보증기간을 고려한 최적 소프트웨어의 보전정책 연구

남경현 · 김도훈†

경기대학교 응용정보통계학과

A Study on Optimal Software Maintenance Policies with Warranty Period

Kyung H. Nam · Do Hoon Kim†

Department of Applied Information Statistics, Kyonggi University

Key Words : Nonhomogeneous Poisson Process, Software Reliability Growth Model, Mean Value Function, Software Release Time, Warranty Period, Total Expected Software Cost

Abstract

In general, a software fault detection phenonenon is described by a software reliability model based on a nonhomogeneous Poisson process(NHPP). In this paper, we propose a software reliability growth model considering the differences of the software environments in both the testing phase and the operational phase. Also, we consider the problem of determining the optimal release time and the optimal warranty period that minimize the total expected software cost which takes account of periodic software maintenance(e.g. patch, update, etc). Finally, we analyze the sensitivity of the optimal release time and warranty period based on the fault data observed in the actual testing process.

1. 서 론

현재와 같은 정보화 사회에서 컴퓨터 시스템인 하드웨어의 기능 및 중요성은 나날이 증가하고 있으며, 이에 따라 운영시스템, 제어시스템 등 컴퓨터 시스템을 제어하는 소프트웨어의 역할 또한 중요한 부분을 차지하고 있다. 이에 따라 하드웨어뿐만 아니라 소프트웨어 제품의 고품질화가 계속적으로 요구되고 있는 상황에서 소프트웨어 신뢰성은 극히 중요한 연구 영역으로 다루어지고 있다. 소프트웨어 신뢰성의 향상은 소프트웨어의 품질에 대한 관심으로 돌려졌으며, 소프트웨어의 품질을 향상시키기 위한 많은 연구가 진행되고 있다. 즉, 소프트웨어 구성 모듈을 포함한 제품 전체의 품질보증, 제품의 평균수명의 측정, 그리고 소프트웨어 제품이 일정기간 고장 없이 기능을 발휘하게 하기 위한 품질관리활동 등 신뢰성과 관련된 품질요인들이 큰 비중을 차지하고 있다.

소프트웨어 신뢰성 평가는 계획된 명세서대로 소프트웨어 기능이 제대로 이행되는지를 정량적으로 평가하기 위한 통계적 기법이다. 개발단계 중 시험단계에서의 소프트웨어 신뢰도를 정량적으로 평가하기 위한 측도는 많이 개발되어 있으나 운용단계에서의 신뢰도 평가를 위한 연구는 아직 많은 연구가 이루어져 있지 않은 상태이다.

소프트웨어 제품의 신뢰성을 평가할 때 시험단계(testing phase)와 운용단계(operational phase) 상에서의 차이는 엄격히 구별되어야 한다. 시험단계의 신뢰성평가는 시험과정에서 관측된 결함 탐지 메커니즘의 통계적 특성을 기초로 평가가 수행되지만, 운용단계의 신뢰성평가는 시험노력과 시험방법 등의 시험단계에서의 환경과 사용빈도와 같은 운용단계에서의 환경과의 차

† 교신저자 dhkim1125@kyonggi.ac.kr

※ 본 논문은 2008학년도 경기대학교 학술연구비(일반연구과제) 지원에 의하여 수행되었음.

이를 예측함에 의해 수행된다. 그러나 규정할 수 없는 수많은 사용자가 운용단계에서 소프트웨어를 사용하게 되며, 이러한 환경의 차이를 예측하는 것이 쉽지 않다는 데 있다. 결과적으로 소프트웨어 신뢰성을 평가하기 위해 시험단계에서 계측된 통계적 데이터를 이용함과 동시에 시험단계와 운용단계의 실질적인 환경간 차이를 규명함으로써 보다 현실적인 모형제안이 요구되고 있다.

Yamada, Tanio and Osaki(1989, 1991)는 시험단계와 운용단계와의 차이를 구별함으로써 얻어지는 평균값 함수를 도출하였으며, 와이블 과정이라 불리는 NHPP에 기초한 평균값 함수를 유도하였다. 또한 Yamada (1993)는 Norden/Rayleigh manpower curve를 이용한 평균값 함수를 갖는 결함 발견 과정을 제안하였다.

최규식과 김용경(2003)은 시험단계와 운용단계의 신뢰도를 범용소프트웨어와 전용소프트웨어로 구분하여 비교·분석을 하였으며, Rinsaka and Dohi(2005, 2006)는 소프트웨어 수명주기에서 시험단계와 운용단계의 차이를 고려하여 소프트웨어 신뢰도 성장모형을 제안하고, 최적의 소프트웨어 출시시점과 보증기간을 결정하였다. 그러나 기존의 연구에서는 시험단계와 운용단계의 환경차이를 고려하였으나, 시험단계의 순간강도함수의 패턴이 운용단계에서도 계속 유지된다는 가정 하에 모형을 전개하였다. 실제로 운용단계에서는 계속적인 보전활동을 통하여 소프트웨어를 유지·보수하고 있으며, 범용소프트웨어의 경우 패치(patch)나 업데이트(update)를 통한 주기적 보전을 실시하고 있다. 따라서 이러한 운용단계의 보전활동을 고려한 현실적인 소프트웨어 신뢰도 성장모형의 개발이 필요하다.

