소프트웨어 시험 노력 추정 시그모이드 모델

이 상 운†

요 약

소프트웨어 시험단계에 투입되는 노력의 분포를 추정하는 대표적인 모델로 Weibull 분포(Rayleigh와 지수분포 포함)가 있다. 이 모델은 시험 시작시점에서 실제로 많은 노력이 투입되는 점을 표현하지 못한다. 또한 다양한 형태를 갖고 있는 실제 시험 노력의 분포를 적절히 표현하지 못하고 있다. 이러한 문제점을 해결하기 위해 본 논문은 시그모이드 모델을 제안하였다. 신경망 분야에서 적용되고 있는 시그모이드 함수로부터 소프트웨어 시험 노력을 적절히 표현할 수 있도록 함수 형태를 변형시켰다. 제안된 모델은 다양한 분포 형태를 보이고 있는 실제 수행된 소프트웨어 프로젝트로부터 얻어진 6개의 시험 노력 데이터에 적용하여 적합성을 검증하였다. 제안된 시그모이드 모델은 기존의 Weibull 모델보다 성능이 우수하여 소프트웨어 시험노력을 추정하는데 있어 와이블 모델의 대안으로 채택될 수 있을 것이다.

Sigmoid Curve Model for Software Test-Effort Estimation

Sang-Un Lee†

ABSTRACT

Weibull distribution Iincluding Rayleigh and Exponential distribution is a typical model to estimate the effort distribution which is committed to the software testing phase. This model does not represent standpoint that many efforts are committed actually at the test beginning point. Moreover, it does not properly represent the various distribution form of actual test effort. To solve these problems, this paper proposes the Sigmoid model. The sigmoid function to be applicable in neural network transformed into the function which properly represents the test effort of software in the model. The model was verified to the six test effort data which were got from actual software projects which have various distribution form and verified the suitability. The Sigmoid model may be selected by the alternative of Weibull model to estimate software test effort because it is superior than the Weibull model.

키워드 : 시험노력(Test-Effort), Weibull 함수(Weibull Function), 시그모이드 함수(Sigmoid Function), 시험노력 패턴(Testing Effort Pattern)

1. 서 론

대형 소프트웨어 프로젝트의 1%만이 계획된 기간과 예상 비용한도 내에서 고객을 만족시키며 완료되었는데 반해, 대부분의 프로젝트들은 1년 이상의 일정이 지연되고 초기 예상 비용의 2배 정도가 초과되었다[1]. 이와 같은 이유로 인해, 프로젝트 관리 측면에서 소프트웨어 개발 및 유지보수 비용을 줄이고자 체계적인 연구가 수행되고 있으며, 소프트웨어 비용산정 및 프로젝트의 일정관리 모델을 개발하는 계기가 되었다.

소프트웨어 개발에 투입되는 총 노력의 규모를 추정하는 모델로는 LOC(Line Of Code)를 이용한 Boehm의 COCOMO (COnstructive COst MOdel) 모델[2, 3] 등과 기능점수(FP, Function Point)를 이용한 Albrecht[4, 5], Kemerer[6], Matson [7] 등이 있다. 이 연구들은 모두 소프트웨어 프로젝트의 생명주기 전반에 걸쳐 투입되는 총 개발노력과 비용만을 추정하는 것이며, 생명주기의 각 단계별로 투입되는 노력의 규모, 시간에 따른 투입 노력의 규모 변화를 추정할 수 없다. 이에 반해, 소프트웨어 생명주기 전체와 생명주기의 각 단계인 요구사항 분석, 설계, 코딩, 시험과 유지보수 단계에서 투입되는 노력이 시간에 따라 어떤 분포를 따르는지를 연구한 대표적인 사례는 Putnam[8]의 SLIM(Software LIfe cycle Management) 모델이 있다. 150개의 소프트웨어 프로젝트가 Norden[9]과 Putnam[8]에 의해 연구되었으며, 소프트웨어 생명주기 전반에 걸쳐 소요되는 총 노력뿐만 아니라 생명주기의 세부 단계별로 투입되는 노력도 Rayleigh 분포를 따름을 관찰하여 SLIM 모델을 제안하였다. 또한 Yamada et al.[10, 11]는 지수(Exponential), Rayleigh와 와이블(Weibull) 분포를 따르는 모델들을 제안하였다.

