Regular paper | Journal of the KIECS. pp. 935-944, vol. 18, no. 5, Oct. 31. 2023, t. 121, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2023.18.5.935

Inverse-type 수명분포에 근거한 유한고장 NHPP 소프트웨어 개발비용 모형의 성능에 관한 비교 연구

박승규*

Comparative Study on the Performance of Finite Failure NHPP Software Development Cost Model Based on Inverse-type Life Distribution

Seung-Kyu Park*

요 약

본 연구에서는 신뢰성 연구에 적합하다고 알려진 Inverse-type(: Inverse-Exponential, Inverse-Rayleigh) 수명분포를 유한고장 NHPP(: Nonhomogeneous Poisson Process) 기반의 소프트웨어 개발비용 모형에 적용한 후, 성능을 결정하는 속성을 분석하였다. 또한, 모형의 효율성을 평가하기 위해 Goel-Okumoto 기본 모형과 함께 비교하였다. 고장 시간 데이터를 이용하여 모형의 성능을 분석하였고, 모수의 계산은 MLE(: Maximum Likelihood Estimation)를 적용하였다. 결론적으로, 첫째, 개발비용을 결정하는 m(t)를 분석한 결과, Inverse-Exponential 모형이 참값에 대한 오차가 적어 효율적이었다. 둘째, 개발비용과 함께 방출시간을 분석한 결과 Inverse-Rayleigh 모형이 가장 좋은 것으로 확인되었다. 셋째, 제안된 모형의 속성(m(t), 비용, 방출시간)을 종합적으로 평가한 결과, Inverse-Rayleigh 모형의 성능이 가장 우수하였다. 따라서 소프트웨어 개발자가 초기 프로세스에서 본 연구 데이터를 효율적으로 활용할 수 있다면, 비용에 영향을 미치는 속성들을 사전에 탐색하고 분석할 수 있을 것이다.

ABSTRACT

In this study, the Inverse-type (: Inverse-Exponential, Inverse-Rayleigh) life distribution, which is known to be suitable for reliability research, was applied to a software development cost model based on finite failure NHPP(: Nonhomogeneous Poisson Process), and then the attributes that determine the model's performance were analyzed. Additionally, to evaluate the efficiency of the model, it was compared with the Goel-Okumoto basic model. The performance of the model was analyzed using failure time data, and MLE (: Maximum Likelihood Estimation) was applied to calculate the parameters. In conclusion, first, as a result of analyzing m(t), which determines the development cost, the Inverse-Exponential model was efficient due to its small error in the true value. Second, as a result of analyzing the release time along with the development cost, the Inverse-Rayleigh model was confirmed to be the best. Third, as a result of comprehensive evaluation of the attributes (m(t), cost, and release time) of the proposed model, the Inverse-Rayleigh model had the best performance. Therefore, if software developers can effectively utilize this research data in the early process, they will be able to proactively explore and analyze attributes that affect cost.

키워드

Development Cost Model, Finite Failure NHPP, Inverse-type Distribution, Performance Analysis
개발비용 모형, 유한고장 NHPP, 인버스 타입 분포, 성능 분석

* 교신저자 : 박승규
• 접 수 일 : 2023. 08. 27
• 수정완료일 : 2023. 09. 19
• 게재확정일 : 2023. 10. 17
• Received : Aug. 27, 2023, Revised : Sep. 19, 2023, Accepted : Oct. 17, 2023
• Corresponding Author : Seung-Kyu Park,
Email : skpark@nsu.ac.kr

Ⅰ. 서 론

창조와 혁신이 주도하는 4차 산업혁명 시대에서는 소프트웨어와 인공지능을 융합한 첨단기술이 우리 일상 속으로 빠르게 들어오고 있다. 인공지능 시대에는 다양하고 복잡한 데이터를 오류 없이 처리할 수 있는 신뢰성 높은 소프트웨어가 필요하다. 이 때문에 소프트웨어 개발자들은 고품질 소프트웨어를 개발하기 위해 신뢰성 연구에 집중하고 있지만, 개발비용도 큰 문제가 되고 있다. 따라서 이러한 문제를 해결하기 위해 소프트웨어 개발자들은 경제적인 비용으로 고품질의 안정적인 소프트웨어를 개발하기 위해 많은 시간과 노력을 투자하고 있다. 이에 NHPP(: Nonhomogeneous Poisson process)를 적용한 많은 소프트웨어 신뢰성 모델이 다양한 형태로 연구되고 있으며, 개선된 모델로 진화하고 있다[1].

