

# 고장 데이터의 플롯을 이용한 소프트웨어 신뢰도 성장 모델의 성능평가

정혜정<sup>†</sup>·양해솔<sup>††</sup>·박인수<sup>†††</sup>

## 요약

소프트웨어 신뢰도 성장 모델은 다양하게 연구되어져 있다. 그러나 이러한 모델에서 정확한 모수를 추정하는 것은 그리 쉽지 않다. 특히 고장 데이터에 대하여 소프트웨어 신뢰도 성장 모델의 추정이 정확히 이루어져야만 모델을 설명하는 모수의 추정도 정확하게 이루어질 수 있다. 이러한 측면에서 테스트를 통해서 얻어진 소프트웨어의 고장 데이터의 정규확률점수를 구해서 두 개의 값에 대한 플롯을 그려보고 그려진 결과를 이용해서 분포를 예측하여 예측된 분포에 적합한 소프트웨어 신뢰도 성장 모델을 적용한다면 상당히 정확한 테스트 결과를 얻을 수 있을 것이다. 본 논문에서는 고장 데이터의 플롯을 통한 결과를 통해서 분포를 예측하고 모델을 성능평가 척도에 따라서 모의실험을 하여 그 결과를 통해서 소프트웨어 신뢰도 성장 모델의 적합성을 검증하는 연구이다. 연구결과 고장데이터의 정규점수를 이용한 플롯을 보고 소프트웨어 신뢰도 성장 모델을 예측할 수 있었고 이러한 예측을 통해서 모델을 선정한다면 모델의 성능평가에서도 우수함을 확인할 수 있다.

## Performance estimation for Software Reliability Growth Model that Use Plot of Failure Data

Hye-Jung Jung<sup>†</sup> · Hae-Sool Yang<sup>††</sup> · In-Soo Park<sup>†††</sup>

### ABSTRACT

Software Reliability Growth Model that have been studied variously. But measurement of correct parameter of this model is not easy. Specially, estimation of correct model about failure data must be establish and estimation of parameter can consist exactly. To get correct testing, we calculate the normal score and describe the normal probability plot. Use the normal probability plot, we estimate the distribution for failure data. In this paper, we estimate the software reliability growth model for through the normal probability plot. In this research, we applies software reliability growth model through distribution characteristics of failure data. If we see plot, we determine the software reliability growth model, we can make sure superior in model's performance estimation.

키워드 : 소프트웨어 신뢰성 성장 모델(Software Reliability Growth Model), 신뢰도 함수(Reliability Function), 정확도(Accuracy), 노이즈(Noise), 고장데이터 플롯(Plot of Failure Data)

### 1. 서론

소프트웨어 신뢰도 성장 모델에 대한 연구는 고장 데이터의 특징과 분포의 특성에 따라서 진행되고 있다. 테스트를 통해서 고장 데이터가 얻어지면 이러한 데이터를 소프트웨어 신뢰도 성장 모델에 적용하여 주어진 모델에 대한 모수를 추정하고 추정된 모수를 이용하여 신뢰도 함수나 평균고장시간 등의 예측값을 측정하게 된다. 이러한 결과는 또한 소프트웨어 양도시간의 결정에도 적용된다. 소프트웨어는 개발 노력의 절반이상을 소프트웨어의 테스트에 기울고 있으며 일반적으로 소프트웨어의 개발에 있어서 일정 에 대한 차질이나 비용초과에 대한 주요한 이유도 역시 테

스팅 할 충분한 시간과 적절한 자원을 할당하지 못한데서 비롯된 결과로 보인다. 소비자의 욕구를 100% 완벽하게 충족할 수 있는 소프트웨어를 개발할 수는 없지만 최대의 만족을 줄 수 있는 소프트웨어의 개발을 위해서 개발된 소프트웨어를 양도하기까지 정확한 테스트를 거쳐서 우수한 소프트웨어를 개발하는 것은 소프트웨어 개발자의 입장에서 상당히 중요한 문제이다. 소프트웨어 테스트를 통해서 소프트웨어의 신뢰도 성장 모델에 대한 연구는 상당히 많이 진행되어 오고 있다. 그러나 실질적으로 개발된 소프트웨어의 실험 데이터를 분석해서 소프트웨어 신뢰도 성장 모델에 적용하는 방법에 대한 연구는 진행되지 않은 실정이다. 또한 테스트 데이터를 소프트웨어 신뢰도 성장 모델에 적용하여 기본적인 신뢰도 함수값을 찾게 되면 이러한 데이터가 소프트웨어 신뢰도 함수를 얼마나 정확히 측정하고 있는가에 대한 평가 연구는 다소 진행된 것이 있으나 이것도

† 종신회원 : 평택대학교 정보통계학과 교수  
 †† 종신회원 : 호서대학교 벤처전문대학원 교수  
 ††† 정 회 원 : 산업자원부 기술표준원 공업연구관  
 논문접수 : 2003년 5월 29일, 심사완료 : 2003년 7월 14일

연구가 미비한 상황이다. 이 분야에 대한 연구는 대체적으로 소프트웨어 신뢰도 성장 모델을 먼저 제안하고 제안된 모델에다가 실험 데이터를 적용시켜 모수를 추정하고 추정된 모수를 이용하여 신뢰도 함수값을 구하는 방법으로 진행되었다. 본 논문에서는 이러한 테스트의 가정을 통해서 연구되어야 하는 소프트웨어 신뢰도 성장 모델을 기존의 연구방법과는 다르게 실질적으로 주어진 실험 데이터를 정규점수를 구해서 정규확률 플롯을 그려 보고 정규확률 플롯의 형태에 따라서 소프트웨어 신뢰도 성장 모델을 찾아오는 방법으로 연구를 진행 한다. 본 논문에서는 먼저 2장에서 소프트웨어 신뢰도 성장 모델을 특징별로 소개하고 모델에 대한 기본적인 모수 추정방법을 소개한다.

