

고장 간격 수명 시간을 이용한 로그 포아송 실행 시간 모형의 소프트웨어 최적방출시간 특성에 관한 연구

신현철* · 김희철**

요 약

소프트웨어 제품을 개발하여 테스트를 거친 후 사용자에게 인도하는 시기를 결정하는 일은 아주 실제적이고 흥미 있는 일이 된다. 이러한 문제를 소프트웨어 최적 방출시기라고 한다. 본 연구에서는 소프트웨어의 결함을 제거하거나 수정 작업 중에도 새로운 결함이 발생할 가능성이 있는 무한고장수를 가진 비동질적인 포아송 과정에 기초하고 고장간격 수명 시간의 형태에 따른 방출시간의 비교에 대하여 연구 되었다. 소프트웨어 요구 신뢰도를 만족시키고 소프트웨어 개발 및 유지 총비용을 최소화 시키는 최적 소프트웨어 방출 정책이 된다. 본 논문의 수치적인 예에 서는 모의실험 자료 즉, 강도함수가 일정한 경우, 증가하는 경우, 감소하는 경우를 적용하여 최적 방출시기를 추정하고 그 결과를 나열 하였다.

The Property of Software Optimal Release Time Based on Log Poission Execution Time Model Using Interval Failure Times

Hyun-Cheul Sin* · Hee-Cheul Kim**

ABSTRACT

It is of great practical interest to deciding when to stop testing a software system in development phase and transfer it to the user. This decision problem called an optimal release policies. In this paper, because of the possibility of introducing new faults when correcting or modifying the software, we were researched release comparative policies which based on infinite failure NHPP model and types of interval failure times. The policies which minimize a total average software cost of development and maintenance under the constraint of satisfying a software reliability requirement can optimal software release times. In a numerical example, applied data which were patterns, if intensity function constant or increasing, decreasing, estimated software optimal release time.

Key words : Software Release Policies, Log Poission Execution Time Model), Infinite Failure Mean Value Function

접수일 : 2010년 3월 2일; 채택일 : 2010년 3월 10일

* 백석문화대학 컴퓨터정보학부

** 남서울대학교 산업경영공학과

1. 서 론

소프트웨어 신뢰성은 컴퓨터 시스템에 대한 적용과 이에 대한 연구 분야에서 중요한 역할을 담당해 오고 있다. 소프트웨어 고장으로 인한 컴퓨터 시스템의 고장은 우리 사회에 엄청난 손실을 유발할 수도 있다. 따라서 소프트웨어 신뢰도는 현대의 소프트웨어 생산품 개발에서 중요한 문제 가운데 하나이다.

소프트웨어 신뢰도 엔지니어링에서의 연구 활동은 지난 30년 전부터 행해져 오고 있고 많은 신뢰도 성장 모형들이 소프트웨어에 남아 있는 고장들의 수와 소프트웨어 신뢰도의 추정에 관한 문제들을 제안해 왔다. 일반적으로 소프트웨어 개발과정은 설계단계, 디자인, 코딩 그리고 테스트 단계를 거친다. 이러한 과정을 거친 후 소프트웨어 제품을 방출하게 되는데 방출이후에 발견되지 않은 고장들이 나타난다면 이것들에 대한 보전 비용(Maintenance cost)은 크게 증가할 것이다. 결국, 소프트웨어 시스템 시험을 끝내고 그것을 사용자에게 넘기는 시기 결정은 매우 중요한 사항이 된다. 이러한 소프트웨어 방출시간에 대한 연구들은 대부분 유한 고장 NHPP(Non-Homogeneous Poisson Process)모형을 사용하였다[1-4]. 이러한 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 발생할 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[5]과 Log-power 모형[6]을 이용한 방출 문제에 대한 문제들이 이미 연구되었고 최근에도 이와 관련된 문제에 대한 연구는 방출시간 이후의 할인율과 사후 대책에 대한 연구가 Yang and Xie(2000)와 Huang(2005)에 의해 연구되고 있다[7, 8].

