본 논문의 수치적인 예에서는 고장 간격 시간 자료를 적용하여 기존의 로그 포아송 실행시간 모형과 로그 파우어 모형의 대안으로서 극값 분포모형이 또 하나의 대안이 될 수 있음을 입증하였다.

Key words : Software Release Policies; Infinite Failure Mean Value Function; Extreme Distribution Property

Ⅰ. 서론

소프트웨어 방출시간에 대한 연구들은 대부분 유한 고장 NHPP(Non-Homogeneous Poisson Process)모형을 사용하였다[1, 2]. 유한(finite)고장 NHPP모형은 소프트웨어가 유한개의 고장이 있고 고장 제거 단계에서는 새로운 고장이 발생하지 않는다는 가정을 한 모형이다. 그러나 실제 고장 제거 단계에서도 새로운 고장이 발생 할 수 있다. 따라서 본 연구에서는 무한(Infinite) 고장 NHPP 모형을 이용하여 최적 방출시기에 대한 문제를 제안하고자 한다. 이 분야에서는 Musa-Okumoto의 대수 포아송 실행시간 모형[3,4] 과 로그-파우어 모형[5,6]을 이용한 방출 문제에 대한 문제들이 이미 연구되었고 최근에도 이와 관련된 문제에 대한 연구는 Yang 과 Xie(2000) 와 Huang(2005)에 의해 연구되고 있다[7, 8]. 본 연구에서는 소프트웨어의 결함을 제거하거나 수정 작업 중에도 새로운 결함이 발생될 가능성이 있는 무한고장수를 가진 최소 및 최대값을 적합 시키는데 효율적인 특성을 가진 극값분포를 이용한 최적 방출시기에 관한 문제를 다루었다.

본 논문의 2절에서는 관련 연구로서 무한고장 NHPP, 기존모형, 요구 신뢰도와 비용 최소화를 고려한 방출시간 그리고 제안된 극값 분포모형 및 수치적인 예를 약술하였고 3절에서는 그 결론을 나열 하였다.

* 남서울大學校 産業經營工學科
(Department of the Industrial & Management Engineering Namseoul University)
接受日:2011年 3月 2日,修正完了日: 2011年 3月 28日

II. 본론

1. 무한고장 NHPP

NHPP 모형에서 평균값 함수 $m(t)$(Mean value function)와 강도 함수(Intensity function) $\lambda(t)$는 다음과 같은 관계로 표현할 수 있다[1, 9].

$$m(t)=\int_0^t \lambda(s)ds \ , \frac{d\,m(t)}{dt}=\lambda(t) \qquad (1)$$

따라서 $N(t)$는 모수 $m(t)$을 가진 포아송 확률 밀도 함수(Probability density function; Pdf)로 알려져 있다. 즉,

$$P\{N(t)=n\}=\frac{[m(t)]^n \cdot e^{-m(t)}}{n!},\ n=0,1,2,\ldots\infty \qquad (2)$$

이처럼 시간 관련 모형(Time domain models) 들은 NHPP에 의해서 확률 고장 과정으로 설명이 가능하다. 이러한 모형들은 고장 강도 함수 $\lambda(t)$가 다르게 표현됨으로서 평균값 함수 $m(t)$도 역시 다르게 나타나고 무한 고장 NHPP 모형들은 각 수리 시점에서도 고장이 발생할 수 있는 상황을 추가하기 위하여 기록 멈춤 통계량(Record breaking statistics)을 사용하는 RVS(Record Value Statistics)모형을 사용할 수 있다고 하였고 이 RVS 모형과 NHPP 모형에 관해서 평균값함수는 다음과 같이 된다고 하였다[9].

$$m(t)= -\ln(1-F(t)) \qquad (3)$$

따라서 (1)식 과 (3)식을 연관시키고 $f(t)$을 확률밀도함수, $F(t)$을 분포함수라고 하면 다음과 같은 관계식에 의해 NHPP의 강도함수는 $F(t)$의 위험함수($h(t)$가 된다.

$$\lambda(t)=m'(t)=f(t)/(1-F(t))=h(t) \qquad (4)$$

결국 무한 고장 NHPP 모형의 평균값 함수와 고장 강도 함수는 각각 다음과 같이 유도된다[9].