3장에서는 모델에 대한 성능 평가 방법으로 정확도와 노이즈에 대하여 소개하고 4장에서는 Musa의 3가지 소프트웨어 고장 데이터를 이용하여 정규점수를 구해서 정규확률 플롯을 그려보고 3장에서 소개한 정확도와 노이즈를 Musa의 데이터를 이용하여 시뮬레이션 해 본 결과를 제시하고 모델별로 평가한 결과를 기술하고 5장에서는 이러한 결과를 통해서 나온 결과를 분석하고 앞으로의 연구과제를 제시한다.

**2. 소프트웨어 신뢰도 성장 모델**

1972년에 Jelinski-Moranda[13]의 연구를 시초로 하여 소프트웨어 신뢰도 성장 모델에 대한 연구가 활발히 진행되었다. 일반적으로 소프트웨어의 신뢰도를 측정하기 위해서는 소프트웨어의 테스트 과정에서 얻어진 데이터를 소프트웨어 신뢰도 성장 모델에 접근시켜서 모수를 추정한다. 모수를 추정하는 방법도 다양하게 연구되었다. 기존에 연구된 방법으로는 최우추정법을 가장 많이 이용하고 있으며, 그 외에도 베이즈 추정법이나 깁스샘플링 방법 등을 이용하고 있다. 이와 같이 모수가 추정되고 나면 추정된 모수를 이용하여 신뢰도에 관련된 여러 가지 함수를 구하게 된다. 그러나 이러한 신뢰도에 대한 측정은 모수를 추정하는 방법 부터 상당히 수리적인 어려움이 있어 일반 소프트웨어 개발업체에서 테스트 과정을 통하여 개발된 소프트웨어에 대한 신뢰도를 평가하는 것은 거의 이루어지고 있지 않는 실정이다.

소프트웨어 신뢰도 성장 모델의 시초인 Jelinski & Moranda(JM)[13]에 의해서 제안된 소프트웨어 신뢰도 성장 모델은 지수분포를 따르는 함수로서 다음과 같다

$$f(t_i) = (N - i + 1)\phi \exp(-(N - i + 1)\phi t_i) \quad (2.1)$$

식 (2.1)에서 N은 소프트웨어의 초기 단계에 내장 되어 있는 총 오류의 수가 되며  $\phi$ 는 실패강도를 나타낸다.

이 모델에서의 기본 가정은 아래와 같다.

- ① 초기 소프트웨어 실패의 수는 알려 지지 않은 상수이다.
- ② 발견된 오류는 즉시 제거되어지며 새로운 오류를 만

들지 않는다.

- ③ 실패사이의 시간은 독립이고 지수분포를 따르는 확률 변수이다.
- ④ 남아있는 소프트웨어 실패율은 항상 같은 상수이다.

위에 제안된 JM 모델을 중심으로 다양한 연구가 진행되었으며, 이러한 연구는 상당히 많은 소프트웨어 신뢰도 연구자들에게 기본적인 연구의 방향을 제시하였다.

식 (2.1)에서 소개한 JM 모델은 최우추정법을 이용하여 모수를 추정하였다.

먼저 소프트웨어 신뢰도 성장 모델의 우도함수를 구해서 각각 구하고자 하는 모수에 대하여 미분하여 구하는 방식으로 모수를 추정하는 방법이다. JM 모델에 대한 우도함수는 식 (2.2)와 같다.

$$L(t_1, \dots, t_n : N_0, \phi) = \phi^n \{ \prod_{i=1}^n (N_0 - i + 1) \} \times \exp \left\{ -\phi \sum_{i=1}^n (N_0 - i + 1) t_i \right\}$$

$$\frac{\ln L}{N_0} = \sum_{i=1}^n \frac{1}{N_0 - i + 1} - \sum_{i=1}^n \phi t_i = 0$$

$$\frac{\ln L}{\phi} = \frac{n}{\phi} - \sum_{i=1}^n (N_0 - i + 1) t_i = 0 \quad (2.2)$$

위의 방정식을 N과  $\phi$ 에 대하여 각각 편미분해서 정리하면 식 (2.3)과 같은 결과를 얻을 수 있다.

$$\phi = n \left\{ \sum_{i=1}^n (N_0 - i + 1) t_i \right\}^{-1}$$

$$\frac{1}{N_0} + \dots + \frac{1}{N_0 - n + 1} = \frac{n \sum_{i=1}^n t_i}{\sum_{i=1}^n (N_0 - i + 1) t_i} \quad (2.3)$$

JM 모델에서 위와 같은 최우추정법으로 모수를 추정하게 되면 초기해의 결정에 따라서 구하고자 하는 모수가 다소의 영향을 받는다. 특히 JM 모델은 모수 N은 유한의 양수이어야 하고  $\phi$ 도 또한 양수이어야 한다는 조건에 위배되는 해가 구해지는 경우도 생길수 있다는 것이다. 이러한 문제를 해결하기 위해서 초기해를 줄 때 초기해의 범위를 정의한 연구가 진행되어 아래와 같은 결과를 얻었다.

$$\frac{\sum_{i=1}^n (i-1) t_i}{\sum_{i=1}^n t_i} > \frac{n-1}{2}$$

위의 조건을 만족하는 범위내에서 초기해를 주게 되면 N,  $\phi$ 의 조건을 만족하는 해를 구할 수 있다. 위의 JM 모델에서 주어진 결과를 이용하여 신뢰도 함수를 구하여 보면

$$R(t_i) = e^{-(N - i + 1)\phi}$$

이다. 식 (2.3)의 모수 추정 결과를 이용하면 신뢰도 함수값을 구할 수 있다. 또한 소프트웨어 신뢰도 성장 모델을 통해서 얻어질 수 있는 신뢰도의 다양한 정량적인 값들을 구할 수 있다. 그러나 Littlewood-Verrall(LV)[5]은 JM 모델에서 지적인 모수추정에 있어서 초기치의 값에 영향을 받는 문제점의 해결방안으로 베이지안 방법으로 모수를 추정하는 연구를 하여 발표하였다. 소프트웨어 신뢰도 성장 모델에서 모수를 추정하기 위하여 추정을 원하는 모수에 사전분포를 주고 사후분포를 계산하여 계산된 사후분포에서 베이지추정값을 구하는 방법에 대한 연구를 진행시켰다. Littlewood-Verrall[5]이 제안한 소프트웨어 신뢰도 성장 모델은 고장 발생시간에 대한 함수로 지수함수를 제안하였다.

$$f(t_i) = \lambda_i \exp^{-\lambda_i t_i} \tag{2.4}$$

식 (2.4)에서  $\lambda_i$ 에 대한 사전분포를 주고 사후분포를 구하여 모수를 추정하는 방법으로 연구를 진행시켰다.