본 연구에서는 대수 포아송 실행시간 모형의 적용에 대한 문제 즉, 강도함수가 감소추세인 경우와 일정한 경우 그리고 증가하는 경우를 적용한 무한 고장 NHPP 모형을 이용하여 최적 방출시기에 관한 문제를 다루고자 한다. 본 논문의 제 2장에서는 관련 연구로서 무한고장과 기록 값 통계량 및 모수 추정에 대하여 약술하였고 제 3장에서는 요구 신뢰도와 비용 최소화를 고려한 방출시간에 대하여 서술하고 제 4장과 제 5장에서는 각각 수치적인 예와 그 결론을 나열 하였다.

2. 관련 연구

2.1 무한 고장 NHPP와 기록값 통계량

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

$$m(t) = \int_0^t \lambda(s) ds, \quad \frac{dm(t)}{dt} = \lambda(t) \quad (2.1)$$

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

$$P\{N(t) = n\} = \frac{[m(t)]^n \cdot e^{-m(t)}}{n!}, \quad (2.2)$$

$$n = 0, 1, 2, \dots, \infty$$

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 이러한 NHPP 모형들은 유한 고장 모형과 무한 고장 범주로 분류한다[10]. 유한 고장 NHPP 모형들은 충분한 테스트 시간이 주어지면

결함들(Faults)의 기대값이 유한 값 $(\lim_{t \rightarrow \infty} m(t) = \theta < \infty)$ 을 가지고 반면에 무한 고장 NHPP 모형들은 무한 값을 가진다고 가정된다[10].

일반적으로 일반화된 순서 통계량(Generalized Order Statistics : GOS) 모형을 이용한 모형은 소프트웨어 테스트 시점에서 미지의 N 개의 결함들을 가지고 있고 이 N 개의 결함들로부터 임의의 확률 밀도 함수(Pdf)에 따라 발생한 n 개의 순서 통계량이 고장 시점이 된다. 이 모형은 각 수리 시점에서 새로운 결함이 발생하지 않는다는 가정을 한다. 그러나 실제 상황에서는 수리 시점에서도 고장이 발생할 수도 있다. 이러한 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 평균값 함수는 다음과 같이 된다고 하였다[4].

$$m(t) = -\ln(1 - F(t)) \tag{2.3}$$

즉, $e^{-m(t)} = 1 - F(t)$.

따라서 식 (2.1)과 식 (2.3)와 연관시키고 $f(t)$ 을 확률밀도함수, $F(t)$ 을 분포함수라고 하면 다음과 같은 관계식에 의해 NHPP의 강도함수는 $F(t)$ 의 위험함수($h(t)$)가 된다.

$$\lambda(t) = m'(t) = f(t)/(1 - F(t)) = h(t) \tag{2.4}$$

결국 무한 고장 NHPP 모형의 평균값 함수와 고장 강도 함수는 각각 다음과 같이 유도된다[11, 12].

$$m(t) = -\ln(1 - F(t)) \tag{2.5}$$

$$\lambda(t) = m'(t) = f(t)/(1 - F(t)) = h(t) \tag{2.6}$$

시간 $(0, t]$ 까지 조사하기 위한 시간 절단(Time truncated) 모형은 n 번째 까지 고장 시점 자료를

$$x_k = \sum_{i=1}^k t_k \tag{2.7}$$

$(k=1, 2, \dots, n; 0 \leq x_1 \leq x_2 \leq \dots \leq x_n)$

이라고 하고 θ 을 모수공간이라고 하면 우도 함수는 다음과 같이 알려져 있다[10, 11].

$$L_{NHPP_{\text{eff}}}(\theta | D_l) = \prod_{i=1}^n \left(\frac{f(x_i)}{1 - F(x_i)} \right) (1 - F(x_n)) = \left(\prod_{i=1}^n h(x_i) \right) (1 - F(x_n)) \tag{2.8}$$

2.2 로그 포아송 실행 시간 모형

로그 포아송 실행시간(Log Poission execution time : PET)모형[5, 11]은 1984년에 Musa와 Okumoto에 의해서 소개된 무한 고장 소프트웨어 모형으로 평균값함수와 강도함수는 다음과 같이 알려져 있다.

$$m(t) = \frac{1}{\theta} \ln(\lambda_0 \theta t + 1) \tag{2.9}$$

$$\lambda(t) = \frac{\lambda_0}{\lambda_0 \theta t + 1} \tag{2.10}$$

한편, 식 (2.3)과 식 (2.8)을 이용하면 우도함수는 다음과 같이 유도 할 수 있다.

$$L(\theta, \lambda_0 | \underline{x}) = \left(\prod_{i=1}^n \frac{\lambda_0}{\lambda_0 \theta x_i + 1} \right) \cdot e^{-\frac{1}{\theta} \ln(\lambda_0 \theta x_n + 1)} \tag{2.11}$$

단, $\underline{x} = (x_1, x_2, x_3, \dots, x_n)$.