$$m(t)= -\ln(1-F(t)) \qquad (5)$$
$$\lambda(t)=m'(t)=f(t)/(1-F(t))=h(t) \qquad (6)$$

시간 $(0,t]$ 까지 조사하기 위한 시간절단(Time truncated) 모형은 n번째 까지 고장 시점 자료를 x_n이라고 하고 Θ을 모수공간이라고 하면 우도 함수는 다음과 같이 알려져 있다[9].

$$L_{NHPP_{INFIN}}(\Theta \mid D_t)=\prod_{i=1}^{n}\left(\frac{f(x_i)}{1-F(x_i)}\right)(1-F(x_n)) \qquad (7)$$
$$=\left(\prod_{i=1}^{n}h(x_i)\right)(1-F(x_n))$$

2. 기존 모형

가. 로그 포아송 실행시간모형

로그 포아송 실행시간(Log Poission execution time)모형[3, 5]는 1984년에 Musa 와 Okumoto에 의해서 소개된 무한 고장 소프트웨어 모형으로 평균값함수와 강도함수는 다음과 같이 알려져 있다.

$$m(t)=\frac{1}{\theta}ln(\lambda_0\theta t+1),\lambda(t)=\frac{\lambda_0}{\lambda_0\theta t+1} \qquad (9)$$

최우추정치 $\hat{\theta}_{MLE}$ 와 $\hat{\lambda_0}_{MLE}$은 다음과 같이 구할 수 있다고 하였다[5, 6].

$$\hat{\theta}_{MLE}=\frac{1}{n}ln(\hat{\phi}\ x_n+1)\ ,\ \hat{\lambda_0}_{MLE}=\hat{\phi}/\hat{\theta}_{MLE} \qquad (10)$$

즉, 이 ϕ 근을 구하기 위해서는 수치 해석적 방법으로 다음과 같은 식을 이용하여 계산할 수 있다고 하였다.

$$\frac{\partial \ln L(\phi \mid \underline{x})}{\partial\phi}= \qquad (11)$$
$$\frac{n}{\phi}-\sum_{i=1}^{n}\frac{x_i}{\phi x_i+1}-\frac{n\,x_n}{(\phi\,x_n+1)\ln(\phi x_n+1)}=0$$

단, $\phi\left(=\hat{\lambda}_{0_{MLE}}\cdot\hat{\theta}_{MLE}\right)$는 (11) 식의 근이 된다.

나. 로그 파우어 (Log Power ; LP)모형

로그 파우어 (Log Power ; LP)모형[5, 6]은 1999년에 Xie 와 Homg에 의해서 발전된 무한 고장 소프트웨어 모형으로 평균값함수와 강도함수는 다음과 같이 알려져 있다.

$$m(t)=a\ln^b(1+t),\ \lambda(t)=\frac{a\,b\ln^{b-1}(1+t)}{1+t} \qquad (12)$$

다음과 같은 식을 만족하는 $\hat{a}_{MLE}$ 와 $\hat{b}_{MLE}$을 수치 해석적 방법으로 계산할 수 있다고 하였다[5].

$$\frac{\partial \ln L(a,b \mid \underline{x})}{\partial a}=\frac{n}{a}-\ln^b(1+x_n)=0 \qquad (13)$$

$$\frac{\partial \ln L(a,b \mid \underline{x})}{\partial b}=\frac{n}{b}-\ln\left(\sum_{i=1}^{n}\ln(1+x_i)\right) \qquad (14)$$
$$-a\ln^b(1+x_n)\ln(\ln(1+x_n))=0$$

3. 요구신뢰도와 비용 최소화를 고려한 방출시간

NHPP 모형에서 테스트 시점 x_n(마지막 고장시점)에서 소프트웨어 고장이 일어난다고 하는 가정 하에서 신뢰구간 $(x_n,\ x_n+x]$ (단, x는 임무시간(Mission time))동안 소프트

웨어의 고장이 일어나지 않을 확률인 신뢰도 $\hat{R}(x \mid x_n)$는 다음과 같이 됨이 알려져 있다[8, 9].