위의 모델에서 실패율에 해당되는 모수  $\lambda_i$ 의 특성에 따라서 사전분포는 감마분포를 제안하였다.

$$f(\lambda_i | \alpha, \psi(i)) = \frac{\psi(i)^\alpha \lambda_i^{\alpha-1} \exp(-\psi(i)\lambda_i)}{\Gamma(\alpha)} \tag{2.5}$$

여기에서

$\lambda_i = (N-i+1)\phi$ 이다. 그리고 N은 소프트웨어 개발 초기에 담고 있는 총 오류의 수이고  $\phi$ 는 실패강도이다. 위의 제안된 모델에 대한 사전분포를 이용하여 실패시간에 대한 함수를 구하면

$$f(t_i : \alpha, \psi(i)) = \frac{\psi(i)^\alpha}{\Gamma(\alpha)} \times \frac{\Gamma(\alpha+1)}{t_i + \psi(i)} \tag{2.6}$$

과 같다.

Littlewood-Verrall이 제안한 소프트웨어 신뢰도 성장 모델에서 모수추정 방법은 최우추정법을 이용하였다. 즉, 고장발견시간의 함수에 대하여 우도함수를 구하면 식 (2.7)과 같다.

$$f(t_1, \dots, t_n | \alpha, \psi(i)) = \frac{\alpha^n \prod_{i=1}^n \psi(i)^\alpha}{(t_i + \psi(i))^{\alpha+1}} \tag{2.7}$$

위의 모델에서  $\psi(i) = \beta_0 + \beta_1 i$ 라고 가정하고 미지의 모수를 추정하면 식 (2.8)과 같이 정리된다.

$$\sum_{i=1}^n \left( \frac{\alpha}{\beta_0 + \beta_1 i} - \frac{\alpha+1}{t_i + \beta_0 + \beta_1 i} \right) = 0$$

$$\sum_{i=1}^n \left( \frac{\alpha i}{\beta_0 + \beta_1 i} - \frac{(\alpha+1)i}{t_i + \beta_0 + \beta_1 i} \right) = 0 \tag{2.8}$$

식 (2.8)의 계산결과에서 얻어진 모수를 이용하여 LV 모델에 대하여 신뢰도 함수를 구하여 보면 다음과 같다

$$R_i(t_i) = \left( \frac{\psi(i)}{t_i + \psi(i)} \right)^\alpha$$

다음은 Keller-Littlewood(KL)가 제안한 모델로써 앞에서 제안한 LV 모델에서 실패율에 대한 사전분포를 모수의 위치를 바꾸어서 제안한 모델이다. 즉 고장시간에 대한 분포는 LV가 제안한 식 (2.4)와 같이 지수함수를 따르는 분포이다.

그리고 위의 모델에서 고장을  $\lambda_i$ 에 대한 사전분포는 식 (2.9)와 같다

$$f(\lambda_i | \alpha, \psi(i)) = \frac{\beta^\psi(i) \lambda_i^{\psi(i)-1} \exp(-\beta \lambda_i)}{\Gamma(\psi(i))} \tag{2.9}$$

여기에서

$\psi(i) = (\alpha_0 + \alpha_1 i)^{-1}$ 이고 i번째 오류가 발견된 뒤에 신뢰도 함수는 아래와 같이 구해진다.

$$R_i(t_i) = \left( \frac{\beta}{t_i + \beta} \right)^{\psi(i)}$$

KL 모델도 모수추정 방법은 LV 모델의 모수추정 방법과 같이 우도함수를 이용한 최우추정법으로 추정하였다.

다음은 Langberg-Singpuralla(LS)[9]에 의해서 제안된 소프트웨어 신뢰도 성장 모델이다. Langberg-Singpuralla가 제안한 소프트웨어 신뢰도 성장 모델은 Jelinski-Moranda [11]에 의해서 제안된 소프트웨어 신뢰도 성장 모델에서 모수추정 방법을 베이지안적 입장에서 모수를 추정하는 형태로 연구를 진행하였다.

Langberg-Singpuralla[9]에 의해서 제안된 소프트웨어 신뢰도 성장 모델은 다음과 같다.

$$f(t_i) = (N-i+1)\phi \exp(-(N-i+1)\phi t_i) \tag{2.10}$$

이때 주어진 모수는 전체 오류의 수를 나타내는 N과 실패강도를 나타내는  $\phi$ 이다. 이때 소프트웨어 개발 초기단계에 오류의 수를 나타내는 N은 모수의 특성에 따라서 사전분포로 포아송분포를 주고 고장을 발생하는 정도를 나타내는 실패강도  $\phi$ 에는 모수의 특성에 따라서 감마사전분포를 따른다고 제안하였다.