본 연구에서 제시한 NHPP 기반 소프트웨어 신뢰성 비용모델 관련 연구들을 살펴보면 먼저, 소프트웨어 잔존 결함을 기반으로 최적의 방출시간 방법에 대한 전략이 제안되고 실제 데이터 적용을 통하여 검증되었다[2]. 테스트 범위를 포함한 모델을 통해 제품 신뢰성을 정량적으로 예측하고 소프트웨어 비용모델을 통해 요구 사항에 따라 예상되는 총비용을 최소화하는 방법이 제안[3]되었으며, 단계형 NHPP 모델을 활용하여 소프트웨어 시스템 비용 분석을 한 후 신뢰성 모형 선택에서 유용성을 입증되었다[4]. Gamma 계열 분포를 적용하여 소프트웨어 개발모형에서 비용과 시간의 속성을 이용한 최적의 소프트웨어 방출 전략이 제시[5]되었고, NHPP Burr-Hatke-Exponential 분포를 적용한 소프트웨어 개발모형에서 비용과 방출시간의 속성을 분석한 데이터가 제시[6]되었다. Gompertz 모형을 적용하여 소프트웨어 제품을 개발하는 과정에서 발생할 수 있는 비용 문제가 제시[7]되는 한편, NHPP 기반 Inverse-Exponential 신뢰도 모형의 성능과 관련된 새로운 속성 문제를 제시하고 이를 지수형 분포와 비교하여 해결하는 연구 등이 있었다[8].

본 연구에서는 다양한 형태의 수명분포를 설명할 수 있어 신뢰성 연구에 적합하다고 널리 알려진 Inverse -type 분포를 유한고장 NHPP 기반의 소프트웨어 개발비용 모형에 적용한 후, 성능을 결정하는 속성을 비교 분석하였다. 또한, 분석된 데이터를 통해 최적의 모형도 함께 제안하고자 한다.

Ⅱ. 관련 연구 및 기술

2.1 NHPP 소프트웨어 신뢰성 모형

NHPP는 주어진 시간을 적용하거나 단위당 일정 개수의 결함을 적용하여 성공한 발생 횟수를 근거로 미래의 발생 횟수를 예측하는 확률 기반형 모형이다. 이 모형은 결함이 발생하면 즉시 제거될 뿐만 아니라 새로운 결함이 발생하지 않는다고 가정하기 때문에 오류 탐지 측면에서 효율적인 것으로 알려져 있다. 소프트웨어 결함의 누적 개수를 N(t), 평균값 함수를 m(t)라고 가정하면, N(t)는 식 (1)과 같이 매개변수 m(t)를 갖는 포아송 확률 밀도를 따른다.

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

여기서, $m(t)$는 고장 발생 기댓값의 속성을 나타내는 평균값 함수이다.

$$m(t)=\int_0^t \lambda(s)ds \quad \cdots(2)$$

따라서, 고장 발생 강도의 속성을 나타내는 강도함수 $\lambda(t)$는 식 (3)과 같다.

$$\frac{d\,m(t)}{dt}=\lambda(t) \quad \cdots(3)$$

일반적으로, NHPP 모형은 결함을 수리하는 동안에는 고장이 발생하지 않는다는 유한고장과 결함을 수리하는 동안에도 고장이 발생한다는 무한고장으로 분류된다.

본 연구에서는 유한 고장(Finite Failure)을 기반으로 연구하고자 한다.

따라서, 유한고장 NHPP 모형에서 시간 t까지 발견될 수 있는 고장의 기댓값을 θ, 누적분포함수를 $F(t)$, 확률밀도함수를 $f(t)$라고 하면 신뢰도의 성능을 결정하는 속성 함수는 식 (4), (5)와 같다[9].

$$m(t)=\theta \cdot F(t) \quad \cdots(4)$$

$$\lambda(t) = \theta \cdot F(t)' = \theta \cdot f(t) \quad \cdots(5)$$

따라서, NHPP 모형의 우도함수(Likelyhood Function)는 식 (6)과 같다.

$$L_{NHPP}(\Theta|\ \underline{x}\ ) = \left(\prod_{i=1}^{n} \lambda(x_i)\right) \exp[-m(x_n)] \quad \cdots(6)$$

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

2.2 유한고장 NHPP Goel-Okumoto 기본 모형

소프트웨어 신뢰성 분야에서는 Goel-Okumoto 모형이 기본 모형으로 가장 잘 알려져 있다. 특히 Goel-Okumoto 기본 모형에서는 소프트웨어 결함별 고장 발생 시간의 분포를 따르는 수명분포가 지수 분포를 따른다고 알려져 있다. 따라서, 신뢰도 성능을 결정하는 속성 함수는 식 (7), (8)과 같다[10].

$$m(t) = \theta\,(1 - e^{-b\,t}\ ) \quad \cdots(7)$$

$$\lambda(t) = \theta\ b\ e^{-b\,t} \quad \cdots(8)$$

따라서, 식 (7), (8)을 식(6)에 대입하여 정리하면 로그 우도함수는 식 (9)와 같다.

$$\ln L_{NHPP}(\Theta|\underline{x}) = n\ln\theta + n\ln b - b\sum_{k=1}^{n} x_k - \theta\left(1 - e^{-b x_n}\right) \quad \cdots(9)$$

정리하면, 모수(θ, b)의 최우추정값 $\hat{\theta}_{MLE}$와 $\hat{b}_{MLE}$는 식 (10), (11)과 같이 이분법으로 계산할 수 있다.