$$\hat{R}(x \mid x_n) = \exp\left(-\int_{x_n}^{x_n+x} \lambda(\tau)d\tau\right) = \exp[-\{m(x+x_n)-m(x_n)\}] \quad (15)$$

따라서 로그 포아송 실행시간모형에 대한 신뢰도는 평균값함수 (15)식과 $t = x_n$을 이용하면 다음과 같이 표현된다.

$$R(x \mid t) = \exp\left(-\frac{1}{\theta}[\ln(\lambda_0\theta(x+t)+1)-\ln(\lambda_0\theta t+1)]\right) \quad (16)$$

따라서 소프트웨어 방출시간 T_R이 신뢰도$R_0 = \hat{R}(x \mid t)$을 확보해야 한다면 다음 방정식을 만족해야 한다.

.

$$\ln R_0 = -\frac{1}{\theta}[\ln(\lambda_0\theta(x+T_R)+1)-\ln(\lambda_0\theta T_R+1)] \quad (17)$$

유사한 방법으로 로그 파우어 모형에서도 다음과 같이 유도된다[7].

$$R(x \mid t) = \exp[-a(\ln^b(1+(x+t))-\ln^b(1+t))] \quad (18)$$

$$\ln R_0 = -a(\ln^b(1+(x+T_R))-\ln^b(1+T_R)) \quad (19)$$

비용 최소화와 관련된 최적 방출시간은 신뢰도와 함께 비용모형에 의해서 결정된다. 소프트웨어 방출시간을 T로 표현하고 $m(T)$와 $m(\infty)$을 각 각 $(0, T]$와 $(0, \infty)$의 기간에 발견된 기대 고장수라고 표현하고 $C(T)$을 소프트웨어 라이프사이클(life cycle) 동안에 기대되는 소프트웨어 비용이라고 하면 $C(T)$는 다음과 같이 표현 된다[4, 7].

$$C(T) = c_1 m(T) + c_2[m(\infty)-m(T)] + c_3 T \quad (20)$$

위 식에서 c_1는 테스팅 동안에 하나의 고장을 수리하는 비용이고 c_2 가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$), 그리고 c_3는 단위 시간당 테스팅 비용을 나타낸다. 이와 관련하여 총비용의 최소화는 무한고장 평균값 함수를 가진 NHPP 모형에 대하여 발생 할 수 있다. 무한 수명에 대한 비용함수 $C(T)$식인 (20)과 (22)에서 $m(\infty)$은 직접 추정할 수 없기 때문에 이 식을 사용하기 위해서는 소프트웨어 수명시간 인 T_{LC} 을 지정하여 분석한다[4]. 이러한 T_{LC}는 소프트웨어마다 서로 다른 임의의 값이기 때문에 유한 고장 NHPP 모형이라고 할 수 는 없다. 따라서 비용함수를 고려하여 소프트웨어의 모든 수명에서 총비용을 최소화함으로서 최적 테스팅 시간을 결정 할 수 있고 다음과 같은 식을 만족하면 비용함수 $C(T)$는 유일한 최소값을 가진다[5].

$$\frac{dC(T)}{dT} = 0, \quad \frac{d^2C(T)}{d^2T} > 0 \quad (21)$$

결국 소프트웨어 지정된 수명T_{LC}을 이용한 로그 포아송 실행시간모형 비용함수 $C(T)$는 다음과 같이 유도된다.

$$C(T) = c_1 m(T) + c_2[m(\infty)-m(T)] + c_3 T = (c_1-c_2)\frac{1}{\theta}ln(\lambda_0\theta T+1) + \frac{c_2}{\theta}ln(\lambda_0\theta T_{LC}+1) + c_3 T \quad (22)$$

T 에 관해서 비용함수 $C(T)$을 미분하면 다음과 같은 방정식을 만족하는 최적방출시간 T_C 를 계산 할 수 있다[5, 6].