모수 최우추정법(MLE)을 이용하기 위한 로그 우도 함수는 다음과 같이 유도된다.

$$\ln L(\theta, \lambda_0 | \underline{x}) = n \ln \lambda_0 - \ln \sum_{i=1}^n (\lambda_0 \theta x_i + 1) - \frac{1}{\theta} \ln(\lambda_0 \theta x_n + 1) \tag{2.12}$$

식 (2.12)을 이용하여 최우추정치 $\hat{\theta}_{MLE}$ 와 $\hat{\lambda}_{0MLE}$ 은 다음과 같이 구할 수 있다고 하였다[5, 6]

$$\hat{\theta}_{MLE} = \frac{1}{n} \ln(\hat{\phi} x_n + 1) \quad (2.13)$$

$$\hat{\lambda}_{0MLE} = \hat{\phi} / \hat{\theta}_{MLE} \quad (2.14)$$

단, $\phi (= \hat{\lambda}_{0MLE} \cdot \hat{\theta}_{MLE})$ 는 (2.15) 식의 근이 된다.

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

$$\frac{\partial \ln L(\phi | x)}{\partial \phi} = \frac{n}{\phi} - \sum_{i=1}^n \frac{x_i}{\phi x_i + 1} - \frac{n x_n}{(\phi x_n + 1) \ln(\phi x_n + 1)} = 0 \quad (2.15)$$

3. 신뢰도 및 비용 최소화

NHPP 모형에서 테스트 시점 x_n (마지막 고장시점)에서 소프트웨어 고장이 일어난다고 하는 가정 하에서 신뢰구간 $(x_n, x_n + x)$ (단, x 는 임무시간(Mission time))사이에서 소프트웨어의 고장이 일어나지 않을 확률인 신뢰도 $\hat{R}(x | x_n)$ 는 다음과 같이 될 수 알려져 있다[12].

$$\hat{R}(x | x_n) = \exp\left(-\int_{x_n}^{x_n+x} \lambda(\tau) d\tau\right) = \exp\{-m(x+x_n) - m(x_n)\} \quad (3.1)$$

따라서 로그 포아송 실행시간모형에 대한 신뢰도는 평균값함수 식 (2.9)와 $t = x_n$ 을 이용하면 다음과 같이 표현된다[5, 6].

$$R(x | t) = \exp\left(-\frac{1}{\theta} [\ln(\lambda_0 \theta(x+t)+1) - \ln(\lambda_0 \theta t + 1)]\right) \quad (3.2)$$

따라서 소프트웨어 방출시간 T_R 이 신뢰도 $R_0 = \hat{R}(x | t)$ 을 확보해야 한다면 다음 방정식을 만족해야 한다(방정식의 해).

$$\ln R_0 = -\frac{1}{\theta} [\ln(\lambda_0 \theta(x+T_R)+1) - \ln(\lambda_0 \theta T_R + 1)] \quad (3.3)$$

비용 최소화와 관련된 최적 방출시간은 신뢰도와 함께 비용모형에 의해서 결정된다. 소프트웨어 방출시간을 T 로 표현하고 $m(T)$ 와 $m(\infty)$ 을 각각 $(0, T)$ 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 표현된다[7, 8].

$$C(T) = c_1 m(T) + c_2 [m(\infty) - m(T)] + c_3 T \quad (3.6)$$

위 식에서 c_1 는 테스트 동안에 하나의 고장을 수리하는 비용이고 c_2 가동 중에 하나의 고장을 수리하는 비용($c_2 > c_1$), 그리고 c_3 는 단위 시간당 테스트 비용을 나타낸다. 이와 관련하여 총비용의 최소화는 무한고장 평균값 함수를 가진 NHPP 모형에 대하여 발생 할 수 있다. 무한 수명에 대한 비용함수 $C(T)$ 식인 식 (3.3)과 식 (3.6)에서 $m(\infty)$ 은 직접 구체화 되어 있지 않기 때문에 이 식을 사용하기 위해서는 소프트웨어 수명시간 인 T_{LC} 을 지정하여 분석한다[6]. 이러한 T_{LC} 는 소프트웨어마다 서로 다른 임의의 값이기 때문에 유한 고장 NHPP 모형이라고 할 수는 없다.