본 논문의 목적은 비동질적 포아송과정(nonhomogeneous Poisson process : NHPP)을 활용하여 시험단계(testing phase)와 운용단계(operational phase)에서의 환경차이를 고려하여 소프트웨어 신뢰도 성장모형(software reliability growth model, SRGM)을 제안하며, 소프트웨어 중 범용소프트웨어에서의 결함발견과정에 대한 SRGM을 고려한다. 또한 제안된 SRGM을 이용하여 소프트웨어 수명주기 상에서 발생할 수 있는 총기대비용을 유도하며 최적의 소프트웨어 출시시점 및 보증기간 결정을 목적으로 한다. 마지막으로 수치예제를 통하여 시험단계의 결함발견 자료를 기초로 최적의 출시시점 및 보증기간을 결정하며, 여러 모수의 변화에 따른 민감도 분석을 실시한다.

기 호

$N(t)$	시간 t까지 발견된 누적 결함수
$m(t)$	$E[N(t)]$, 평균값 함수
$\lambda(t)$	시간 t에서의 순간 고장강도함수
$\lambda_{pm}(t)$	운용단계(보증기간)에서 보전을 고려한 순간 고장강도함수
a	초기 결함수
b	결함 발견율
t_0	소프트웨어 출시시점
t_w	소프트웨어 보증기간
t_L	소프트웨어 수명주기
c_0	시험단계에서 소프트웨어 결함제거 비용
c_w	보증기간동안 소프트웨어 결함제거 비용
c_L	보증기간이후 소프트웨어 결함제거 비용
c_{t0}	시험단계에서 단위시간당 시험비용
c_{tw}	보증기간동안 단위시간당 시험비용
c_p	단위보전비용
N	보증기간동안 실시된 보전횟수
$EC(t_0, t_w)$	소프트웨어 총기대비용

2. 소프트웨어 신뢰도 성장 모형

2.1 기본가정 및 모형설정

소프트웨어 시스템을 개발하는 과정은 매우 복잡하여 개발과정 동안 예측할 수 없는 결과가 발생한다. 이러한 소프트웨어 시스템의 품질을 측정하고 평가하는 일이 소프트웨어 개발에서 중요한 문제로 대두되고 있으며, 소프트웨어 품질보증 및 보전에 대한 관심도 점차 증대되고 있다. 일반적으로 소프트웨어 개발 단계는 명세서(specification), 설계(design), 코딩(coding), 시험(testing)의 네 단계로 구성되어 있다. 소프트웨어 결함은 소프트웨어 개발의 마지막 단계인 시험단계(testing phase)에서 탐지되고 수정되며 이러한 결함 발견현상은 소프트웨어 신뢰도 성장모형(SRGM)으로서 기술할 수 있다.

지난 20여년 동안 다양한 통계적 모형들이 소프트웨어 신뢰성을 평가하기 위한 목적으로 제안되어 왔다. 비동질적 포아송과정(NHPP)에 기초한 소프트웨어 신뢰도 성장모형(SGRM)은 실제적인 소프트웨어 신뢰성 공학 면에서 매우 성공적인 이론이라는 것이 증명되었다(Musa et al., 1987). NHPP 상에서 주요 논점은 특정 시점까지의 기대고장수를 나타내는 적절한 평균값 함수(mean value function : mvf)를 결정하는 것이다.