$$\frac{(c_1-c_2)\lambda_0}{\lambda_0\theta T_C+1} + c_3 = 0 \quad (23)$$

위 식에서 최적방출시간은 소프트웨어 지정 수명시간 인 T_{LC} 와 의존하지 않는다는 것을 알 수 있다. 이러한 사실은 무한 고장 평균값 함수를 가진 NHPP모형들이 새로운 결점들이 발생함으로서 몇 개의 고장이 야기 될 수 있는 점을 고려한 모형으로 적합 시킬 수 있다[4]. 따라서 신뢰성 요구를 만족하고 총 비용을 최소화하는 상황이 최적 방출 시간이다. 결국 로그 포아송 실행시간모형을 사용한 최적 방출시간 T_{OP} 는 T_R과 T_C에 대하여 다음을 만족한다[6].

$$T_{OP} = Max(T_C, T_R) \quad (24)$$

(24) 식에서 T_R과 T_C는 다음 두 방정식에 의해서 계산 될 수 있다.

$$\ln R_0 = -\frac{1}{\theta}[\ln(\lambda_0\theta(x+T_R)+1)-\ln(\lambda_0\theta T_R+1)] \quad (25)$$

$$\frac{(c_1-c_2)\lambda_0}{\lambda_0\theta T_C+1} + c_3 = 0 \quad (26)$$

단, 신뢰도 $R_0 = \hat{R}(x \mid t)$

유사한 방법으로 로그 파우어 모형에서도 다음과 같이 유도된다.

$$\ln R_0 = -a(\ln^b(1+(x+T_R))-\ln^b(1+T_R)) \quad (27)$$

$$(c_1-c_2)\frac{a\,b\ln^{b-1}(1+T_C)}{1+T_C} + c_3 = 0 \quad (28)$$

4. 제안된 극값 분포모형

이 절에서 극값 분포(Extreme value distribution)[9]모형에 대하

여 제안 하고자 한다. 이 극값 분포는 Gumbel 분포라고도 하는데 여러 표본 중에서 최대값 혹은 최소값을 찾는데 사용되는 분포이고 기후 혹은 날씨를 모형 화 할 때 많이 사용되는 분포로 알려져 있다. 이 극값 분포의 확률 밀도 함수와 분포 함수는 각각 다음과 같다[9]

$$f(t \mid \beta_0, \beta_1) = \exp\left(\beta_0 + \beta_1 t - \frac{e^{\beta_0+\beta_1 t} - e^{\beta_0}}{\beta_1}\right) \quad (29)$$

단, 모수 $-\infty < \beta_0 < \infty$, $\beta_1 > 0, t \geq 0$.

$$F(t \mid \beta_0, \beta_1) = 1 - \exp\left(-\frac{e^{\beta_0+\beta_1 t} - e^{\beta_0}}{\beta_1}\right) \quad (30)$$

(5)식과 (6)식을 이용하여 무한 NHPP로 접근하면 평균값 함수와 강도 함수는 다음과 같이 표현 할 수 있다.

$$m(t) = -\ln(1-F(t)) = \frac{e^{\beta_0+\beta_1 t} - e^{\beta_0}}{\beta_1} \quad (31)$$

$$\lambda(t) = m'(t) = f(t)/(1-F(t)) = \exp(\beta_0 + \beta_1 t) \quad (32)$$

본 논문에서는 극값분포의 특성을 유지하면서도 보다 간결하게 하기 위하여 $\beta_0 = 1$ 으로 가정한 모형을 사용하고자 한다. 한편, (7)식을 이용하면 우도함수는 다음과 같이 유도 할 수 있다.

$$L(\beta_1 \mid \underline{x}) = \left(\prod_{i=1}^{n} \exp(1+\beta_1 x_i)\right) \cdot \exp\{-(e^{1+\beta_1 x_n} - e^1)/\beta_1\} \quad (33)$$

단, $\underline{x} = (x_1 \leq x_2 \leq x_3 \cdots \leq x_n]$.