따라서 비용함수를 고려하여 소프트웨어의 모든 수명에서 총비용을 최소화함으로써 최적 테스트 시간을 결정 할 수 있고 다음과 같은 식을 만족하면 비용함수 $C(T)$ 는 유일한 최소값을 가진다[7].

$$\frac{dC(T)}{dT} = 0, \quad \frac{d^2 C(T)}{d^2 T} > 0 \quad (3.7)$$

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

$$C(T) = c_1 m(T) + c_2 [m(\infty) - m(T)] + c_3 T \quad (3.8)$$

$$= (c_1 - c_2) \frac{1}{\theta} \ln(\lambda_0 \theta T + 1)$$

$$+ \frac{c_2}{\theta} \ln(\lambda_0 \theta T_{LC} + 1) + c_3 T$$

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

$$\frac{(c_1 - c_2)\lambda_0}{\lambda_0 \theta T_C + 1} + c_3 = 0 \quad (3.9)$$

위 식에서 최적방출시간은 소프트웨어 지정 수명 시간인 T_{LC} 와 의존하지 않는다는 것을 알 수 있다. 이러한 사실은 무한 고장 평균값 함수를 가진 NHPP 모형들이 새로운 결점들이 발생함으로써 몇 개의 고장이 야기 될 수 있는 점을 고려한 모형으로 적합시킬 수 있다[5]. 실제로 만족할 만한 신뢰도가 부여되고 동시에 시스템 고장과 연계된 기대 총비용을 최소화시키기 위하여 필요하다면 충분한 테스트를 계속해야 한다. 따라서 신뢰성 요구를 만족하고 총 비용을 최소화하는 상황이 최적 방출 시간이다. 따라서 로그 포아송 실행시간모형을 사용한 최적 방출시간 T_{OP} 는 T_R 과 T_C 에 대하여 다음을 만족한다[8].

$$T_{OP} = \text{Max}(T_C, T_R) \quad (3.10)$$

식 (3.9)에서 T_R 과 T_C 는 다음 두 방정식에 의해서 계산 될 수 있다.

$$\ln R_0 = -\frac{1}{\theta} [\ln(\lambda_0 \theta (x + T_R) + 1) - \ln(\lambda_0 \theta T_R + 1)] \quad (3.11)$$

$$\frac{(c_1 - c_2)\lambda_0}{\lambda_0 \theta T_C + 1} + c_3 = 0 \quad (3.12)$$

단, 신뢰도 $R_0 = \hat{R}(x | t)$

4. 수치적인 예

이 장에서는 모의자료를 이용하고자 한다. 모의 고장 시간 자료는 통계 패키지인 Minitab 14 버전 [13]에서 척도모수($\beta=5$)를 고정하고 강도함수가 감소추세인 경우($\alpha=0.5$)와 일정한 경우($\alpha=1$) 그리고 증가하는 경우($\alpha=2$)을 랜덤 추출하여 자료를 완성하였다.

모의자료 30번의 고장이 발생된 자료이며 <표 1>에 나열 되어 있고 로그 포아송 실행 모형에 대한 모수 추정은 최우추정법을 이용하였고 비선형 방정식의 계산방법은 수치해석적 기본 방법인 이분법(Bisection method)을 사용하였다. 이러한 계산은 초기값을 10^{-2} 와 10을, 허용 한계(Tolerance for width of interval)는 10^{-10} 을 주고 수렴성을 확인 하면서 충분한 반복 횟수인 100번을 C-언어를 이용하여 모수 추정을 수행하였다. 각 모형에 대한 모수의 추정값들의 결과는 <표 2>에 요약되었다.