그러나 실제로 시험단계에 투입된 노력 분포는 다양한 패턴을 갖고 있어 Rayleigh, 지수 또는 와이블 분포를 따르지 않는 경우가 빈번히 발생한다. 또한, 와이블이나 Ray-

† 정 회 원 : 국립원주대학 여성교양과 교수
논문접수 : 2003년 8월 28일, 심사완료 : 2004년 5월 24일

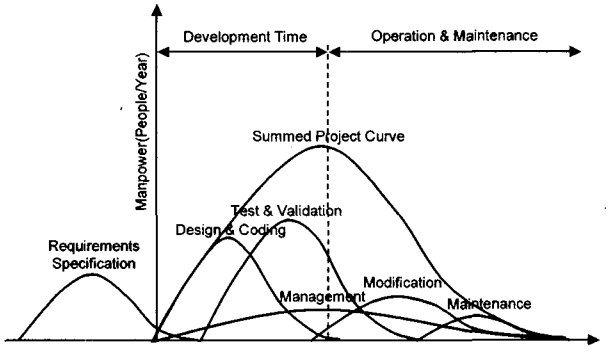
leigh 분포가 시험시간에 대한 함수인 경우 시험시작 초기 시점 ($t=1$)에서, 1 이하의 값을 얻기 때문에 이 시점에서 시험에 소요되는 노력의 양을 정확히 표현하지 못한다. 이러한 문제점들을 해결할 수 있는 시험노력 추정 모델이 필요하다.

본 논문은 소프트웨어 시험을 수행하는데 소요되는 노력 분포를 적절히 추정하기 위해 시그모이드 모델을 제안한다. 6개의 소프트웨어 시스템으로부터 얻어진 실측 시험 노력 데이터의 다양한 분포 형태를 대상으로 기존의 Rayleigh, 지수와 와이블 모델이 적절히 적용될 수 없음을 보인다. 다음으로 제안된 시그모이드 모델이 가장 적합함을 통계적 방법으로 검증한다.

2장에서는 시험노력 추정에 관한 연구들을 살펴보고, 문제점을 제시한다. 3장에서는 기존 제안된 모델들의 문제점들을 해결할 수 있는 모델을 제시한다. 이어서 4장에서는 시험단계에 투입되는 노력을 추정하는 모델들의 성능을 실제 데이터 집합들에 적용하여 모델들의 성능을 평가한다. 5장에서는 결론 및 향후 연구과제를 기술한다.

2. 기존 연구 및 연구배경

Norden[9]은 IBM에서 개발된 다양한 하드웨어 개발에 투입되는 노력은 근사적으로 Rayleigh 분포를 따른다는 사실을 관찰하였다. 이후 Putnam[8]에 의해 이 관찰 결과가 소프트웨어 프로젝트에 적용되었다. 150개의 소프트웨어 프로젝트가 Norden[12]과 Putnam[8]에 의해 연구되었으며, 소프트웨어 생명주기 전반에 걸쳐 소요되는 총 노력뿐만 아니라 생명주기의 각 단계 (즉, 요구사항 분석, 설계, 코딩, 시험 및 유지보수 등)에 투입되는 노력도 (그림 1)과 같이 Rayleigh 분포를 따름을 관찰하였다. 임의의 시간 $t_i(i=1, 2, \cdots)$ 시점까지 프로젝트에 투입된 누적 노력 W_i와 임의의 시간 t_i 시점에서 프로젝트에 투입되는 노력 dW_i/dt_i는 식 (1)로 표현된다.