$$\frac{\partial \ln L_{NHPP}(\Theta\,|\,\underline{x}\,)}{\partial\theta} = \frac{n}{\hat{\theta}} - 1 + e^{-\hat{b}\,x_n} = 0 \quad \cdots(10)$$

$$\frac{\partial \ln L_{NHPP}(\Theta\,|\,\underline{x}\,)}{\partial\theta} = \frac{n}{\hat{b}} - \sum_{i=1}^{n} x_n - \hat{\theta} x_n\, e^{-\hat{b}\,x_n} = 0 \quad \cdots(11)$$

2.3 유한고장 NHPP Inverse-Exponential 모형

신뢰성 연구에 적합한 것으로 알려진 Inverse-Weibull 분포는 의학 및 생태학 분야에서 널리 응용되고 있다. 특히 신뢰도 분석에서는 Inverse-Weibull 분포가 매우 일반적인 고장율을 모형화할 수 있다는 것은 이미 잘 알려져 있다. 따라서, Inverse-Weibull 분포의 F(t)함수는 식 (12)와 같다고 알려져 있다[11].

$$F(t) = e^{-(bt)^{-\gamma}} \quad \cdots(12)$$

본 연구에서 제안하는 Inverse-Exponential 분포는 식 (12)에서 형상모수(γ)가 1일 때 성립한다. 이에 따라, F(t) 함수는 식 (13)과 같이 도출되며, 미분하면 식 (14)와 같이 f(t) 함수를 구할 수 있다.

$$F(t) = e^{-(bt)^{-1}} \quad \cdots(13)$$

$$f(t) = F(t)' = b^{-1}t^{-2}e^{-(bt)^{-1}} \quad \cdots(14)$$

따라서, 신뢰도 성능을 결정하는 속성 함수는 식 (15), (16)과 같다[9].

$$m(t) = \theta\, e^{-(bt)^{-1}} \quad \cdots(15)$$

$$\lambda(t) = \theta\, b^{-1}t^{-2}e^{-(bt)^{-1}} \quad \cdots(16)$$

식 (15), (16)을 식 (6)에 대입하면 우도함수를 얻을 수 있다. 따라서 최대 우도 추정(MLE)을 적용하여 모수($\hat{\theta}_{MLE}$, $\hat{b}_{MLE}$)를 계산하는 로그 우도함수는 식 (17)과 같다.

$$\ln L_{NHPP}(\Theta|\underline{x}) = n\ln\theta - nlnb + 2\sum_{i=1}^{n} x_i - \sum_{i=1}^{n}(bx_i)^{-1} - \hat{\theta}\, e^{-(b x_n)^{-1}} = 0 \quad \cdots(17)$$

정리하면, 모수(θ, b)의 최우 추정값 $\hat{\theta}_{MLE}$와 $\hat{b}_{MLE}$는 식 (18), (19)와 같이 이분법으로 구할 수 있다.

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \theta} = \frac{n}{\hat{\theta}} - e^{-(\hat{b}\, x_n)^{-1}} = 0 \quad \cdots(18)$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial b} = -\frac{n}{\hat{b}} + \frac{1}{\hat{b}^2}\sum_{i=1}^{n}\frac{1}{x_i} - \hat{\theta}\,\frac{1}{\hat{b}^2 x_n}\, e^{-(\hat{b}x_n)^{-1}} = 0 \quad \cdots(19)$$

2.4 유한고장 NHPP Inverse-Rayleigh 모형

Inverse-Rayleigh 분포는 다양한 유형의 수명분포를 설명할 수 있기 때문에 신뢰성 연구에 널리 적용되고 있다. 특히, 여러 연구에서 Inverse-Rayleigh 분포를 적용함으로써 다양한 신뢰성 시험장치의 수명분포를 근사화할 수 있다고 하였다. 따라서, 척도 모수(b)를 고려한 F(t) 함수는 식 (20), (21)과 같다[12].

$$F(t) = \exp\left(-\frac{b}{t^2}\right) \quad \cdots(20)$$

$$f(t) = F(t)' = \frac{2b}{t^3}\exp\left(-\frac{b}{t^2}\right) \quad \cdots(21)$$

따라서, 신뢰도 성능을 결정하는 속성 함수는 식 (22), (23)과 같다.

$$m(t) = \theta\left[\exp\left(-\frac{b}{t^2}\right)\right] \quad \cdots(22)$$

$$\lambda(t) = \theta\left[\frac{2b}{t^3}\exp\left(-\frac{b}{t^2}\right)\right] \quad \cdots(23)$$

따라서, 식 (22), (23)을 식(6)에 대입하여 정리하면 로그 우도함수는 식 (24)와 같다.

$$\ln L_{NHPP}(\Theta|\underline{x}) = n\ln 2 + n\ln\theta + n\ln b + b\sum_{i=1}^{n}\ln\left(\frac{1}{x_i^2}\right) - b\sum_{i=1}^{n}\frac{1}{x_i^2} - \theta\exp\left(-\frac{b}{x_n^2}\right) \quad \cdots(24)$$

따라서, 정리하면, 모수(θ, b)의 최우추정값 $\hat{\theta}_{MLE}$와 $\hat{b}_{MLE}$는 식 (25), (26)과 같이 이분법으로 구할 수 있다.