<표 3>에서는 $c_1=10(\$)$, $c_2=100(\$)$ 그리고 $c_3=3(\$)$ 라고 가정하고 시스템 수명시간은 2000시간이고 임무시간을 x 을 1.0이고 R_0 을 0.95(95%)를 투입하여 수치해석적 기본 방법인 이분법(Bisection method)을 사용하였다. 이러한 계산은 초기값을 20와 10000을, 허용 한계(Tolerance for width of interval)는 10^{-10} 을 주고 수렴성을 확인 하면서 충분한 반복 횟수인 100번을 C-언어를 이용하여 추정을 수행하였다. 각 모형에 대한 추정시간의 결과와 최적방출시간은 <표 3>에 요약되었다.

같은 조건하에서 가동 중에 하나의 고장을 수리

<표 1> 모의 고장 간격 및 시간 자료

고장 시점	$\alpha=0.5$	고장 시간	$\alpha=1$	고장 시간	$\alpha=2$	고장 시간
x_1	0.43	0.43	4.40	4.40	2.81	2.81
x_2	0.39	0.82	3.71	8.11	4.47	7.27
x_3	9.64	10.46	1.06	9.17	5.03	12.30
x_4	2.81	13.27	2.12	11.29	1.95	14.25
x_5	17.11	30.38	0.97	12.26	3.66	17.91
x_6	0.01	30.39	11.93	24.19	1.61	19.52
x_7	0.09	30.49	17.32	41.51	5.53	25.04
x_8	3.06	33.54	17.94	59.44	3.29	28.33
x_9	2.27	35.82	5.53	64.97	4.03	32.36
x_{10}	0.55	36.37	1.97	66.95	2.91	35.28
x_{11}	23.87	60.23	0.29	67.24	4.90	40.18
x_{12}	9.28	69.51	0.64	67.88	2.99	43.17
x_{13}	81.35	150.85	4.08	71.95	2.67	45.84
x_{14}	0.20	151.06	6.78	78.73	3.36	49.20
x_{15}	3.43	154.49	0.91	79.64	5.47	54.66
x_{16}	12.68	167.17	4.19	83.83	2.08	56.74
x_{17}	3.74	170.91	3.31	87.14	10.17	66.90
x_{18}	9.06	179.97	8.73	95.86	6.38	73.28
x_{19}	5.65	185.62	3.59	99.46	3.23	76.52
x_{20}	1.02	186.64	10.49	109.95	3.23	79.74
x_{21}	0.52	187.16	1.67	111.62	1.98	81.72
x_{22}	0.75	187.91	3.44	115.06	4.44	86.17
x_{23}	1.36	189.28	7.57	122.63	3.96	90.13
x_{24}	0.05	189.33	2.87	125.50	1.33	91.45
x_{25}	2.54	191.86	16.81	142.31	2.81	94.27
x_{26}	0.66	192.52	1.62	143.93	9.30	103.57
x_{27}	14.16	206.69	0.17	144.10	1.54	105.10
x_{28}	0.02	206.71	1.91	146.01	4.77	109.87
x_{29}	1.85	208.56	3.10	149.11	6.28	116.15
x_{30}	41.99	250.55	7.51	156.62	5.64	121.80

<표 2> Log Poission Execution모형의 모수 추정값

속성	MLE	
$\alpha = 0.5$	$\hat{\theta} = 0.00344$	$\hat{\lambda}_0 = 0.12613$
$\alpha = 1$	근이 수렴하지 않음	
$\alpha = 2$	$\hat{\theta} = 0.00507$	$\hat{\lambda}_0 = 0.08553$

<표 3> 최적 방출시간($c_1 = 10, c_2 = 100, c_3 = 3$ 인 경우)

속성	\hat{T}_R (추정시간)	\hat{T}_C (추정시간)	최적방출시간
$\alpha = 0.5$	3362.116	4111.437	4111.437
$\alpha = 2$	1506.450	1305.008	1506.450

<표 4> 최적 방출시간($c_1 = 10, c_2 = 90, c_3 = 3$ 인 경우)

속성	\hat{T}_R (추정시간)	\hat{T}_C (추정시간)	최적방출시간
$\alpha = 0.5$	3362.116	3142.445	3362.116
$\alpha = 2$	1506.450	647.546	1506.450

하는 비용인 c_2 를 90으로 감소시킨 결과는 <표 4>에 요약 되었다. <표 3>과 <표 4>에서 최적방출시간은 강도함수가 감소추세인 경우($\alpha=0.5$)인 경우는 비용관련 방출시기가 되고 증가하는 경우($\alpha=2$ 는 신뢰도 관련 방출시기가 최적 방출시간이 된다. 전체적으로는 강도함수가 감소추세인 경우보다 강도함수가 증가추세인 경우가 상대적으로 최적 방출시기가 길지 않기 때문에 더 효율적임을 알 수 있다. 따라서 로그 포아송 실행 모형에서 강도함수가 증가추세인 경우 즉, 고장률이 증가추세가 증가하는 추세를 가진 데이터에서는 효율적 모형이 됨을 알 수 있었다.