(그림 1) 소프트웨어 개발노력 분포

$$W_i = E_{LC}(1-e^{-at^2}),\ \frac{dW_i}{dt_i} = 2E_{LC}ate^{-at^2} \quad (1)$$

여기서 E_{LC}(Life Cycle Effort, E_{LC})는 Rayleigh 곡선 아래에 있는 영역으로 넌 인원으로 표시되며, 소프트웨어 생명주기 전반에 걸쳐 투입된 총 노력이다. $a=1/2t_d^2$이며, t_d는 dW_i/dt_i가 최대가 되는 시점으로 경험적으로 볼 때, 시스템이 운영되는 시점에 근접한다. 따라서, $\frac{t_{dW_i}}{dt_i}\max = t_d$를 시스템의 개발기간이라 하며, 소프트웨어 생명주기 전반에 걸쳐 총 소요되는 노력의 약 40%에 도달하는 시점이다. $a=1/2t_d^2$로 치환하면, $\frac{dW_i}{dt_i}$는 식 (2)가 된다.

$$\frac{dW_i}{dt_i} = \frac{E_{LC}}{t_d}2te^{\frac{-t^2}{2t_d^2}} \quad (2)$$

소프트웨어 시험에 투입되는 총 노력을 E_T, 시험 수행 일자를 $t_i\,(i=1, 2, \cdots)$, 1일 동안 수행된 시험노력(Test Case 수, 투입된 시험노력, 시험시간 등)을 w_i라 하자. 한, t_i 시점까지 투입된 누적 시험노력을 W_i라 하자. 시험노력(Test-Effort)은 시험단계에서 소요되는 노력(Manpower), CPU 작동시간(Number of CPU Hours), 테스트 케이스의 수(Number of Test Cases) 이다. Putnam[8]의 SLIM 모델을 시험단계에 적용하려면 시험단계에서 최대로 투입되는 노력의 시점 t_d와 시험단계에 투입되는 총 노력을 추정해야만 하는 문제가 발생한다.

Yamada et al.[10]는 소프트웨어 시험 중 투입되는 시험노력과 소프트웨어 생명주기에 투입되는 총 노력의 규모가 식 (3)의 Rayleigh 분포를 따르는 모델을 제안하였으며, Rayleigh 분포의 대안으로 식 (4)의 지수 분포를 제안하였다. 여기서 α는 상수이다.

$$w_i = 2E_T a t_i e^{-at_i^2} \quad (3)$$

$$w_i = E_T a e^{-at_i} \quad (4)$$

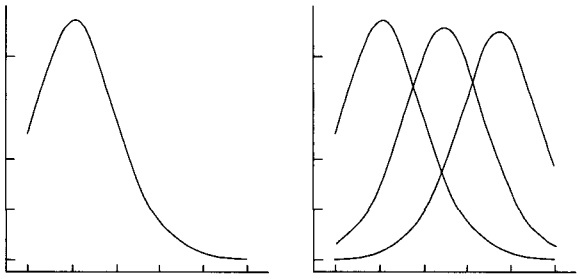
Yamada et al.[11]은 실제 투입된 시험 노력에 대해 지수분포나 Rayleigh 분포로 표현하는데 많은 문제점이 있음을 지적하여 소프트웨어 시험 중 결함 발견율은 단위시간에 투입되는 시험 노력의 규모에 비례한다는 가정하에 시험노력의 규모를 추정하는데 식 (5)의 와이블 분포를 따르는 모델을 제안하였다. 여기서, a는 분포의 규모(Scale)를 결정하는 모수, m은 분포의 모양(Shape)을 결정하는 모수이다.

$$w_i = E_T a m t_i^{m-1} e^{-at_i^m} \quad (5)$$

식 (5)의 와이블 분포에서 $m=1$이면 식 (4)의 지수분포가 되고, $m=2$이면 식 (3)의 Rayleigh 분포가 된다. 따라서,

제안된 모델들은 모두 와이블 분포를 따르는 모델이라고 할 수 있다.

이상과 같은 연구 결과를 토대로 할 때, 시험단계에 투입되는 시험노력은 (그림 2)(a)와 같이 일반적으로 와이블(지수와 Rayleigh 분포 포함) 분포를 따라야 한다. 그러나 실제로 시험단계에 투입된 노력의 분포는 (그림 2)(b)와 같이 다양한 형태를 따르고 있다.