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \theta} = \frac{n}{\hat{\theta}} - \exp\left(-\frac{\hat{b}}{x_n^2}\right) = 0 \quad \cdots(25)$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial b} = \frac{n}{\hat{b}} + \sum_{i=1}^{n}\ln\left(\frac{1}{x_i^2}\right) - \sum_{i=1}^{n}\frac{1}{x_i^2} + \frac{\hat{\theta}}{x_n^2}\exp\left(-\frac{\hat{b}}{x_n^2}\right) = 0 \quad \cdots(26)$$

2.5 유한고장 NHPP 소프트웨어 개발비용 모형

본 연구에서 전개한 평균값 함수 $m(t)$를 적용한, 유한고장 NHPP 소프트웨어 개발비용 모형은 식 (27)과 같이 각 구성 요소별 비용의 총합으로 구성된다[13].

$$E_t = E_1 + E_2 + E_3 + E_4 = E_1 + C_2\times t + C_3\times m(t) + C_4\times[m(t+t') - m(t)] \quad \cdots(27)$$

단, E_t는 개발 총비용, $m(t)$는 제안된 NHPP 모형의 성능을 결정하는 속성 함수이다.

따라서, 전체 소프트웨어 개발비용을 구성하는 각 구성 요소별 비용의 속성은 다음과 같다.

① E_1는 소프트웨어 초기 개발비용을 나타내며, 상수로 간주 된다.

② E_2는 단위 시간당 테스팅 비용으로, 실제로 적용되는 산업 분야별로 비용은 상이하다.

$$E_2 = C_2\times t \quad \cdots(28)$$

단, C_2는 단위 시간당 테스팅 비용이고, t는 테스팅 시점이다.

③ E_3는 한 개의 고장을 제거하는 비용이다.

$$E_3 = C_3\times m(t) \quad \cdots(29)$$

단, C_3는 테스팅 과정에서 발견된 1개의 고장을 제거하는 비용이다.

④ E_4는 소프트웨어 시스템에 남아 있는 잔존 결함들을 제거하는 비용이다.

$$E_4 = C_4 \times [m(t+t') - m(t)] \quad \cdots(30)$$

단, C_4는 소프트웨어 방출 후, 사용단계에서 운용자가 발견한 고장을 수리하는 비용이고, t'는 소프트웨어를 정상적으로 운용할 수 있는 시간이다.

여기서, C_4는 C_2와 C_3 보다는 현실적으로 높은 비용 구조를 갖기 때문에, 본 연구에서도 C_4 비용을 C_2와 C_3 비용보다 높게 설정하였다.

모든 소프트웨어 개발자들은 비용이 최소가 되는 시점에 소프트웨어를 출시하고자 할 것이다. 따라서, 최적의 소프트웨어 방출 시간은 소프트웨어 개발 비용이 최소가 되는 시점이 될 것이다. 즉, 다음 식 (31)을 만족해야 한다.

$$\frac{\partial E_t}{\partial t} = E' = (E_1 + E_2 + E_3 + E_4)' = 0 \quad \cdots(31)$$

III. 소프트웨어 고장시간을 이용한 개발비용 모형의 성능 분석

본 연구에서는 소프트웨어 시스템의 정상적인 운영 중에 수집된 고장 시간 데이터 (Software Failure Time Data)를 이용하여 제안된 모형의 속성을 분석하였다[14]. 표 1에서 제시한 데이터는 187.35시간 동안 고장이 30번 발생한 것을 나타내고 있다.

표 1. 수집된 소프트웨어 고장 시간 데이터
Table 1. Collected software failure time data

Failure number	Failure time(hours)	Failure time(hours)×10^{-1}
1	4.79	0.479
2	7.45	0.745
3	10.22	1.022
4	15.76	1.576
5	26.10	2.610
6	35.59	3.559
7	42.52	4.252
8	48.49	4.849
9	49.66	4.966
10	51.36	5.136
11	52.53	5.253
12	65.27	6.527
13	69.96	6.996
14	81.70	8.170
15	88.63	8.863
16	107.71	10.771
17	109.06	10.906
18	111.83	11.183
19	117.79	11.779
20	125.36	12.536
21	129.73	12.973
22	152.03	15.203
23	156.40	15.640
24	159.80	15.980
25	163.85	16.385
26	169.60	16.960
27	172.37	17.237
28	176.00	17.600
29	181.22	18.122
30	187.35	18.735

또한, 표 1에서 제시한 데이터가 본 연구에 적용 가능한지를 검증하기 위해 그림 1과 같이 라플라스 추세 테스트(Laplace Trend Test)를 사용하였다[15].