<표 1> 소프트웨어 신뢰도 성장모형과 평균값함수

소프트웨어 신뢰도 성장모형	모형 형태	평균값함수 $m(t)$	비 고
Goel-Okumoto(G-O) (1979)	Concave	$m(t)=a(1-e^{-bt})$ $a(t)=a$ $b(t)=b$	a : 탐지될 기대 결함수 b : 결함발견율
Delayed S-shaped SRGM(1983)	S-shaped	$m(t)=a(1-(1+bt)e^{-bt})$ $a(t)=a$ $b(t)=\frac{b^2t}{1+bt}$	$a(t)$: 탐지될 기대결함함수 $b(t)$: 결함발견 함수
Yamada imperfect debugging model 1 (1992)	Concave	$m(t)=\frac{ab}{\alpha+b}(e^{\alpha t}-e^{-bt})$ $a(t)=ae^{\alpha t}$ $b(t)=b$	α : 결함도입율
Yamada imperfect debugging model 2 (1992)	Concave	$m(t)=a[1-e^{-bt}](1-\frac{\alpha}{b})+a\alpha t$ $a(t)=a(1+\alpha t)$ $b(t)=b$	
Chang change-point model(1997)	Concave	$m(t)=\begin{cases}a(1-e^{-b_1t}), & 0\le t\le\tau,\\ a(1-e^{-b_1\tau-b_2(t-\tau)}), & t>\tau.\end{cases}$ $a(t)=a$ $b(t)=\begin{cases}b_1, & 0\le t\le\tau,\\ b_2, & t>\tau.\end{cases}$	τ : 변화점
Pham-Zhang coverage model (2003)	S-shaped and concave	$m(t)=a\left(1+\alpha t-\frac{bt+1}{e^{bt}}\right)-\frac{a\alpha(1+bt)}{be^{bt+1}}\left[\ln(bt+1)+\sum_{i=1}^{\infty}\frac{(1+bt)^{i+1}-1}{(i+1)!(i+1)}\right]$ $a(t)=a(1+\alpha t)$ $b(t)=1-(1+bt)e^{-bt}$	α : 결함도입율
Shyur imperfect change-point model(2003)	Concave	$m(t)=\begin{cases}\frac{a}{1-\alpha_1}[1-e^{-(1-\alpha_1)b_1t}], & 0<t<\tau\\ \frac{a}{1-\alpha_2}[1-e^{-(1-\alpha_1)b_1\tau-(1-\alpha_2)b_2(t-\tau)}]\\ +\frac{m(\tau)(\alpha_1-\alpha_2)}{1-\alpha_2}, & t>\tau\end{cases}$ $a(t)=a$ $b(t)=\begin{cases}b_1, & 0\le t\le\tau,\\ b_2, & t>\tau.\end{cases}$	τ : 변화점

지금까지의 다양한 NHPP SRGM들은 모형에 맞는 다양한 가정들로 이루어져 왔다. <표 1>은 다양한 가정들로 이루어진 SRGM들과 평균값 함수를 정리한 표이다.

$\{N(t), t\ge 0\}$를 시간 t까지 발견된 누적 결함수를 나타내는 계수과정(counting process)이라 하자. NHPP의 평균값함수 $m(t)$를 이용하여 SRGM은 다음과 같이 나타낼 수 있다.

$$\Pr\{N(t)=n\}=\frac{\{m(t)\}^n}{n!}\exp\{-m(t)\},\ (n=0,1,2,\cdots). \quad (2.1)$$

여기서, $m(t)$를 NHPP의 평균값함수라 하고, 시간 t

까지 발견된 기대누적 결함수를 나타낸다.

또한 순간 고장강도함수 $\lambda(t)$는 다음과 같이 정의된다.

$$\lambda(t)=\frac{dm(t)}{dt}. \quad (2.2)$$

여러 SRGM 중 Goel-Okumoto(1979)는 NHPP를 기초로 한 SRGM에 관하여 논의하였다. 평균값 함수(mean value function)와 강도함수(intensity function)를 이용하여 테스트 이후에 남아있는 결함수와 신뢰도를 추정하였다. <표 1>에서와 같이 Goel-Okumoto (1979)의 평균값함수와 순간 고장강도함수는 다음과 같다.

$$m(t)=a[1-\exp(-bt)] \quad 2.3)$$

$$\lambda(t)=ab\exp(-bt). \quad (2.4)$$

여기서, a는 탐지될 기대 결함수로서 $\lim_{t\to\infty}m(t)=\lim_{t\to\infty}E[N(t)]=a$이며, b는 결함 발견율을 나타낸다. 본 논문에서는 시험단계와 운용단계(보증기간)의 평균값 함수 및 순간 고장강도함수를 Goel-Okumoto (1979) 모형으로 설정하여 전개한다.

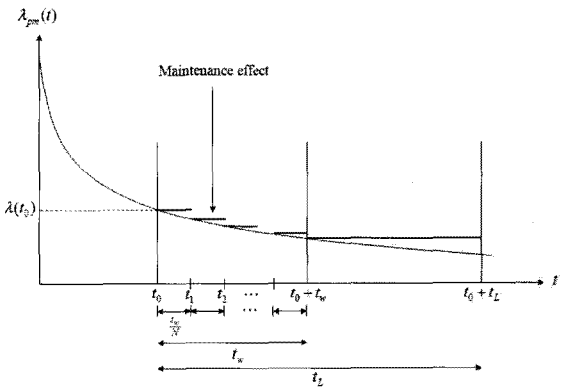
본 논문에서 고려한 시험단계와 운용단계의 환경 차이를 고려한 소프트웨어 신뢰도 성장모형을 전개하기 위해 다음과 같은 가정을 한다.