(a) 와이블 분포 형태 (b) 다양한 분포 형태

(그림 2) 시험노력 분포 형태

3. 와이블 시험노력 추정 모델의 대안

3.1 시그모이드 모델

실제 시험노력 데이터 분포를 살펴본 결과 시험 시작 시점에서의 큰 투입 노력 값을 표현할 수 있으면서, 시간에 따른 시험노력의 분포를 보다 정확히 표현할 수 있는 모델이 필요하며, 기존의 와이블 분포로는 시험단계 투입노력 분포 표현이 불가능함을 알 수 있다. 이들 모델의 대안으로 식 (6)의 시그모이드 함수(Sigmoid Function) 형태를 적용할 수 있다.

$$W_i = \frac{1}{1+e^{-at_i}}, \quad w_i = \frac{e^{-at_i}}{(1+e^{-at_i})^2} \tag{6}$$

시그모이드 함수는 그리스 문자인 Sigma 글자와 같은 형태를 갖고 있어 불리워진 함수로 신경망에서 뉴론의 입력과 출력간의 비선형성 정도를 표현하는데 사용되고 있다. 식 (6)의 시그모이드 함수를 이용해 소프트웨어 시험 노력의 분포를 적절히 표현하기 위해 시험에 투입되는 총 노력 E_T를 곱하였다. 이 경우에도 실제 시험 노력의 분포를 적절히 표현하지 못할 수 있다. 따라서, 식 (6)의 a를 $(t_i-a)/b$로 변경하면 다양한 형태의 시험 노력 실측 데이터의 분포를 적절히 표현하는 유연성을 보일 수 있다. 이와 같이 식 (6)의 형태를 취하는 변형된 식 (7)의 모델을 제안하여 소프트웨어 시험노력을 추정한다. 여기서 a와 b는 상수이다. 본 제안 모델을 시그모이드 모델이라 칭한다.

$$w_i = \frac{E_T e^{-a}}{(1+be^{-a})^2}, \quad a = \frac{t_i-a}{b}, \quad a>0,\ b>0 \tag{7}$$

3.2 모수 추정

소프트웨어 시스템이 t_n 시점까지 시험이 진행되었다고 가정하자. 각 시험시간에 관련된 시험노력 $w_i, i=1,2,\cdots,n$ 데이터를 얻을 수 있다. 현재 시점에서 투입된 시험노력 추정과 $t_{n+d}, (d \geq 1)$ 시점에서 투입되어야 하는 노력을 예측하기 위해서는 먼저 모델에 있는 모수를 추정해야만 한다. 모수를 추정하는 방법들 중에서 최우추정법(Maximum Likelihood Estimation Method, MLE)을 많이 사용하고 있으나 수학적 어려움으로 인해 이의 대안으로 최소자승법(Least Squares Method)도 많이 적용하고 있다. 최소자승 추정치는 식 (8)을 최소화시킴으로써 얻어진다.

$$\sum_{i=1}^{n}[W_i-\overline{W_i})]^2 \tag{8}$$

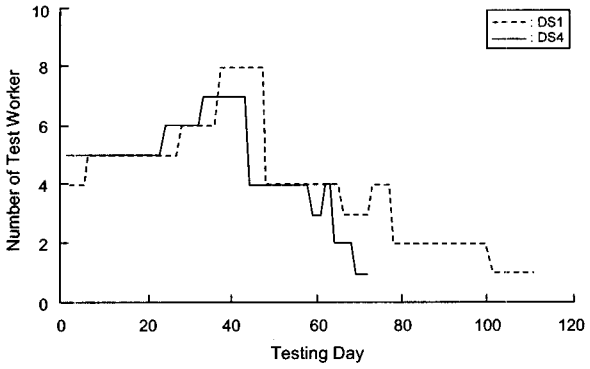
여기서, $\overline{W_i}$는 i 시점까지 투입된 시험노력의 누적 평균값이다.