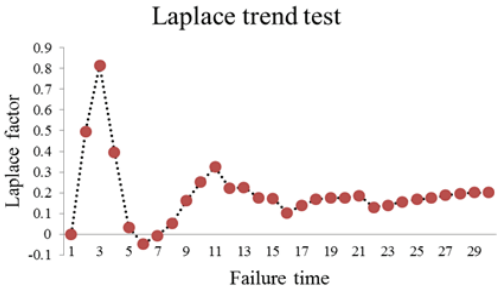

그림 1. 라플라스 추세 테스트의 결과
Fig. 1 Results of laplace trend test

일반적으로, 라플라스 추세 테스트의 결과가 "-2와 2" 사이에 분포하면, 극단 값이 존재하지 않고, 안정적이기 때문에 신뢰할 수 있다고 한다. 적용된 데이터의 분석 결과가 그림 1과 같이 "-2와 2 사이에 분포"하기 때문에, 신뢰할 수 있는 데이터가 된다.

표 2. 각 모형에 대한 모수추정
Table 2. Parameter estimation of each model

Type	NHPP Model	MLE	
		$\hat{\theta}$	$\hat{b}$
Basic	Goel-Okumoto	32.9261	0.1297
Inverse-type Distribution	Inverse-Exponential	41.2881	0.1692
	Inverse-Rayleigh	30.0100	1.6520

본 연구에서 제안된 모형의 모수($\hat{\theta}$, $\hat{b}$)의 추정은 최우 추정법(maximum likelihood estimation method, MLE)을 적용하였고, 그 결과는 표 2와 같다[16].

3.1 평균값 함수(m(t))의 속성 분석

그림 2는 참값(Real Value)을 예측하고 추정하는 성능을 나타내는 $m(t)$함수에 대한 속성 분석을 나타내고 있다[17]. 분석결과, Inverse-Exponential 모형이 참값에 대해 가장 작은 오차(에러)을 보여서, 다른 모형들보다 물리적으로 참값을 정확히 예측할 수 있기 때문에 신뢰도 성능 부분에서 효율적임을 알 수 있다.

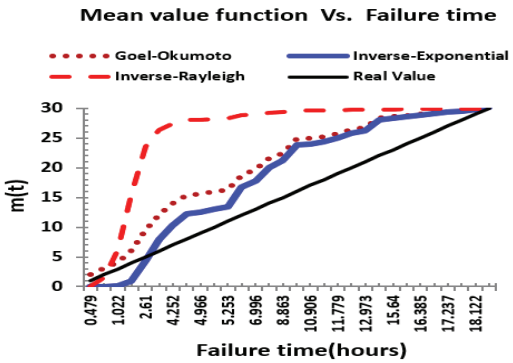

그림 2. m(t)의 속성 분석
Fig. 2 Attribute analysis of m(t)

3.2 개발비용과 방출시간의 속성 분석

본 연구에서는 실제 소프트웨어 개발 조건과 동일하게 시뮬레이션하기 위해 개발비용을 [가정 1] ~ [가정 4]와 같이 가정하였다. 이를 위해 전체 소프트웨어 개발비용(E_t)를 구성하는 각 비용 요소 (C_2, C_3, C_4)를 각각 2배씩 증가시켜서 총 소프트웨어 개발비용의 변화를 비교, 분석하고자 한다[18].

① [가정1 : 기본 조건]

$$E_1 = 50\$,\ c_2 = 5\$,\ c_3 = 1.5\$,\ c_4 = 10\$,\ t' = 50(\mathrm{H}) \quad \cdots(32)$$

[가정 1]과 같은 기본 조건을 적용한 시뮬레이션 결과는 그림 3과 같다. 분석 결과, 비용곡선의 패턴은 초기 단계는 크게 감소한 후, 방출시간이 흐를수록 점진적으로 증가하는 추이를 보이고 있다. 이러한 이유는 초기 단계에서는 소프트웨어에 내재한 결함을 발견될 확률이 높고, 쉽기 때문에 비용은 크게 감소하지만, 후반 단계에서는 남아 있는 결함이 검출되여 제거될 수 있는 물리적인 확률은 점점 감소하기 때문이다. 결국, 방출시간이 흐를수록 개발비용 곡선의 추이는 점진적으로 증가한다.

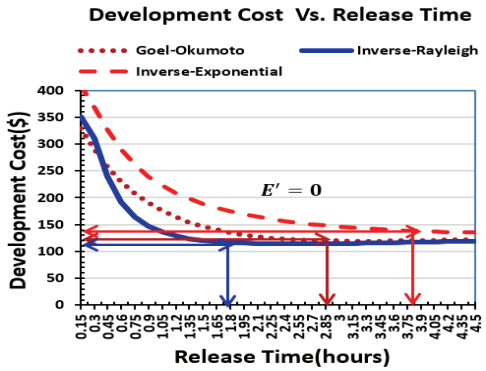

그림 3. 가정1의 개발비용과 방출시간의 속성 분석
Fig. 3 Attributes analysis of development cost and release time under assumption 1

그림 3과 같이 속성을 분석한 결과, Goel-Okumoto 모형의 개발비용이 120[$]일 때 방출시간은 2.85[h]이고, Inverse-Rayleigh 모형의 개발비용이 110[$]일 때 방출시간은 1.725[h]이고, Inverse-Exponential 모형의 개발비용이 140[$]일 때 방출시간은 3.825[h]이다. 따라서, 제안된 모형들 모두 유사한 패턴을 보였지만, Inverse-Rayleigh 모형이 개발비용이 낮고 방출시간이 빠르기 때문에 가장 효율적임을 알 수 있다.