최우추정법(MLE)을 이용하기 위한 극값모형 로그 우도 함수는 (33)식과 관련하여 다음과 같이 유도된다.

$$\ln L(\beta_0 \mid \underline{x}) = n + (\beta_1 \sum_{i=1}^{n} x_i) - (e^{1+\beta_1 x_n} - e^1)/\beta_1 \quad (34)$$

(34)식에서 β_1에 대하여 편미분 하여 다음과 같은 식을 만족하는 $\hat{\beta}_{1MLE}$ 을 수치 해석적 방법으로 계산할 수 있다.

$$\frac{\partial \ln L(\beta_1 \mid \underline{x})}{\partial \beta_1} = \sum_{i=1}^{n} x_i - \frac{x_n(e^{1+\beta_1 x_n} - e^1)}{\beta_1} + \frac{(e^{1+\beta_1 x_n} - e^1)}{\beta_1^2} = 0 \quad (35)$$

극값 분포 모형에 대한 신뢰도는 (31)식에서$\beta_0 = 1$ 인 경우와 (15) 식을 이용하고 $t = x_n$라고 하면 다음과 같이 표현된다.

$$R(x \mid t) = \exp\left(e^{1+\beta_1 t}(1 - e^{\beta_1 x})/\beta_1\right) \quad (36)$$

따라서 소프트웨어 방출시간 T_R이 신뢰도$R_0 = \hat{R}(x \mid t)$ 을 확보해야 한다면 다음 방정식을 만족해야 한다.

$$\ln R_0 = e^{1+\beta_1 T_R}(1 - e^{\beta_1 x})/\beta_1 \quad (37)$$

한편, 극값 분포 모형 비용 함수 $C(T)$는 다음과 같이 유도된다.

$$\begin{aligned} C(T) &= c_1 m(T) + c_2[m(\infty) - m(T)] + c_3 T \quad (38) \\ &= (c_1 - c_2)\frac{1}{\beta_1}(e^{1+\beta_1 T} - e^1) \\ &\quad + c_2 \frac{1}{\beta_1}(e^{1+\beta_1 T_{LC}} - e^1) + c_3 T \end{aligned}$$

단, c_1는 테스팅 동안에 하나의 고장을 수리하는 비용이고 c_2 가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$), 그리고 c_3는 단위 시간당 테스팅 비용을 나타내고 T_{LC}는 소프트웨어 수명시간을 의미한다. T 에 관해서 비용함수 $C(T)$을 미분하면 다음과 같은 방정식을 만족하는 비용을 고려한 최적방출시간 T_C 를 계산 할 수 있다.

$$(c_1 - c_2)(e^{1+\beta_1 T} - e^1) + c_3 = 0 \quad (39)$$

따라서 극값분포 모형을 사용한 최적 방출시간 T_{OP} 는 T_R과 T_C에 대하여 다음을 만족한다[6].

$$T_{OP} = Max(T_C, T_R) \quad (40)$$

(40) 식에서 T_R과 T_C는 다음 두 방정식에 의해서 계산 될 수 있다.

$$\ln R_0 = e^{1+\beta_1 T_R}(1 - e^{\beta_1 x})/\beta_1 \quad (41)$$

$$(c_1 - c_2)(e^{1+\beta_1 T} - e^1) + c_3 = 0 \quad (42)$$

단, 신뢰도 $R_0 = \hat{R}(x \mid t)$

5. 수치적인 예

이 장에서 고장 간격 시간 자료(Failure interval time data) S27[12]를 가지고 극값모형에 근거한 최적 방출시기를 분석하고자 한다. 이 자료는 1197.945 시간단위에 41번의 고장이 발생된 자료이며 신뢰 모형들을 분석하기 위하여 우선 자료에 대한 추세 검정이 선행 되어야 한다[12]. 추세 분석은 라플라스 추세 검정(Laplace trend test)이 사용된다. (그림 1)의 라플라스 추세 검정의 결과는 라플라스 요인(Factor)이 -2와 2 사이에 존재함으로서 신뢰성장(Reliability growth) 속성을 나타내고 있다. 따라서 이 자료를 이용하여 신뢰도와 소프트웨

어 방출시기를 추정하는 것이 가능하다[12].

소프트웨어 신뢰성 모형의 모수 추정은 최우추정법을 이용하였고 비선형 방정식의 계산방법은 수치 해석적 기본 방법인 이분법(Bisection method)을 사용하였다. 이러한 계산은 초기값을 0와 20을, 허용 한계(Tolerance for width of interval)는 10^{-10} 을 주고 수렴 성을 확인 하면서 충분한 반복 횟수인 100번을 C-언어를 이용하여 모수 추정을 수행하였다. 각 모형에 대한 모수의 추정 값들의 결과는 <표 1>에 요약되었다.