제안한 사전분포는 아래와 같다.

$$f(t) = \frac{\exp(-\lambda)\lambda^t}{t!}$$

$$f(\phi = \phi) = \frac{\beta^\alpha \phi^{\alpha-1} \exp(-\beta\phi)}{\Gamma(\alpha)}$$

위의 사전분포를 이용하여 LV 모델에서 제안한 방법으로 모수를 추정하고 추정된 모수를 이용하여 신뢰도 함수를 구해보면 아래와 같다.

$$R_i(t_i) = \left( \frac{\beta}{(N-i+1)t_i + \beta^\alpha} \right) \times (N-i+1)$$

위에서 제시한 네 개의 모델은 모두 JM 모델의 지수함수를 따르는 특성을 이용한 모델이다. 다소 성질이 비슷한 소프트웨어 신뢰도 성장 모델로써 모수추정 방법에 있어서 차이를 보이고 있다. 첫 번째 소개한 JM 모델은 마코브 성질을 따르는 소프트웨어 신뢰도 성장 모델로써 추우추정법을 이용하여 모수를 추정하였다. 두 번째 소개한 LV 모델은 모수추정에 있어서 베이지안 방법을 이용하였으며 KL 모델은 LV 모델에서 모수에 주어지는 사전분포의 형태를 다양하게 변형하여 제안하였다. 마지막에 소개된 LS 모델은 실패시간 사이에 종속적인 관계를 가지는 소프트웨어 신뢰도 성장 모델을 연구하기 위해서 제안된 모델로써 모수추정은 베이지안 방법을 이용하였다.

네 개의 제시된 모델은 실패시간 사이에 모두 지수함수의 성질을 가지는 모델이며 JM 모델을 제외하고 베이지안 입장에서 사전분포를 이용하여 모수를 추정하는 방식으로 모수를 예측하였다.

**3. 소프트웨어 신뢰도 성장 모델의 성능평가 방법**

소프트웨어 신뢰도 성장 모델의 모수추정에 있어서 정확도를 평가할 수 있는 척도에 대한 연구는 많지 않다. 대체적으로 모수에 대한 정확도 추정은 MSE(mean square error)나 BIAS를 이용해서 평가하고 있는 실정이다. 그리고 소프트웨어 성능에 대한 비교 평가는 Davies[10]에 의해서 정확도와 노이즈를 이용한 연구 결과가 있다.

본 논문에서는 Musa의 데이터를 이용하여 먼저 정규점수를 구하고 구해진 정규점수를 이용하여 정규화를 플롯을 그려 본 뒤 주어진 모델의 성능을 평가하려 한다. 위에서 제시한 네가지 모델중에서 모수추정을 베이지안 방법으로 추정한 LV 모델, KL 모델, LS 모델에 대하여 정확도와 노이즈를 이용한 평가를 시도하고 시뮬레이션 결과를 비교하여 평가하려 한다.

먼저 정확도의 평가 방법은 Prequential Likelihood(PL) 함수로 예측시스템의 타당성을 측정하는 방법으로 비교를 위한 3개의 제시된 소프트웨어 신뢰도 성장 모델에 대하여 정확도라는 평가척도를 계산하여 보았다. 대체적으로 연속적인 소프트웨어 고장시간들 사이에 다음에 발생할 소프트웨어의 고장시간의 예측은 상당히 중요하다. 정확도라는 소프트웨어 성능평가의 척도는 관측된 고장시간  $t_1, \dots, t_{i-1}$ 을 이용하여  $i$ 번째 고장이 발생할 확률을 이용하여 소프트웨어가 고장 데이터를 설명하는 정확도를 측정하는 평가 방법이다.

일반적으로 주어지는 소프트웨어 신뢰도 성장 모델에 대한 분포함수와 신뢰도 함수는 아래의 방법으로 계산되어진다.

$$F_i(t) = P(T_i < t)$$

$$R_i(t) = 1 - F_i(t) \tag{3.1}$$

식 (3.1)을 이용하여 주어진 소프트웨어 신뢰도 성장 모델

에 대하여 분포함수를 구하게 되고 이러한 분포함수의 미분을 통해서 확률밀도함수를 구할 수 있다. 계산된 확률밀도함수를 이용하여 소프트웨어 신뢰도 성장 모델의 성능에 대한 평가로 정확도를 계산한다. 정확도의 계산방법은 아래와 같다.

$$LOG(PL_n) = LOG(\prod_{i=j+1}^{i+n} f_i(t_i)) \tag{3.2}$$

위의 PL 함수에 대입하여 정확도를 평가하면 확률밀도함수가 0과 1사이의 값으로 이런 값을 곱해서 얻어진 정확도의 값은 상당히 작은 값으로 얻어질 것이다. 그러므로 모델 별로 수치의 변화를 좀 더 정확히 비교하기 위해서 측정값에 자연대수를 취하여 정확도를 비교하여 보았다. 자연대수를 취하여 보면 원값이 0과 1사이의 값이므로 모두 음수의 값으로 결과가 구해진다. 구하여진 값 중에서 PL 값이 크면 클수록 예측치가 정확하다고 평가할 수 있다.

다음은 모델의 성능 평가를 위한 두 번째 척도로 노이즈를 구하기 위한 방법이다. 오류가 발생되어지는 평균시간에 의해서 구해질 수 있는 편기는 식 (3.3)의 계산 수식에 의해서 결정되어진다.

$$\frac{\sum_{i=1}^n (t_i - E(T_i))^2}{\sum_{i=1}^n (t_i - t)^2} \times \frac{n-1}{n-2} \tag{3.3}$$

$t_i$ :  $i$  번째 오류가 발견된 시간  
 $t$ : 임의의 시간

식 (3.3)의 노이즈를 이용하여 소프트웨어 신뢰도 성장 모델의 성능을 평가할 수 있다. 모델의 예측 동작에서 노이즈가 작은값은 노이즈가 큰 값보다 적은 노이즈를 가진다고 분석할 수 있으며 성능면에서 우수하다고 평가할 수 있다. 만약 주어진 자료가  $t_1, \dots, t_{i-1}$ 라고 한다면 다음에 발생할 고장시간  $t_i$ 를 예측할 수 있으며 이러한 예측 값을 통해서 예측값에 대한 기대치를 계산할 수 있다. 두 개의 값을 계산하고 나면 계산된 결과를 이용하여  $i$ 번째 값에 대한 노이즈를 계산할 수 있다.

식 (3.2)와 식 (3.3)에서 각 모델에 대한 성능평가의 척도인 정확도와 노이즈를 측정하는 방법에 대하여 소개하였다. 이러한 방법을 이용하여 앞에서 제시한 3개의 모델 LV 모델, KS 모델, LS 모델에 대한 정확도와 노이즈를 계산하면 아래와 같다.