가 정

1. 소프트웨어 시스템은 시간 $t=0$에서 시험(testing)을 시작한다.
2. 소프트웨어 개발자는 소프트웨어 출시이후 일정한 보증기간 t_w동안 주기적인 보전활동(예, patch, update 등)을 실시한다.
3. 주어진 보증기간 동안 소프트웨어 보전은 $\frac{t_w}{N}(t_w \geq 0)$ 주기 만큼 주기적 보전을 실시하며, N번째 보전에서 보전활동을 중단한다.
4. 소프트웨어 제품의 수명주기 $t_L>0$은 알려져 있다.
5. 소프트웨어 결함이 발견되면 즉시 제거하며, 새로운 결함은 도입되지 않는다.

위와 같은 가정을 토대로 보증기간을 고려한 소프트웨어 신뢰도 성장모형은 <그림 1>과 같이 나타낼 수 있다. 소프트웨어 개발과정을 살펴보면 시험기간 동안 계속되는 시험을 실시함으로써 결함이 발생할 경우 통상적으로 디버깅을 수행한다. 소프트웨어 출시 이후 보전기간 동안은 보전팀 (maintenance team)을 구성하여 시험을 실시함으로써 결함발생 시에 대비한다. 보전기간 이후에는 보전팀을 해체하므로, 결함발생 시에는 긴급하게 팀을 구성하여 운영할 수 있으나 중대결함이 아닌 이상 디버깅을 수행하지 않는다. 즉, 소프트웨어 출시시점 t_0까지는 계속되는 디버깅으로 인해 고장이 감소되는 형태를 보이지만 출시이후에는 주기적인 패치나 업데이트를 통하여 보전활동을 실시함으로써 계단함수의 형태를 갖게 된다. 보증기간 t_w가 끝나면 중대한 결함이 발견되지 않는 이상 추가적인 디버깅을 하지 않으므로 고장강도함수는 일정하게 유지하게 된다.

또한 본 논문에서는 소프트웨어 중 패키지소프트웨어라 불리는 범용소프트웨어에 대한 소프트웨어 신뢰도 성장모형을 고려한다. 범용소프트웨어는 소프트웨어 개발자가 고객의 요구를 파악하여 소프트웨어를 개발한 후 불특정 다수인을 상대로 출시하는 경우를 말하며, 본 논문에서 가정한 바와 같이 주기적인 보전활동을 실시한다. 이러한 가정은 매우 현실적인 가정이라 할 수 있다.



<그림 1> 보증기간을 고려한 소프트웨어 신뢰도 성장모형

2.2 소프트웨어 총기대비용

소프트웨어 시험 및 보증기간 등 수명주기 과정에서 발생할 수 있는 모든 비용을 고려하여 소프트웨어 총기대비용(total expected software cost)을 유도하고, 이를 최소로 하는 최적의 출시시점 및 보증기간을 결정하는 것이 본 논문의 목적이므로 다음과 같이 총기대비용을 산출한다.

1) 시험단계 $(0, t_0)$동안의 결함제거 기대비용

: 시험단계 $(0, t_0)$동안 제거된 기대 누적결함수는 $E[N(t_0)] = m(t_0)$이므로 시간 t_0까지 발견된 모든 결함을 제거하기 위한 기대비용은 다음과 같다.

$$EC_0(t_0, t_w) = c_0 E[N(t_0)] = c_0 m(t_0). \quad (2.5)$$

2) 보증기간 $[t_0, t_0 + t_w)$동안의 결함제거 기대비용

: 보증기간 $[t_0, t_0 + t_w)$동안 제거된 기대 누적결함수는 $E[N(t_0 + t_w) - N(t_0)] = m(t_0 + t_w) - m(t_0)$이므로 기간 $[t_0, t_0 + t_w)$까지 발견된 모든 결함을 제거하기 위한 기대비용은 다음과 같다.

$$\begin{aligned} EC_w(t_0, t_w) &= c_w E[N(t_0 + t_w) - N(t_0)] \\ &= c_w [m(t_0 + t_w) - m(t_0)]. \end{aligned} \quad (2.6)$$

3) 보증기간 이후 $[t_0 + t_w, t_0 + t_L)$동안의 결함제거 기대비용

: 보증기간 이후 $[t_0 + t_w, t_0 + t_L)$동안 제거된 기대 누적결함수는 $E[N(t_0 + t_L) - N(t_0 + t_w)] = m(t_0 + t_L) - m(t_0 + t_w)$이므로 기간 $[t_0 + t_w, t_0 + t_L)$까지 발견된 모든 결함을 제거하기 위한 기대비용은 다음과 같다.

$$\begin{aligned} EC_L(t_0, t_w) &= c_L E[N(t_0 + t_L) - N(t_0 + t_w)] \\ &= c_L [m(t_0 + t_L) - m(t_0 + t_w)]. \end{aligned} \quad (2.7)$$