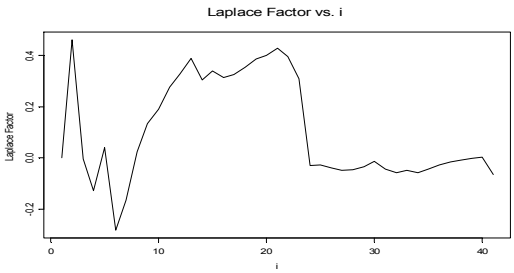

Fig. 1. Laplace trend test
그림 1. 라플라스 추세 검정

Table 1. Parameter estimators of each model
표 1. 각 모형의 모수 추정값

Model	MLE	
PET	$\hat{\theta} = 0.020031$	$\widehat{\lambda_0} = 0.053067$
LP	$\hat{a} = 16.0583379$	$\hat{b} = 0.478584$
ED	$\beta_0 = 1$(고정)	$\widehat{\beta_1} = 8.36 \times 10^{-2}$

주) PET: Log Poission execution time,
LP: Log Power, ED: Extreme Distribution.

<표 2>에서는 비용을 고려한 방출시간에 대하여 추정하였다. 이 표에서 상대적으로 극값모형의 방출시간이 다른 모형에 비해 길지 않기 때문에 효율적으로 나타나고 있지만 <표 3>에 신뢰도를 고려한 방출시간의 추정에 대해서는 로그 포아송 실행시간모형이 효율적으로 나타나고 있다. 그러나 <표 4>에서는 c_1 =10($), c_2 =20($), c_3 =0.5($) 라고 가정하고 시스템 수명시간은 2000시간이고 임무시간 x가 10이고 R_0을 0.95(95%)를 투입한 경우에도 극값모형이 우수한 모형으로 나타나고 있다.

결국 로그 포아송 모형은 비용측면에 민감성을 보이고 있으므로 비용측면에 대한 관리가 필요하고 제안된 극값모형과 로그 파우어 모형은 비용보다는 신뢰도에 상대적으로 민감하기 때문에 신뢰도에 대한 관리가 필요함을 알 수 있다. 그러나 이 결과는 거의 극단값 분포를 따르는 자료를 적합 시킨 결과이고 다른 분포를 따르는 자료를 적합 시키면 결과는 달라질 수 있지만 극값분포 모형도 이 분야에 새로운 모형으로 선택 할 수 있음을 보여주고 있다.

Table 2. Release time based on cost
표 2. 비용을 고려한 방출시간

c_1	c_2	c_3	PET	LP	ED
5	10	0.1	11903.601	163.292	8.727
10	15	0.2	5951.800	86.939	17.390
15	20	0.3	3967.867	60.258	25.991
20	25	0.4	2975.900	46.498	34.530
25	30	0.5	2380.720	38.045	43.009
30	35	0.6	1983.933	32.299	51.427
35	40	0.7	1700.514	28.953	59.787
40	45	0.8	1487.950	24.953	68.088
45	50	0.9	1322.622	22.453	76.331

주) c_1: 테스팅 동안에 하나의 고장을 수리하는 비용,
c_2: 가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$),
c_3: 단위 시간당 테스팅 비용.

Table 3. Release time based on reliability(95%)
표 3. 95% 신뢰도를 고려한 방출시간

Mission Time	PET	LP	ED
5	10.951	306.005	58.087
10	24.529	579.879	76.134
15	38.106	844.467	90.854
20	51.684	1103.478	104.776
25	65.261	1358.547	118.463
30	78.839	1610.589	132.075
35	92.416	1860.187	145.663
40	105.994	2107.744	159.244
45	119.571	2353.556	172.823

Table 4. Optimal release time
표 4. 최적 방출시간

Model	추정시간	최적방출시간
	$\widehat{T_C}$, $\widehat{T_R}$	$\widehat{T_{OP}} = Max(T_C, T_R)$
PET	$\widehat{T_R} = 24.529$, $\widehat{T_C} = 2380.720$	2380.720
LP	$\widehat{T_R} = 579.879$, $\widehat{T_C} = 38.045$	579.879
ED	$\widehat{T_R} = 76.134$, $\widehat{T_C} = 43.009$	76.134