LV 모델

$$LOG(PL_n) = LOG \left( \prod_{i=j+1}^{i+n} a \frac{(\beta_0 + \beta_1)^a}{(t_i \beta_0 + \beta_1 i)^{a+1}} \right)$$

KL 모델

$$LOG(PL_n) = LOG \left( \prod_{i=j+1}^{i+n} \frac{1}{a_0 + a_1 i} \times \frac{\beta^{\frac{1}{a_0 + a_1 i}}}{(t_i + \beta)^{\frac{a_0 - a_1 i + 1}{a_1 + a_1 i}}} \right)$$

LS 모델

$$LOG(PL_n) = LOG\left(\prod_{i=j+1}^{j+n} \frac{(N-i+1)\beta^\alpha}{((N-i+1)t_i + \beta)^{\alpha+1}}\right) \quad (3.4)$$

식 (3.4)를 이용하여 정확도를 구하면 모델별로 고장 데이터에 대하여 가장 성능이 우수한 모델을 선별할 수 있다.

LV 모델

$$\frac{\sum_{i=1}^n \left(t_i - \left(-\frac{\beta_0 + \beta_1 i}{1 - \alpha}\right)\right)^2}{\sum_{i=1}^n (t_i - t)^2} \times \frac{n-1}{n-2}$$

KL 모델

$$\frac{\sum_{i=1}^n \left(t_i - \left(-\beta \frac{\alpha_0 + \alpha_1 i}{\alpha_0 + \alpha_1 i - 11 - \alpha}\right)\right)^2}{\sum_{i=1}^n (t_i - t)^2} \times \frac{n-1}{n-2}$$

LS 모델

$$\frac{\sum_{i=1}^n \left(t_i - \frac{1}{\alpha - 1}\right)}{\sum_{i=1}^n (t_i - t)^2} \times \frac{n-1}{n-2} \quad (3.5)$$

식 (3.5)의 계산 식에 의해서 산출된 결과를 이용하여 모델에 대한 노이즈를 평가 할 것이다. 식 (3.4)와 식 (3.5)의 계산 결과를 이용하여 고장 데이터에 대하여 정확도와 노이즈를 구하게 되면 소프트웨어 신뢰도 성장 모델의 성능평가가 이루어진다. 제시한 소프트웨어 신뢰도 성장 모델 3개에 대하여 계산된 결과를 이용하여 정확도와 노이즈를 구해 보면 모델에 대한 성능평가도 비교 분석 할 수 있다. 본 논문의 4장에서는 Musa의 데이터를 이용해서 각 데이터에 대한 정규확률점수를 구하고 그 결과를 이용하여 정규확률 플롯을 나타낸다. 또한 정규확률 플롯의 결과를 이용하여 모델에 대한 평가를 1차적으로 실시하고 또한 성능평가의 척도인 정확도와 노이즈를 이용하여 모델별 성능평가에 대한 비교분석을 실시한다.

#### 4. 고장 데이터에 대한 플롯과 모델 성능평가

##### 4.1 고장 데이터에 대한 플롯

본 논문에 이용한 데이터는 Littlewood[6]가 사용한 Rome Air Development Center에서 얻어진 Musa의 소프트웨어 신뢰도 데이터의 일부이며 아래에 주어진 값은 소프트웨어 테스트 과정에서 얻어진 고장 발생시간이다.

데이터에 따른 고장 시간의 데이터는 <표 1>, <표 2>, <표 3>과 같다.

<표 1> Musa의 제 1 소프트웨어 테스트 데이터

3, 30, 113, 81, 115, 9, 2, 112, 15, 138, 50, 77, 24, 108, 88, 670, 120, 26, 114, 325, 55, 242, 68

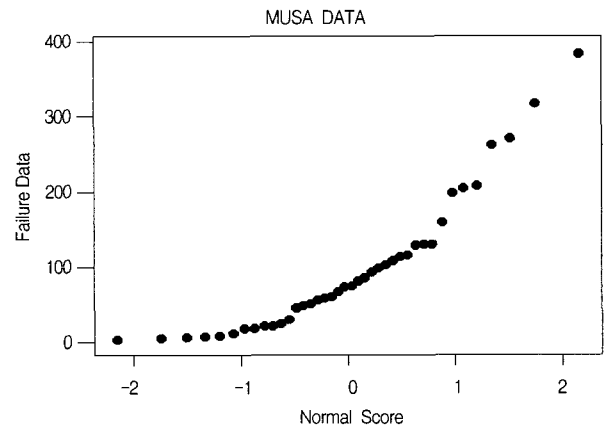
<표 2> Musa의 제 2 소프트웨어 테스트 데이터

479, 266, 277, 554, 1034, 949, 694, 579, 117, 170, 117, 1274, 469, 1174, 693, 1908, 135, 277, 596, 437, 2230, 437, 340, 405

<표 3> Musa의 제 3 소프트웨어 테스트 데이터

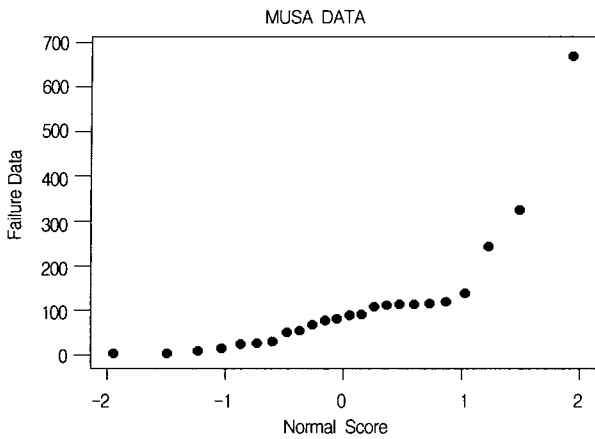
39, 10, 4, 36, 4, 5, 4, 91, 49, 1, 25, 1, 4, 30, 42, 9, 49, 44, 32, 3, 78, 1, 30, 205

위의 세 가지 데이터에 대하여 먼저 정규확률 플롯을 그려서 각 데이터의 분포적 특성을 파악하려 한다. 분석 방법은 통계패키지 Minitab을 이용하여 주어진 Musa의 세 가지 고장 데이터에 대하여 정규확률 점수를 구하고 실제 데이터와 정규확률점수 사이의 관계를 플롯한 정규확률 플롯을 나타낸다. (그림 1), (그림 2), (그림 3), (그림 4)는 먼저 제시한 소프트웨어 신뢰도 성장 모델에서 사용한 지수함수의 난수를 발생시켜서 지수함수의 정규확률 플롯의 형태를 파악하고 Musa의 데이터 3개에 대하여 정규확률 점수를 이용한 정규확률 플롯을 나타낸 결과이다.