② [가정2: 가정1에서 C_2값이 2배로 증가한 경우]

$$E_1 = 50\$,\ c_2 = 10\$,\ c_3 = 1.5\$,\ c_4 = 10\$,\ t' = 50(\mathrm{H}) \quad \cdots(33)$$

[가정 2]는 단위 시간당 테스팅 비용 (C_2)를 [가정 1]에 비해 2배로 증가시킨 상황이다. [가정 2]와 같은

조건을 적용한 시뮬레이션 결과는 그림 4와 같다.

그림 4과 같이 속성을 분석한 결과, Goel-Okumoto 모형의 개발비용이 140[$]일 때 방출시간은 2.85[h]이고, Inverse-Rayleigh 모형의 개발비용이 125[$]일 때 방출시간은 1.725[h]이고, Inverse-Exponential 모형의 개발비용이 160[$]일 때 방출시간은 3.825[h]이다. 즉, [가정 1]과 비교하여 개발비용은 증가했지만 방출시간은 전혀 변동되지 않았다.

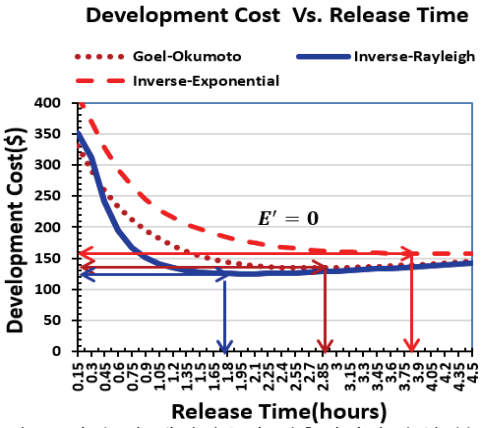

그림 4. 가정2의 개발비용과 방출시간의 속성 분석
Fig. 4 Attributes analysis of development cost and release time under assumption 2

이 경우, 소프트웨어 방출 전에 테스팅 비용이 증가하지 않도록 물리적으로 빠르고 정확한 테스트가 필요하다[19]. 따라서, Inverse-Rayleigh 모형은 제안된 다른 모형보다 개발비용이 낮고 방출시간이 빠르기 때문에 가장 효율적이다.

③ [가정3: 가정1에서 C_3값이 2배로 증가한 경우]

$$E_1 = 40\$,\ c_2 = 5\$,\ c_3 = 3\$,\ c_4 = 10\$,\ t' = 40(\mathrm{H}) \quad \cdots(34)$$

[가정 3]은 테스팅 과정에서 발견된 1개의 결함을 제거하는 비용 (C_3)를 [가정 1]에 비해 2배로 증가시킨 상황이다. [가정 3]과 같은 조건을 적용한 시뮬레이션 결과는 그림 5와 같다.

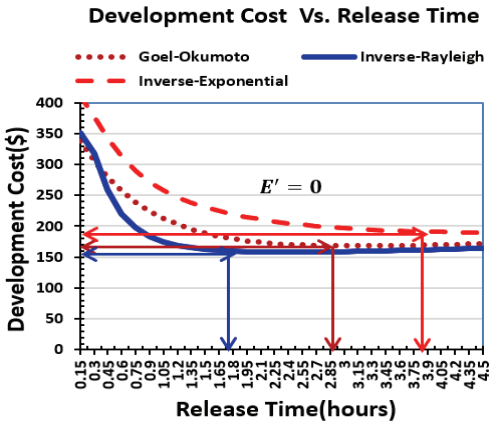

그림 5. 가정3의 개발비용과 방출시간의 속성 분석
Fig. 5 Attributes analysis of development cost and release time under assumption 3

그림 5와 같이 속성을 분석한 결과, Goel-Okumoto 모형의 개발비용이 170[$]일 때 방출시간은 2.85[h]이고, Inverse-Rayleigh 모형의 개발비용이 160[$]일 때 방출시간은 1.725[h]이고, Inverse-Exponential 모형의 개발비용이 190[$]일 때 방출시간은 3.825[h]이다. 즉, [가정 1]과 비교하여 개발비용은 증가했지만, 방출시간은 전혀 변동되지 않았다. 따라서, 이 경우 단위 결함을 제거하는 비용이 증가하지 않도록 소프트웨어 테스트 단계에서 물리적으로 가능한 많은 결함을 한번에 제거해야 될 것이다. 또한, Inverse-Rayleigh 모형은 다른 모형보다 개발비용이 낮고 출시 시간이 빠르기 때문에 상대적으로 효율적임을 알 수 있다.

④ [가정4 : 가정1에서 C_4값이 2배로 증가한 경우]

$$E_1 = 40\$,\ c_2 = 5\$,\ c_3 = 1.5\$,\ c_4 = 20\$,\ t' = 40(\mathrm{H}) \quad \cdots(35)$$

[가정 4]는 소프트웨어 출시 후 운영 단계에서 운영자가 발견한 결함을 수정하는 비용(C_4)을 [가정 1]에 비해 2배로 증가시킨 상황이다. [가정 4]와 같은 조건을 적용한 시뮬레이션 결과는 그림 6과 같다.