4) 보증기간 $[t_0, t_0 + t_w)$동안 보전활동 비용

: 보증기간동안 발생하는 비용은 주기적인 보전활동의 횟수에 비례하므로 다음과 같이 설정한다.

$$EC_p(t_0, t_w) = c_p N. \quad (2.8)$$

5) 시험단계와 보증기간동안의 테스트비용

: 테스트비용은 t_0와 t_w에 비례하므로 다음과 같이 설정한다.

$$EC_t(t_0, t_w) = c_{t0} t_0 + c_{tw} t_w. \quad (2.9)$$

따라서 1)~5)를 토대로 총기대비용 $EC(t_0, t_w)$는 다음과 같이 유도한다.

$$\begin{aligned} EC(t_0, t_w) &= c_0 m(t_0) + c_w [m(t_0 + t_w) - m(t_0)] \\ &\quad + c_L [m(t_0 + t_L) - m(t_0 + t_w)] \\ &\quad + c_p N + c_{t0} t_0 + c_{tw} t_w. \end{aligned} \quad (2.10)$$

식 (2.10)에 Goel-Okumoto(1979)의 평균값함수인 식 (2.3)을 대입하면 총기대비용은 다음과 같다.

$$\begin{aligned} EC(t_0, t_w) &= ac_0(1 - e^{-bt_0}) + \frac{t_w}{N} abc_w e^{-bt_0} \sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} \\ &\quad + abc_L e^{-b(t_0 + t_w)}(t_L - t_w) + c_p N + c_{t0} t_0 + c_{tw} t_w. \end{aligned} \quad (2.11)$$

소프트웨어 총기대비용 $EC(t_0, t_w)$을 최소로 하는 최적의 소프트웨어 출시시점과 보증기간은 다음 장에서 유도한다.

3. 최적 출시시점 및 보증기간 결정

본 장에서는 $EC(t_0, t_w)$을 최소로 하는 최적의 출시시점 t_0^*과 보증기간 t_w^*가 존재하는가를 보이기 위하여 다음과 같은 가정을 한다. 현실적으로 시험기간동안 결함제거비용은 보증기간동안의 결함제거비용보다 높으며, 보증기간 이후에는 다시 보전팀을 긴급하게 구성하여 결함을 제거하므로 결함제거비용은 높아지게 마련이다.

$$A-1 \quad c_L > c_w > c_0 \quad (3.1)$$

$$A-2 \quad \frac{t_w}{N} bc_w \sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} > c_0 \quad (3.2)$$

$$A-3 \quad b\left(\frac{t_w}{N} c_w \sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} + c_L e^{-bt_w}(t_L - t_w)\right) > c_0 \quad (3.3)$$

또한

$$\begin{aligned} Q(t_w) = ab\Big\{ & c_0 - \frac{t_w}{N} bc_w \sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} \\ & - bc_L e^{-bt_w}(t_L - t_w)\Big\} + c_{t0} \end{aligned}$$

이라 하자.

정리 1. $c_0, c_w, c_L, c_p, c_{t0}, c_{tw}, N, t_w, t_L$이 주어졌을 때, $EC(t_0, t_w)$를 최소로 하는 최적 출시시점 t_0^*은 A-1~A-3에 따라 다음과 같이 결정된다.

Case 1) $Q(t_w) < 0$일 경우, 총기대비용 $EC(t_0, t_w)$를 최소로 하는 유한하고 유일한 t_0^*가 존재한다.

$$EC(t_0^*, t_w) = ac_0[1-e^{-bt_0^*}] + \frac{t_w}{N}abc_w e^{-bt_0^*}\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} + abc_L e^{-b(t_0^*+t_w)}(t_L - t_w) + c_p N + c_{t0}t_0^* + c_{tw}t_w.$$

Case 2) $Q(t_w) \geq 0$일 경우, $t_0^* = 0$이 된다.

$$EC(t_0^*, t_w) = \frac{t_w}{N}abc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} + abc_L e^{-bt_w}(t_L - t_w) + c_p N + c_{tw}t_w.$$

증명 1.

$$\frac{\partial EC(t_0,t_w)}{\partial t_0} = abe^{-bt_0}[c_0 - \frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} - bc_L e^{-bt_w}(t_L - t_w)] + c_{t0}. \quad (3.2)$$

여기서 식 (3.2)의 우측항을 $\delta(t_0, t_w)$라 하자.

$$\delta(0, t_w) = ab[c_0 - \frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} - bc_L e^{-bt_w}(t_L - t_w)] + c_{t0} \quad (3.3)$$

$$\lim_{t_0\to\infty}\delta(t_0, t_w) = c_{t0} \quad (3.4)$$

이다.