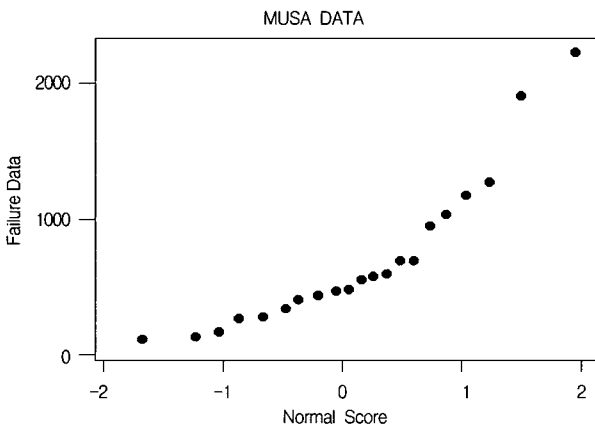
(그림 1) 지수분포를 따르는 데이터의 정규확률 플롯

(그림 1)의 정규확률 플롯은 지수분포 중에서 Musa1의 고장 데이터의 평균시간 111.5의 값을 가지는 난수 40개를 발생시켜서 발생된 난수의 정규확률점수를 구하고 구해진 결과를 그래프로 나타낸 것이다. (그림 1)의 그래프를 통해서 지수함수를 따르는 고장 데이터에 대한 기본적인 정규확률 플롯의 형태를 파악할 수 있다. Musa의 세 가지 데이터를 정규확률 플롯을 그려보고 나타낸 결과를 확인하여 먼저 고장 데이터가 따르는 분포를 파악할 수 있다. MUSA의 고장 데이터를 이용한 정규확률 플롯은 (그림 2), (그림 3), (그림 4)에 제시되어 있다.



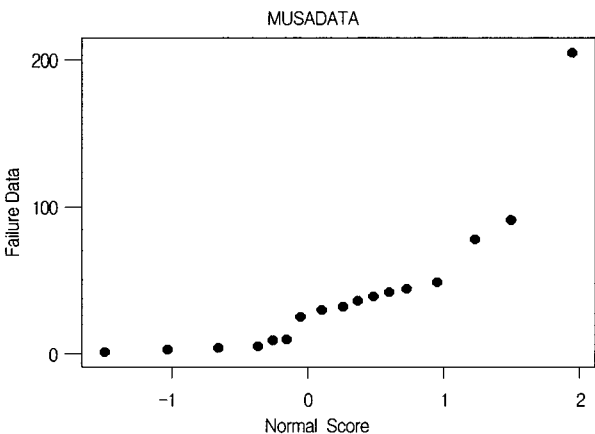
(그림 2) Musa 1 데이터의 정규확률 플롯

(그림 2)는 Musa의 고장 데이터 2를 이용한 정규확률 플롯으로 (그림 1)에서 제시한 지수함수의 형태를 따르고 있음을 알 수 있다.



(그림 3) Musa 2 데이터의 정규확률 플롯

(그림 3)은 Musa 2의 데이터를 이용한 정규확률 플롯으로 (그림 1)의 지수함수 형태를 따르는 형태로 나타나 Musa 2의 데이터도 지수함수의 성질을 가지는 것을 확인할 수 있다.



(그림 4) Musa 3 데이터의 정규확률 플롯

Musa의 데이터를 이용하여 정규확률 플롯을 그린 결과를 살펴보면 (그림 1)에서 지수분포를 따르는 난수를 발생시켜서 정규확률 플롯을 그린 그림의 형태와 모두 유사하다는 것을 알 수 있다. 즉 고장 데이터에 대한 데이터는 모두 지수함수의 성질을 가지는 분포의 특성을 가지고 있다는 것을 쉽게 파악할 수 있다. 또한 Musa의 세 개 데이터 중에서 세 번째 데이터가 가장 난수를 발생시킨 정규확률 플롯과 유사하다는 것을 파악할 수 있으며 다음은 Musa 1 고장 데이터가 유사하다는 것을 알 수 있다.

즉, (그림 1)를 통해서 증명될 수 있는 이론적인 내용은 Musa 3의 고장 데이터가 가장 지수함수의 성질을 따르고 있음을 확인할 수 있으며, 다음은 Musa 1의 데이터 Musa 2의 데이터 순으로 평가할 수 있다. 데이터의 종류별 평가는 테스트를 통해서 얻어진 데이터에 대하여 정규확률 플롯을 그려보면 쉽게 확인할 수 있다. 다음은 앞에서 제시한 3개의 소프트웨어 신뢰도 성장 모델의 성능평가에 대한 예측을 하면 3개의 모델 중에서 지수함수를 고장 데이터의 특성에 따라서 변형없이 그대로 설명하는 소프트웨어 신뢰성 성장 모델은 LV 모델이므로 위에서 제시한 정확도와 노이즈 측면에서 제일 우수한 성능을 가지는 모델로 예측되어 진다.