III 결론

본 연구는 최대값 혹은 최소값을 찾는데 사용되는데 효율성이 있다고 알려진 극값 분포를 적용한 무한고장 NHPP 모형을 이용하여 최적 방출시기에 관한 문제를 알아보았다. 즉,

대용량 소프트웨어가 수정과 변경하는 과정에서 결점의 발생을 거의 피 할 수 없는 상황이 현실이다. 실제로 만족할 만한 신뢰도가 부여되고 동시에 시스템 고장과 연계된 기대 총비용을 최소화시키기 위하여 필요하다면 충분한 테스팅을 계속해야 한다. 따라서 신뢰성 요구를 만족하고 총 비용을 최소화하는 상황이 최적 방출 시간이다. 본 연구에서는 극값분포를 적용한 방출시기 모형을 시도한 결과 극값 모형도 이 분야에서 가능한 모형이 될 수 있음을 확인하였다. 이 연구를 통하여 소프트웨어 개발자들은 방출최적시기를 파악 하는데 어느 정도 도움을 줄 수 있으리라 사료 된다.

참고문헌

[1] 간광연, 장병옥, 김희철. “감마족분포를 이용한 소프트웨어 신뢰 성장의 모형의 분석”, 전기전자학회논문지,9권 2호, pp. 65-73, 2005.

[2] Gokhale. S. S. and Trivedi. K. S. “A time/structure based software reliability model”, Annals of Software Engineering. 8, pp. 85-121. 1999.

[3] Musa, J. D and Okumoto, K. “A Logarithmic Poisson Execution Time Model for Software Reliability easurement,” Proceeding the 7th International Conference on Software Engineering. pp. 230-238, 1984.

[4] 김대경. “Musa-Okumoto의 대수 포아송 실행시간 모형에 근거한 비용-신뢰성 최적 정책”. 품질경영학회지, 제26권 3호, pp. 141-149, 1998.

[5] Almering, V. and Genuchten, M, V and Cloudt, G. and Sonnemans, P. J. M. ” Using Software Reliability Growth Models in Practice ”. IEEE SOFTWARE. pp. 82-88, 2007.

[6] Xie, M. and Horng, G. Y. “Software release time determination based on unbound NHPP model”. Proceeding of the 24th International Conference on Computers and Industrial Engineering. pp. 165-168, 1999.

[7] Yang, B. and Xie. M. ”A study of operational and testing reliability in software reliability analysis”. RELIABILITY ENGINEERING & SYSTEM SAFETY, Vol, 70, pp,323-329, 2000.

[8] Huang, C. Y. “Cost-Reliability-optimal release policy for software reliability models incorporating improvements in testing efficiency, The journal of Systems and software. Vol, 77, pp, 139-155, 2005.

[9] Kuo, L. and Yang, T. Y. ”Bayesian Computation of Software Reliability”. Journal of the American Statistical Association, Vol.91, pp.763-773, 1996.

[10] Pham, H. and Nordmann ,L. and Zhang, X. ”A General mperfect-Software -Debugging Model with S-Shaped Fault-Detection Rate”. IEEE Trans. on reliability, Vol, 48, No 2, pp, 169-175, 1999.

[11] Musa, J. D, Iannino, A. and Okumoto, K. “Software Reliability: Measurement, Prediction, Application” McGraw Hill, New York, 1987.

[12] K. Kanoun , J. C. Laprie. Handbook of Software Reliability Engineering, M.R.Lyu, Editor, chapter Trend Analysis. McGraw-Hill New York, NY: 1996; p.401-437.

저 자 소 개

김 희 철 (정회원)

1992년 : 동국대학교 대학원 통계학과 (이학석사)

2008년 : 동국대학교 대학원 통계학과 (이학박사)

2005년 3월~현재 : 남서울대학교 산업경영공학과 교수

<주관심분야> 소프트웨어 신뢰성 프로그래밍, 전산통계, 해외투자 및 쇼핑몰