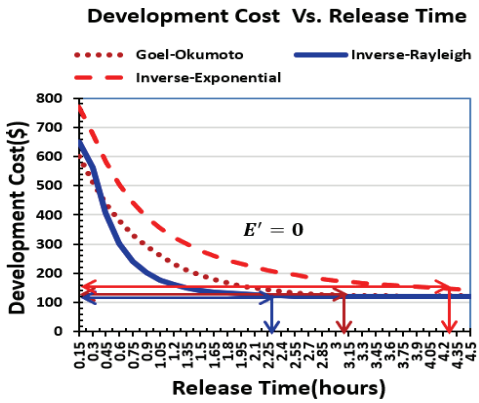

그림 6. 가정4의 개발비용과 방출시간의 속성 분석
Fig. 6 Attributes analysis of development cost and release time under assumption 4

그림 6과 같이 속성을 분석한 결과, Goel-Okumoto 모형의 개발비용이 140[$]일 때 방출시간은 3.075[h]이고, Inverse-Rayleigh 모형의 개발비용이 120[$]일 때 방출시간은 2.25[h]이고, Inverse-Exponential 모형의 개발비용이 160[$]일 때 방출시간은 4.275[h]이다.

[가정 2, 3]과 다르게 개발비용이 증가함에 따라 출시 시간도 함께 지연되는 것을 확인할 수 있다. 즉, 이 경우 소프트웨어를 출시하기 전에 가능한 모든 결함을 줄이기 위해서는 실제 운영 단계가 아닌 개발 테스팅 단계에서 물리적으로 가능한 모든 결함을 제거해야만 된다. 또한, Inverse-Rayleigh 모형은 다른 모형들보다 개발비용이 낮고 출시 시간이 빠르기 때문에 가장 효율적인 모형임을 확인하였다.

3.3 제안된 개발비용 모형의 속성 평가

표 3은 Inverse-type 수명분포를 적용한 NHPP 소프트웨어 개발비용 모형의 성능을 결정하는 속성을 비교하여, 평가한 결과를 보여주고 있다[20].

표 3. 제안 모형의 성능 비교
Table 3. Performance comparisson of Proposed model

Type	NHPP Model	Performance Evaluation		
		m(t)	Cost	Release Time
Basic	Goel-Okumoto	Best	Good	Good
Inverse-type Distribution	Inverse-Exponential	Best	Worst	Worst
	Inverse-Rayleigh	Good	Best	Best

본 연구에서 수행한 속성 연구는 제안된 개발비용 모형의 성능에 영향을 주는 중요한 요인이 된다. 소프트웨어 개발자가 초기 프로세스에서 본 연구 데이터를 효율적으로 활용할 수 있다면, 비용에 영향을 미치는 속성들을 사전에 탐색할 수 있을 것이다. 또한, 표 3를 분석한 결과, 제안된 모형 중 Inverse-Rayleigh 모형이 최적의 모형임을 확인하였다.

IV. 결론 및 향후 연구과제

소프트웨어 개발자가 초기 단계에서 수집한 고장 시간 데이터로 시스템의 신뢰성을 모델링할 수 있다면, 실제 운영 중에 발생할 수 있는 고장을 사전에 예측하고 보다 신뢰성 있는 소프트웨어를 설계할 수 있을 것이다. 따라서 소프트웨어의 고장을 사전에 예측함으로써 개발자는 보다 경제적인 비용으로 고품질의 소프트웨어를 효율적으로 개발할 수 있을 것이다. 이러한 근거로, 본 연구에서는 고장 시간 데이터를 활용하여 신뢰성 분석에 널리 적용되는 Inverse-type 수명분포에 근거하여 NHPP 소프트웨어 개발 모델의 성능과 속성을 새롭게 탐색하고 분석하였다.

본 연구의 결과는 다음과 같다.

첫째, 개발비용에 영향을 미치는 m(t)의 속성을 분석한 결과, Inverse-Exponential 모형이 참값 예측에 있어 오차가 작아서 효율적임을 알 수 있었다.

둘째, 본 연구에서 적용한 가정 조건에서 비용 요소(C2, C3, C4)를 각각 2배로 증가시켜 속성을 분석한 결과, Inverse-Rayleigh 모형이 모든 조건에서 성능이 가장 좋은 것으로 확인되었다.

셋째, 제시한 모형의 속성들(m(t), 비용, 방출시간)을 종합적으로 평가한 결과, Inverse-Rayleigh 모형의 성능이 가장 우수하였다.

결론적으로, 소프트웨어 개발자가 본 연구 데이터를 초기 단계에 활용한다면 신뢰성 분석과 함께 비용 속성을 효율적으로 탐색할 수 있을 것이다. 더불어, 소프트웨어 산업별 신뢰성 있는 고장 시간 데이터를 수집하고 이를 다양한 분포에 적용한 후 최적의 비용 모델을 찾는 후속 연구가 지속적으로 필요할 것이다.