식 (3.2)를 t_0에 대해 편미분하면 다음과 같다.

$$\frac{\partial \delta(t_0,t_w)}{\partial t_0} = ab^2 e^{-bt_0}[c_0 - \frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} - bc_L e^{-bt_w}(t_L - t_w)] + c_{t0}. \quad (3.5)$$

식 (3.5)의 우측항을 $\gamma(t_0, t_w)$라 하면

$$\gamma(0, t_w) = ab^2[c_0 - \frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} - bc_L e^{-bt_w}(t_L - t_w)] + c_{t0} \quad (3.6)$$

이다.

여기서 식 (3.6)의 우측항을 $A(t_w)$라 하면, $A(0) > 0$는 $bc_L t_L > c_0$와 같고, $A(t_L) > 0$는 $\frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} > c_0$와 같다. 그러므로 가정 A-1으로부터 $A(0) < A(t_L)$이 성립한다. 더욱이

$$A'(t_w) = ab^2[\frac{1}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} - \frac{t_w}{N^2}b^2\sum_{k=1}^{N}(k-1)e^{-b(k-1)\frac{t_w}{N}} - b^2c_L e^{-bt_w}(t_L - t_w) - bc_L e^{-bt_w}] < 0, \quad (3.7)$$

$\gamma(0, t_w) > 0$이다.

따라서 $\frac{\partial \gamma(t_0, t_w)}{\partial t_0} < 0$는

$\frac{t_w}{N}bc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} + bc_L e^{-bt_w}(t_L - t_w) > c_0$과 같다.

정리 2. $c_0, c_w, c_L, c_p, c_t, N, t_0, t_L$이 주어졌을 때, $EC(t_0, t_w)$를 최소로 하는 최적 보증기간 t_w^*은 다음과 같이 결정된다.

Case 1) $c_{t0} \geq abe^{-bt_0}[c_L + bc_t t_L - c_w]$일 경우, $t_w^* = 0$이다.

$$EC(t_0, t_w^*) = ac_0[1-e^{-bt_0}] + abc_L e^{-bt_0}t_L + c_p N + c_0 t_0.$$

Case 2) $c_{t0} < abc_L e^{-bt_0}(bt_L + 1)$이고 $c_{t0} > abc_L e^{-b(t_0+t_L)}(bt_L + 1)$일 경우, 총기대비 $EC(t_0, t_w)$를 최소로 하는 유한하고 유일한 t_w^*가 존재한다.

$$EC(t_0, t_w^*) = ac_0(1-e^{-bt_0}) + \frac{t_w^*}{N}abc_w e^{-bt_0}\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w^*}{N}} + abc_L e^{-b(t_0+t_w^*)}(t_L - t_w^*) + c_p N + c_{t0}t_0 + c_{tw}t_w^*.$$

Case 3) $c_t \leq abc_L e^{-b(t_0+t_L)}(bt_L + 1)$일 경우, $t_w^* = t_L$이다.

$$EC(t_0, t_w^*) = ac_0(1-e^{-bt_0}) + \frac{t_w}{N}abc_w\sum_{k=1}^{N} e^{-b(k-1)\frac{t_w}{N}} + c_p N + c_{t0}t_0 + c_{tw}t_L.$$

증명 2. 정리 1과 유사한 방법으로 수행되며, 증명은 생략한다.

이제부터 단위시간당 기대비용 $EC(t_0, t_w)$를 최소화하는 최적의 출시시기와 보증기간을 동시에 고려하는 알고리즘을 소개한다(Rinsaka and Dohi, 2005).

Algorithm

Step 1. 만약 $t_0 \geq 0, 0 \leq t_w \leq t_L$에 대해 식(3.8)을 만족하는 (t_0, t_w)이 존재한다면 Step 2를 실행한다. 그렇지 않으면 Step 3을 실행한다.

$$\frac{\partial EC(t_0,t_w)}{\partial t_0} = \frac{\partial EC(t_0,t_w)}{\partial t_w} = 0 \tag{3.8}$$

Step 2. 헤시안(Hessian) 행렬을 다음과 같이 정의하자.

$$H(t_0,t_w) = -\frac{\partial^2 EC(t_0,t_w)}{\partial^2 t_0}\frac{\partial^2 EC(t_0,t_w)}{\partial^2 t_w} - \left(\frac{\partial^2 EC(t_0,t_w)}{\partial t_0 t_w}\right)^2 \tag{3.9}$$

만약 $H(t_0,t_w) > 0$이고, $\partial^2 EC(t_0,t_w)/\partial^2 t_0 > 0$이라면 Step 1에서 발견된 해가 최적의 보전정책 (t_0^{**}, t_w^{**})이 되고 알고리즘은 종료된다. 그렇지 않으면 Step 3을 실행한다.