4.2 모델의 성능평가 결과

3장에서 제시한 세 개의 소프트웨어 신뢰도 성장 모델에 대하여 성능을 평가하기 위한 척도로 정확도와 노이즈를 식 (3.4)와 식 (3.5)에 모델별로 계산하였다. 계산된 결과를 이용하여 각 모델에 대한 성능평가를 한 시뮬레이션 결과는 <표 4>와 같다

<표 4> 모델별 성능평가 결과

구분	Musa 1 데이터		Musa 2 데이터		Musa 3 데이터	
모델	정확도	노이즈	정확도	노이즈	정확도	노이즈
LV	-113	1.09	-158	1.67	-78	0.26
KL	-127	1.87	-170	2.22	-88	0.60
LS	-289	1.70	-445	2.00	-180	0.49

위의 세 가지 Musa의 고장 데이터를 이용하여 플롯을 그려본 결과 Musa 3의 고장 데이터를 이용한 경우의 정규확률플롯이 지수함수의 성질을 가장 잘 설명하고 있는 것으로 보여진다. 그리고 다른 데이터에 대하여 정규확률 플롯의 결과를 분석하여 보면 Musa 1, Musa 2의 순으로 지수함수의 형태에 근접하고 있음을 알 수 있다. 본 논문에서 사용한 예제의 데이터를 이용하여 앞절에서 제시한 성능평가의 척도인 정확도와 노이즈를 얻기 위한 시뮬레이션 결과에서 정규확률플롯을 이용하여 분석하였을 때 지수함수의 성질을 가장 잘 나타내던 Musa 3의 데이터가 가장 정확도의 측면이나 노이즈의 측면에서 우수하게 나왔으며 다음은 Musa 1의 데이터, Musa 2의 데이터 순으로 나타났다. 본

연구에서는 고장 데이터가 주어졌을 때 이러한 고장 데이터를 임의적으로 모델에 적용시키기 보다는 정규확률 플롯을 이용하여 고장 데이터에 대한 분포를 예측하고 예측된 결과를 이용하여 모델을 적용하면 정확한 모델 예측을 할 수 있으리라는 것이다. 본 연구에서 제시한 정규확률 플롯을 통한 예측결과와 시뮬레이션 결과는 동일하게 얻어졌다.

다음은 제시한 3개의 모델에 대하여 모델별 평가 결과를 살펴보면 Musa의 3개 데이터 모두 플롯된 형태로 보아서 지수함수의 성질을 만족하고 있다는 것을 확인할 수 있었다. 제시된 세 개의 모델 모두 지수함수의 기본 성질을 이용한 것으로 고장 데이터의 기본적인 특성에 따라서 다소 변형한 경우로 제시된 모델이다. 3개의 모델에 대하여 정확도의 측면에서는 지수함수의 성질을 가장 잘 설명해 주는 LV 모델의 성능이 가장 우수한 것으로 평가되었으며 다음은 KL 모델, LS 모델의 순으로 3개의 데이터 예제에서 모두 같은 결과를 얻었다.

본 결과는 (그림 1), (그림 2), (그림 3), (그림 4)에서 제시한 것과 같이 고장 데이터에 대하여 정규확률 플롯을 나타내고 나타난 형태를 통해서 분포를 예측하고 예측된 분포와 적합한 모델을 선택하게 되면 모델 선정을 통한 확률적인 모수추정도 오류를 줄일 수 있다.

## 5. 결론 및 향후과제

기존의 연구는 테스트링 과정에서 소프트웨어 고장 데이터가 주어지면 연구된 기존의 소프트웨어 신뢰도 성장 모델을 선택하고 선택된 모델의 모수를 추정하고 추정된 모수를 이용하여 신뢰도 함수를 구하는 방법으로 연구가 진행되어졌다.

그러나 본 논문에서는 소프트웨어 신뢰도 성장 모델은 확률적인 성질에 의하여 다양하게 연구되어 있으므로 테스트링 과정에서 얻어진 고장 데이터가 어떤 확률적 분포의 특징을 가지고 있는지 확인하여 모델을 선택하는 방법으로 연구를 진행하였다. 즉, 테스트를 통해서 얻어진 고장 데이터에 대하여 정규확률 점수를 구하고 구해진 정규확률 점수와 원 데이터를 이용하여 정규확률 플롯을 그려보고 그려진 결과로서 고장 데이터를 가장 잘 설명할 수 있는 모델에 적용을 시킨다면 더욱 더 정확한 모수추정이 가능하고 이와 같이 정확한 모수의 추정은 각 모델에 대한 신뢰도의 확률적인 추정값도 더욱 더 정확히 구해질 것이다.

Musa의 3개 데이터 모두 정규점수를 구해서 정규확률 플롯을 그려 본 결과 지수함수의 성질을 따른다는 것을 확인하고 지수함수와 관련있는 모델을 중심으로 모델 성능 평가를 실시하여 보았다. 제시된 3개의 모델은 모두 JM 모델의 지수분포의 확률적인 성질을 만족하는 기본형을 응용한 소프트웨어 신뢰도 성장 모델으로써 모수추정 방법에 따라서 다양하게 연구되어진 모델들이다.

특히 지수함수의 성질을 그대로 가지고 있는 LV 모델인 경우는 Musa의 고장 데이터를 설명하는 가장 적합한 모델

로 시뮬레이션 결과가 얻어졌다.

본 논문을 통하여 소프트웨어 신뢰도 성장 모델에 대한 연구를 위해서 고장 데이터에 대한 평가의 정량적인 측정치들을 얻기 위해서는 오류가 발견된 고장 데이터의 정규점수를 구해서 정규확률 플롯을 그려 보고 정규확률 플롯의 형태를 보고 고장 데이터의 분포를 예측한뒤에 가장 적합한 모델을 설정하는 방법이 모델의 성능평가에서 우수하다는 것을 확인할 수 있다. 이와 같은 방법을 통해서 연구가 진행되어지면 소프트웨어 신뢰도 성장 모델과 관련된 후속적인 연구에서도 오류를 줄일 수 있으리라 기대하여 본다.

앞으로 연구과제는 기존에 연구된 소프트웨어 신뢰도 성장 모델에 대한 다양한 특색과 조건을 검색하여 보고 실질적인 실험데이터를 이용하여 결과를 확인하는 연구를 하여야 할 것이다.