감사의 글

이 논문은 2022년도 남서울대학교 학술연구비 지원에 의해 연구되었음.

References

[1] R. Lai, and M. Trivedi, “A Detailed Study of NHPP Software reliability models,” *Journal of Software,* vol. 7, no. 6, 2012, pp. 1296-1306.

[2] S, Chatterjee, J. Singh and A. Roy & A. Shukla, "NHPP-Based Software Reliability Growth Modeling and Optimal Release Policy for N-Version Programming System with Increasing Fault Detection Rate under Imperfect Debugging," *Proceedings of the National Academy of Sciences, India Section A,* vol. 90, no.1, 2020, pp. 11-26.

[3] H. Pham and X. Zhang "NHPP Software Reliability and Cost Models with Testing Coverage," *European Journal of Operational Research,* vol. 145, 2003, pp. 443-454.

[4] R. Shenbagam and Y. Sarada, "On a Cost and Availability Analysis for Software System Via Phase Type Non-homogeneous Poisson Process,” *Communications in Statistics - Theory and Methods,* 2023, pp. 1-22.

[5] J. Kim and Y. Yang, “Comparative Study on the Cost Analysis of Software Development Model Applicable to System Solutions Based on Gamma Family Distribution,” *International Journal of Electrical Engineering and Technology,* vol 12, Issue. 9, 2021, pp. 43-54.

[6] H. Kim, “A Comparative Study on the Cost of Software Development Model Based on Burr-Hatke-Exponential Distribution,” *International Journal of Engineering Research and Technology,* vol. 12, no.11, 2019, pp. 2036-2040.

[7] H. Kim and K. Kim, “Software Development Cost Model based on NHPP Gompertz Distribution,” *Indian Journal of Science and Technology,* vol. 8, no. 12, 2015, pp. 1-5.

[8] T. Yang, “Performance Analysis on the Reliability Attributes of NHPP Software Reliability Model Applying Exponential and Inverse-Exponential Lifetime Distribution,” *Journal of Theoretical and Applied Information Technology,* vol. 100, no. 22, 2022, pp. 6645-6656.

[9] H. Bae, “Performance Attributes Analysis of Software Development Cost Model with Gamma Family Distribution Characteristics,” *Journal of Theoretical and Applied Information Technology,* vol. 101, no. 10, 2023, pp. 3816-3826.

[10] T. Yang, “Comparative Analysis on the Reliability Performance of NHPP Software Reliability Model Applying Exponential-Type Lifetime Distribution,” *International Journal of Performability Engineering.* vol. 18, no. 10, 2022, pp. 679-689.

[11] T. Yang, “Comparative Study on the Performance Evaluation of Infinite Failure NHPP Software Reliability Model with Log-Type Distribution Property,” *ARPN Journal of Engineering and Applied Sciences.* vol. 17, no. 11, 2022, pp. 1209-1218.

[12] Y. Zhang and K. Wu, “Software Cost Model Considering Reliability and Time of Software in Use,” *Journal of Convergence Information Technology,* vol. 7, no. 13, 2012, pp. 135-142.

[13] T. Yang and J. Park, “A Comparative Study of the Software NHPP Based on Weibull Extension Distribution and Flexible Weibull Extension Distribution,” *International Journal of Soft Computing.* vol. 11, no. 4, 2016. pp. 276-281.

[14] Y. Hayakawa and G. Telfar, “Mixed Poisson-type Process with Application in Software Reliability,” *Mathematical and Computer Modelling.* vol. 31, no. 10-12, 2000, pp. 151-156.

[15] C. Lee. and H. Baek, "A Study on The Need for AI Literacy According to The Development of Artificial Intelligence Chatbot“ *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 18, no. 3, 2023. pp. 421-426.

[16] Y. Ju, J. Kim and E. Kim, "Development of

External Expansion Devices and Convergence Contents for Future Education based on Software Teaching Tools," *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 16, no. 6, 2021, pp.1317-1322.

[17] S. Lee, "A Routing Algorithm based on Deep Reinforcement Learning in SDN," *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 16, no. 6, 2021. pp. 1153-1160.

[18] Y. Song, Y. Lee and Y. Goo, "A Study on the Weapon System Software Reliability Testing for the Joint Tactical Data Link System Project Case," *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 17, no. 4, 2022. pp. 663-670.

[19] J. Seo, "A Study on Image Classification using Deep Learning-Based Transfer Learning," *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 18, no. 3, 2023. pp. 413-420.

[20] S. Park, "Comparative Analysis on the Performance of NHPP Software Reliability Model with Exponential Distribution Characteristics" *The Journal of The Korea Institute of Electronic Communication Sciences*, vol. 17, no. 4, 2022. pp. 641-648.

저자 소개

박승규(Seung Kyu Park)

1989년 연세대학교 전기공학과 졸업(공학사)
1991년 연세대학교 대학원 전기공학과 졸업(공학석사)
1996년 연세대학교 대학원 전기공학과 졸업(공학박사)
2017년~현재 남서울대학교 전자공학과 부교수
※ 관심분야 : 정보통신, 정보보안, 소프트웨어공학