Step 3. $\partial EC(0,t_w)/\partial t_w = 0$을 만족하는 $t_w^*(0 \leq t_w \leq t_L)$을 결정하고 다음과 같이 설정한다.

$$EC^{(1)}(t_0^*, t_w^*) = EC(0, t_w^*).$$

Step 4. $\partial EC(t_0,0)/\partial t_0 = 0$을 만족하는 $t_0^*(0 \leq t_0^* < \infty)$을 결정하고 다음과 같이 설정한다.

$$EC^{(2)}(t_0^*, t_w^*) = EC(t_0^*, 0).$$

Step 5. $\partial EC(t_0,t_L)/\partial t_0 = 0$을 만족하는 $t_0^*(0 \leq t_0^* < \infty)$을 결정하고 다음과 같이 설정한다.

$$EC^{(3)}(t_0^*, t_w^*) = EC(t_0^*, t_L^*).$$

Step 6. 소프트웨어 총기대비용의 최소화는 $EC(t_0^{**}, t_w^{**}) = \min_{i=1,2,3} EC^{(i)}(t_0^*, t_w^*)$으로부터 결정되며, 최적의 보전정책은 (t_0^{**}, t_w^{**})이 된다.

4. 수치예제

본 장에서는 시험기간에 발견된 실제 결함자료를 활용하며, 시험단계와 운용단계를 고려한 모형에 적용해 보고자 한다. 실제 결함자료는 Abde-Ghaly et al.(1986)에서 제시한 자료로서 결함당 소프트웨어 고장발생시간($n=86$)을 나타낸 자료이다. 최우추정법을 이용하여 Goel-Okumoto 모형(1979)의 모수를 수치적으로 추정하면 다음과 같다.

$$\hat{a} = 98.5188,\ \hat{b} = 0.0184$$

모수 추정값을 이용하여 평균값함수를 유도하면 다음과 같다.

$$\hat{m}(t) = 98.5188(1 - e^{-0.0184t}),\ t > 0.$$

제안된 비용함수에서 최적의 출시시점 t_0^*과 보증기간 t_w^*를 구하기 위해 Rinsaka and Dohi(2006)에서 제시된 모수를 다음과 같이 가정한다.

$$c_0 = 1.0,\ c_w = 3.0,\ c_L = 20.0,\ c_{tw} = 0.3,$$
$$c_p = 2.0,\ t_L = 1{,}000,\ N = 50$$

<표 2> 최적 출시시점 t_0^* 및 총기대비용 $EC(t_0^*, t_w)$

	$t_w = 200$		$t_w = 210$		$t_w = 220$		$t_w = 230$		$t_w = 250$	
c_{t0}	t_0^*	$EC(t_0^*, t_w)$	t_0^*	$EC(t_0^*, t_w)$	t_0^*	$EC(t_0^*, t_w)$	t_0^*	$EC(t_0^*, t_w)$	t_0^*	$EC(t_0^*, t_w)$
0.20	243.75	488.59	236.47	475.59	228.94	462.11	220.85	447.56	211.69	431.95
0.21	201.47	493.37	195.66	480.72	189.87	468.12	182.16	451.00	175.94	438.39
0.22	162.68	506.48	158.47	494.89	154.34	483.54	148.62	467.43	144.28	456.45
0.23	134.51	537.16	130.90	524.28	126.21	507.25	121.29	489.34	117.94	478.46
0.24	99.85	553.20	95.66	531.82	91.09	508.42	87.14	501.33	83.88	492.92
0.25	68.46	561.05	65.64	539.77	62.12	512.95	58.65	506.52	55.01	495.87

<표 3> 최적 보증기간 t_w^* 및 총기대비용 $EC(t_0, t_w^*)$

	$t_0 = 500$		$t_0 = 510$		$t_0 = 520$		$t_0 = 530$		$t_0 = 550$	
c_{t0}	t_w^*	$EC(t_0, t_w^*)$	t_w^*	$EC(t_0, t_w^*)$	t_w^*	$EC(t_0, t_w^*)$	t_w^*	$EC(t_0, t_w^*)$	t_w^*	$EC(t_0, t_w^*)$
0.20	354.48	716.90	336.76	681.06	316.55	640.20	294.39	595.38	270.84	547.74
0.21	305.74	748.94	291.06	713.00	273.60	670.21	254.45	623.30	234.09	573.43
0.22	251.69	777.82	239.86	741.27	225.48	696.79	209.69	648.02	192.91	596.17
0.23	196.87	776.62	187.81	750.89	176.55	716.44	164.19	667.69	151.05	627.87
0.24	142.06	772.81	135.67	778.03	127.53	743.75	118.60	705.19	109.11	663.57
0.25	88.98	772.13	85.06	799.40	79.96	770.89	74.36	738.15	68.41	696.31

<표 2>는 단위시간당 시험비용 c_t와 보증기간 t_w의 값에 따라 최적 출시시점 t_0^*을 계산한 결과이다. 결과를 살펴보면 고정된 보증기간 하에서 단위시간당 시험비용 c_{t0}가 증가할수록 최적 출시시점 t_0^*은 보다 이른 시점에 배포하게 되고 총기대비용은 증가하게 된다. 또한 고정된 시험비용 하에서 보증기간 t_w가 길어질수록 최적 출시시점 t_0^*은 마찬가지로 빠르게 배포하게 되며 총기대비용도 감소하게 된다. 이러한 결과는 단위시간당 기대비용이 증가하게 되면 총기대비용이 증가하게 되므로 상대적으로 총기대비용을 최소로 하기 위해 빠른 시기에 출시하게 되는 것이다. 또한 보증기간이 길어지게 되면 총기대비용이 증가하게 되고, 보증기간동안 결함을 제거할 수도 있으므로 출시시점이 빨라지게 되는 것이다.