5. 결 론

대용량 소프트웨어가 수정과 변경하는 과정에서 결점의 발생을 거의 피할수 없는 상황이 현실이다. 실제로 만족할 만한 신뢰도가 부여되고 동시에 시스템 고장과 연계된 기대 총 비용을 최소화시키기 위하여 필요하다면 충분한 테스트를 계속해야 한다. 따라서 신뢰성 요구를 만족하고 총 비용을 최소화하는 상황이 최적 방출 시간이다. 본 연구에서는 로그 포아송 실행 모형에서 강도함수가 감소하는 경우와 증가하는 추세를 적용하여 최적 방출시

기를 비교한 한 결과 강도함수가 감소추세보다는 증가추세를 가진 경우가 효율적임을 확인 할 수 있었다. 경우에 따라서는 왜도와 첨도 측면에서 효율적인 카파분포, 지수화지수분포 등 업데이트된 분포에 대한 방출 시기 문제를 비교 분석하는 연구도 가치 있는 일이라 판단되고 이 연구를 통하여 소프트웨어 개발자들은 방출최적시기를 파악 하는데 어느 정도 도움을 줄 수 있으리라 사료 된다.

참 고 문 헌

[1] J. F. Lawless, "Statistical Models and Methods for Lifetime Data", John Wiley and Sons, New York, 1981.

[2] 김희철, "일반화감마분포를 이용한 NHPP 소프트웨어 신뢰도 모형에 관한 연구", 한국컴퓨터정보학회논문지, 제10권, 제6호, pp. 27-35, 2005.

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

[4] 김희철, 신현철, 김경수, "기록값 통계량에 기초한 무한고장 NHPP 소프트웨어혼합 신뢰성장 모형에 관한 연구", 정보보안논문지, 제7권, 제3호, pp. 51-60, 2007.

[5] 김대경, "Musa-Okumoto의 대수 포아송 실행 시간 모형에 근거한 비용-신뢰성 최적 정책", 품질경영학회지, 제26권, 제3호, pp. 141-149, 1998.

[6] M. Xie, and G. Y. Homg, "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] B. Yang and M. Xie, "A study of operational and testing reliability in software reliability analysis", Reliability Engineering and System Safety, Vol. 70, pp. 323-329, 2000.

[8] C. Y. Huang, "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.

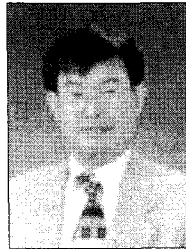
[9] 김희철, 박종구, "Burr분포를 이용한 NHPP 소프트웨어 신뢰성장모형에 관한 연구", 한국해양정보통신학회논문지, 제11권, 제3호, pp. 514-522, 2007.

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

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

[12] 김희철, "Burr 커버리지 함수에 기초한 ENHPP 소프트웨어 신뢰성장모형에 관한 연구" 한국컴퓨터정보학회논문지, 제12권, 제4호, pp. 33-42, 2007.

[13] 김희철, "Minitab과 함께하는 공업통계학의 기초 및 활용", 한울출판사, 2008.



신 현 철

2002년 원광대학교 컴퓨터공학과 (공학박사)
2004년 (주) 아이비루션 자문위원
1994년~현재 백석문화대학 컴퓨터정보학부 교수
관심분야 : 통신공학, 컴퓨터 과학, 소프트웨어 공학



김 희 철

1998년 동국대학교 통계학과 (이학박사)
2000년~2004년 송호대학 정보 산업계열 교수
2005년~현재 남서울대학교 산업경영공학과 교수
관심분야 : 소프트웨어 신뢰성 공학, 웹프로그래밍