4. 적용 예 및 평가

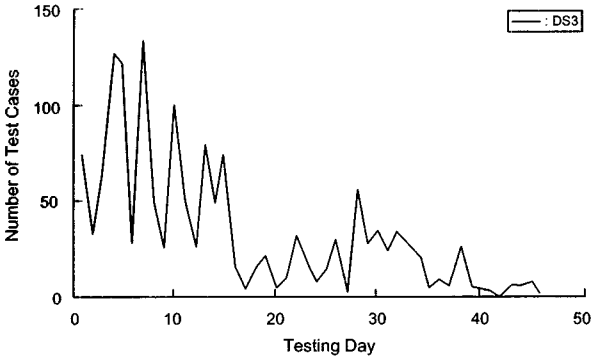
실제로 소프트웨어 시험단계에 투입된 노력의 분포 형태가 이들 분포를 따르는지 획득된 6개의 데이터를 고찰해보자. Data Set 1, Data Set 2,… Data Set 6을 각각 DS1, DS2,…DS6 이라 칭하였다. 6개의 데이터 중 DS1의 실제 데이터는 <표 1>과 같으며, DS1~DS6의 시험 시간에 따른 투입 인력의 규모를 그림으로 표현하면 (그림 3)~(그림 7)과 같이 다양한 형태로 표현된다.

<표 1> 실험에 적용된 DS1 데이터

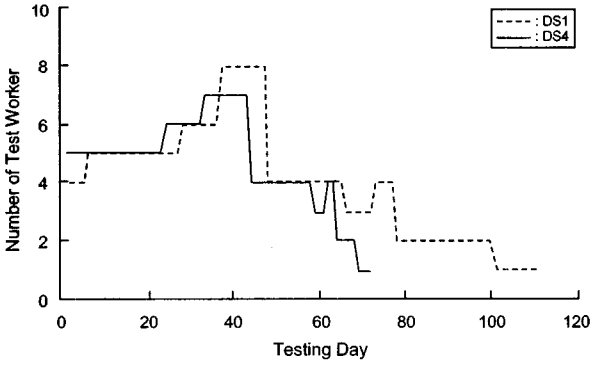
시험일	투입인력	시험일	투입인력	시험일	투입인력	시험일	투입인력
1	4	29	6	57	4	85	2
2	4	30	6	58	4	86	2
3	4	31	6	59	4	87	2
4	4	32	6	60	4	88	2
5	4	33	6	61	4	89	2
6	5	34	6	62	4	90	2
7	5	35	6	63	4	91	2
8	5	36	6	64	4	92	2
9	5	37	6	65	4	93	2
10	5	38	8	66	3	94	2
11	5	39	8	67	3	95	2
12	5	40	8	68	3	96	2
13	5	41	8	69	3	97	2
14	5	42	8	70	3	98	2
15	5	43	8	71	3	99	2
16	5	44	8	72	3	100	2
17	5	45	8	73	4	101	1
18	5	46	8	74	4	102	1
19	5	47	8	75	4	103	1
20	5	48	4	76	4	104	1
21	5	49	4	77	4	105	1
22	5	50	4	78	2	106	1
23	5	51	4	79	2	107	1
24	5	52	4	80	2	108	1
25	5	53	4	81	2	109	1
26	5	54	4	82	2	110	1
27	8	55	4	83	2	111	1
28	5	56	4	84	2		