<표 3>은 단위시간당 시험비용 c_t와 출시시점 t_0의 값에 따라 최적 보증기간 t_w^*을 계산한 결과이다. 결과를 살펴보면 고정된 출시시점 하에서 단위시간당 시험비용 c_{t0}가 증가할수록 최적 보증기간 t_w^*은 보다 짧아지게 되고 총기대비용은 증가하게 된다. 또한 고정된 시험비용 하에서 출시시점 t_0가 길어질수록 최적 보증기간 t_w^*은 마찬가지로 짧아지게 되며 총기대비용은 증가하게 된다. 이러한 결과는 단위시간당 기대비용이 증가하게 되면 총기대비용이 증가하게 되므로 총기대비용을 최소로 하기 위해 보증기간을 짧게 하는 것으로 해석될 수 있다.

<표 4> 최적 출시시점 t_0^{**}과 최적 보증기간 t_w^{**} 및 총기대비용 $EC(t_0^{**}, t_w^{**})$

c_{t0}	t_0^{**}	t_w^{**}	$EC(t_0^{**}, t_w^{**})$
0.20	364.38	328.55	328.52
0.21	321.92	297.53	463.66
0.22	284.33	263.62	523.37
0.23	252.83	235.25	573.78
0.24	223.61	210.89	621.33
0.25	209.32	188.79	653.79

참고문헌

[1] 최규식, 김용경(2003), "운영단계의 소프트웨어 신뢰도에 관한 연구", 「Journal of Information Technology Application and Management」, 10권, 4호, pp. 185-194.

[2] Abde-Ghaly, A. A., Chan, P. Y, and Littlewood, B.(1986), "Evaluation of competing software reliability prediction", *IEEE Transactions on Software Engineering*, Vol. 12, pp. 950-967.

[3] Chang, I. P.(1997), "An analysis of software reliability with change-point models", NSC 85-2121-M031-003, *National Science Council*, Taiwan.

[4] Goel, A. L. and Okumoto, K.(1979), "Time-dependent error-detection rate model for software reliability and other performance measures", *IEEE Transactions on Reliability*, Vol. R-28, No. 3, pp.

206-211.

[5] Musa, J. D., Iannino, A., Okumoto, K.(1987), *Softw are reliability measurement prediction application,* McGraw-Hill, New York

[6] Pham, H. and Zhang, X.(2003), "A software cost model with imperfect debugging, random life cycle and penalty cost", *International Journal of System Science*, Vol. 27, pp. 455-463.

[7] Rinsaka, K., and Dohi, T.(2005), "Determining the optimal software warranty period under various operational circumstances", *International Journal of Quality and Reliability Management*, Vol. 22, pp. 715-730.

[8] Rinsaka, K., and Dohi, T.(2006), "Optimal Testing/ Maintenance Design in a Software Development Project", *Electronics and Communications in Japan*, Vol. 89, pp. 953-961.

[9] Shyur, H. J.(2003), "A stochastic software reliability model with imperfect debugging and change-point", *The Journal of System and Software*, Vol. 66, pp. 135-141.

[10] Yamada, S.(1993), "Software reliability measurement during operational phase its application", *Jou rnal of Computational Software Engineering*, pp. 389-402.

[11] Yamada, S., Ohba, M. and Osaki, S.(1983), "S-shaped reliability growth modeling for software error detection", *IEEE Transactions on Reliability*, Vol. R-32, No. 5, pp. 475-478, 484.

[12] Yamada, S., Tanio, Y. and Osaki, S.(1989), "A software reliability evaluation method during operation phase", *Transactions on IEICE*, J72-D-I, pp. 797 -803.

[13] Yamada, S., Tanio, Y. and Osaki, S.(1991), "Software reliability measurement and assessment methods during operational phase and their comparisons", *Transactions on IEICE*, J7-D-I, pp. 240-248.

[14] Yamada, S., Tokuno, K, and Osaki, S.(1992), "Imperfect debugging models with fault introduction rate for software reliability assessment", *International Journal of System Science*, Vol. 23, pp. 2241-2252.

2010년 6월 4일 접수, 2011년 5월 13일 수정, 2011년 5월 31일 채택