마지막으로 소프트웨어 신뢰도 성장 모델의 성능에 대한 평가 척도의 연구가 미비하다는 것이다. 본 논문에서는 기존에 연구된 정확도나 노이즈로 단순히 수치상의 비교를 통하여 모델에 대한 성능평가를 하였으나 앞으로 이러한 성능평가에 대한 척도는 더욱더 연구가 진행되어야 할 부분이다. 기존에 모수를 추정하고 추정된 모수에 대한 정확도의 평가는 다소 연구된 것이 있으나 모수의 성능에 대한 평가를 이용하여 모델에 대한 평가도 확장해서 연구가 이루어져야 할 것이다. 이러한 평가 척도의 기준을 소프트웨어 신뢰도 성장 모델에도 적용하여 활용하는 것이 좋을 것이다.

## 참 고 문 헌

- [1] A. Bertolino & L. Strigini, "On the use of testability measures for dependence assessment," *IEEE Trans. Soft, Eng.*, Vol.22, pp.97-108, 1996.
- [2] A. Csenki, "Bayes predictive Analysis of a fundamental Software Reliability Model," *IEEE Trans. Reliability*, Vol.39, No.2, pp.177-183, 1990.
- [3] A. L. Goel & K. Okumoto, "Time Dependent Error Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Trans. Reliability*, Vol.28, pp.206-211, 1979.
- [4] B. Littlewood & A. Sofer, "A Bayesian Modification to the Jelinski-Moranda software reliability growth model," *IEE/BCS Soft, Eng., Journal*, pp.31-41, 1987.
- [5] B. Littlewood & J. L. Verrall, "A Bayesian reliability model with a stochastically monotone failure rate," *IEEE Trans. Reliability*, pp.108-114, Vol.23, 1974.
- [6] B. Littlewood, "Forecasting software reliability," In *Software Reliability Modelling and Identification*, Ed. S. Bittanti, Springer-verlag, Berlin, pp.141-209, 1988.
- [7] C. N. Morris, "Parametric empirical Bayes inference : Theory and application," *J. American Statistical Association*, Vol.78, pp.47-65, 1983.
- [8] L. H. Crow & N. D. Singpurwalla, "An Empirically Deve-

veloped Fourier Series Model for Describing Software Failure," IEEE Trans. Reliability, Vol.33, pp.176-183, 1984.

[9] N. Langberg & N. D. Singpurwala, "A Unification of some Software Reliability Model," SIGM Journal on Scientific and Statistical Computation, pp.781-790, 1985.

[10] N. Davies, "The Musa data revisited : alternative methods and structure in software reliability modelling and analysis," In Achieving Safety and Reliability with Computer Sysms, Ed. B. K. Daniels, Elsevier, pp.118-130, 1987.

[11] P. Zeephongsekul & G. Xie. & S. Kumar, "Software-Reliability Growth Model : Primary-Failures Generate Secondary-Faults Under Imperfect Debugging," IEEE Trans. Reliability, Vol.43, pp.408-413, 1994.

[12] S. Campodonico & N. D. Singpurwalla, "A Bayesian Analysis of the Logarithmic Poisson Execution Time Model Based on Expert Opinion and Failure Data," IEEE Trans. Soft.Eng., Vol.20, No.9, pp.677-683, 1994.

[13] Z. Jelinski & P. B. Moranda, "Software Reliability Research, In Statistical Computer Performance Evaluation," New York, Academic Press, pp.465-484, 1972.



**정혜정**

e-mail : jhjung@ptuniv.ac.kr  
 1988년 경북대학교 통계학과(이학사)  
 1991년 경북대학교 통계학과(이학석사)  
 1994년 경북대학교 통계학과(이학박사)  
 1995년~현재 평택대학교 정보통계학과 부교수

2001년~현재 경기도정보화촉진위원, 한국산업표준심의회 S/W 공학분야전문위원(ISO/IEC JTC1/SC7/WG6 국제표준전문가 활동), 평택시정보화추진위원, 한국정보통신기술협회 인증심의위원 등

관심분야 : 소프트웨어공학, 소프트웨어 모형결정, 소프트웨어 품질특성, 소프트웨어 신뢰도 측정, 영상처리



**양해술**

e-mail : hsyang@office.hoseo.ac.kr  
 1975년 홍익대학교 전기공학과(학사)  
 1978년 성균관대학교 정보처리학과(석사)  
 1991년 日本 오사카대학교 정보공학과 S/W공학 전공(공학박사)  
 1975년~1979년 육군중앙경리단 전산실 시스템분석장교

1980년~1995년 강원대학교 전자계산학과 교수  
 1986년~1987년 日本 오사카대학교 객원연구원  
 1994년~1995년 한국정보처리학회 총무이사, 논문편집의원장  
 1995년~2002년 한국 S/W품질연구소 소장  
 1999년~현재 호서대학교 벤처전문대학원 교수  
 2001년~현재 한국정보처리학회 부회장  
 2003년~현재 미국 ACIS 학회 Vice President

관심분야 : 소프트웨어공학(특히, S/W 품질보증과 품질평가, 품질검리와 컨설팅, OOA/OOD/OOP, CASE, SD), 컴퍼넌트관리, CBD기반기술, IT품질경영



**박인수**

e-mail : ispark@ats.go.kr  
 1988년 원광대학교 전자과(학사)  
 1992년 원광대학교 전자계산기 공학과(석사)  
 1978년~1992년 전북지방공산품검사소 연구원

1992년~1994년 전남지방공업기술원 기술지원과장  
 1994년~현재 산업자원부 기술표준원 공업연구관  
 2000년~현재 한국산업기술대학교 컴퓨터공학과 겸임교수  
 2001년~현재 한국산업표준심의회 소프트웨어공학분야 전문위원 (ISO/IECJTC1/SC7/WG6 국제표준전문가 활동)

관심분야 : 소프트웨어공학(S/W품질평가), 임베디드 S/W, 계측·제어, 프로그래밍