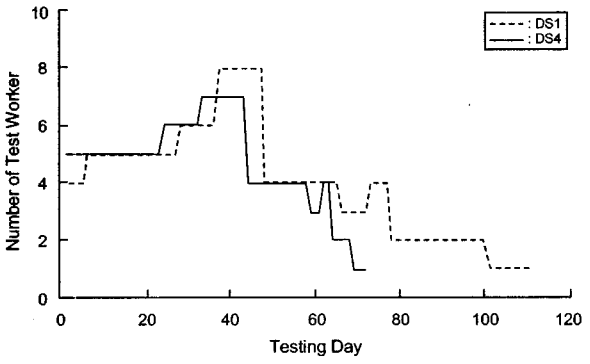
(그림 3) 시험단계 투입 인력



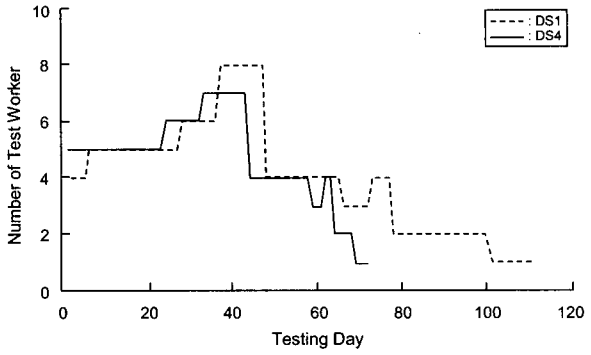
(그림 4) Test Cases



(그림 5) 시험과 디버깅 시간



(그림 6) 디버깅 시간



(그림 7) 시험단계 투입 인력

DS1은 실시간 제어 시스템으로 Tohma와 Tokunaga[13]으로부터 얻어졌으며 72일 동안에 481개의 결함이 발견되었고, 총 331명의 인력이 투입되었다. DS2는 Tohma, Jacoby, Murata와 Yamamoto[14]의 Table 3 데이터로 22일 동안 86개의 결함을 디버깅하였으며, 93시간이 투입되었다. DS3는 Tohma, Jacoby, Murata와 Yamamoto[14]의 Table 5 데이터로 46일 동안 266개의 결함을 발견하는데 1523개의 Test Case가 수행되었다. DS4는 Tohma, Jacoby, Murate와 Yamamoto[14]의 Table 1 데이터로 111일 동안 운영시험을 한 결과 481개의 결함이 발견되었으며, 이에 투입된 노력은 482명이다. DS5는 Musa, Iannino와 Okumoto[15]의 데이터로 실시간 명령을 수행하는 21,000 라인의 System T1 프로그램으로 21주 동안 고장 식별과 제거에 1141시간이 투입되었다. DS6는 Tohma, Tokunaga, Nagase와 Murata[16]의 Table 9 데이터로 16일 동안 198개의 결함을 발견하는데 289명의 인력이 투입되었다.

DS1, DS3와 DS4는 직관적으로 볼 때, 시험 시작 초반부에 많은 노력이 투입되어 Rayleigh나 와이블 분포와 편향된 결과를 보인다. DS3는 지수 분포를 따른다고 할 수 있으나 DS2, DS2, DS4와 DS5는 Rayleigh, 지수와 와이블 분포를 따르지 않는다고 할 수 있다. 따라서, 이들 데이터들에 와이블(Rayleigh와 지수 분포 포함) 분포를 이용해 투입되는 개발노력을 추정하면 많은 편차를 가진 결과를 얻을 수 있다. 결론적으로, (그림 2)~(그림 6)의 실제 데이터들을 분석하여 본 결과 기존 모델들을 적용시 다음과 같은 문제점이 발생한다.

- 실제 투입된 시험노력이 기존의 제안된 와이블(Rayleigh와 지수 분포 포함) 분포를 따르지 않는 경우가 빈번히 발생한다.
- 시험 시작시점에서 투입되는 시험노력의 양을 기존 제안 모델들은 거의 표현할 수 없다.

따라서, 시험 시작 시점에서 프로젝트에 투입되는 노력의 양을 보상해 줄 수 있으며, 시험단계에서 투입되는 시험노력의 분포도 보다 정확히 표현해 줄 수 있는 모델 개발이

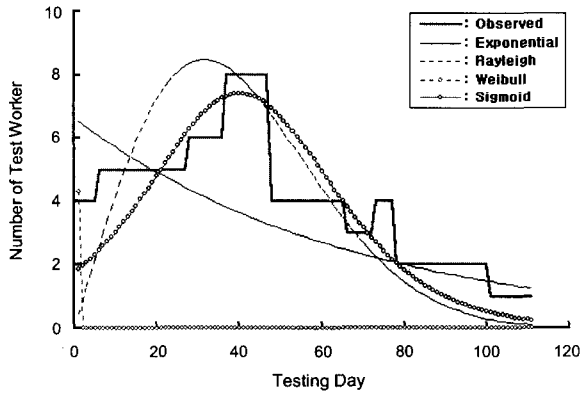
필요하다.

제안된 시그모이드 모델의 적합성을 평가하기 위해, (그림 3)~(그림 7)의 6개 데이터에 대해 Rayleigh, 지수, 와이블 ($m > 2$)과 시그모이드 모델을 적합시켜 보자. 6개의 데이터 (DS1~DS6)에 대해 통계적 분석도구를 이용하여 최소자승 추정법에 따라 모수를 추정한 결과 <표 2>의 값을 얻었다.

〈표 2〉 모수 추정

모 델	모수	Data Set					
		DS1	DS2	DS3	DS4	DS5	DS6
Weibull 식 (3)	$\hat{\alpha}$	8.105	0.001	0.001	7.629	8.310	0.001
	$\hat{m}$	4	3	3	4	3	3
Exponential 식 (4)	$\hat{\alpha}$	0.015	0.050	0.110	5.996	0.040	0.055
Rayleigh 식 (2)	$\hat{\alpha}$	0.001	0.008	0.001	0.001	0.007	0.010
Sigmoid 식 (7)	$\hat{\alpha}$	0.001	8.246	19.010	0.001	13.244	10.092
	$\hat{b}$	14.727	3.813	17.485	12.529	2.273	2.601

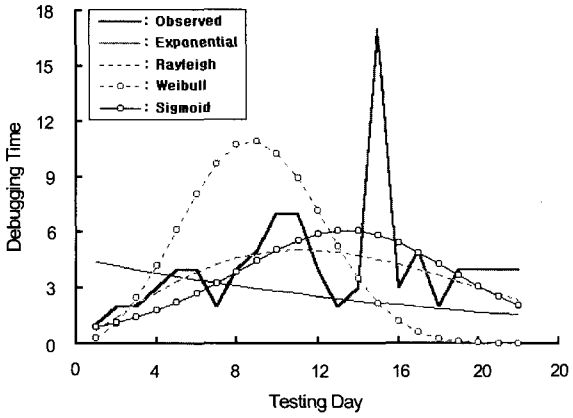
<표 2>의 모수 추정 결과를 적용하여 DS1에 대한 Rayleigh, 지수, 와이블과 제안된 시그모이드 모델의 추정 결과와 실측치(Observed Value)는 (그림 8)에 제시하였다.



(그림 8) 시간에 따른 시험인력의 실측치와 추정치(DS1)

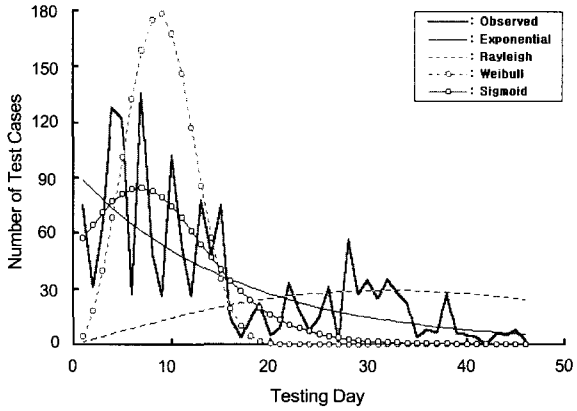
직관적으로 볼 때, DS1의 시험노력(시험인력)에 대해 와이블 모델은 시험노력 분포를 표현하는데 실패하였으며, 지수 모델은 시험 시작시점의 큰 노력의 양은 표현할 수 있었지만 시험노력의 분포 형태를 표현하지는 못하였다. 또한, Rayleigh 모델은 개발노력의 분포는 잘 표현하고 있으나 시험 시작시점에서의 개발노력의 양을 표현하지 못하며, 일정한 시점이 지난 후에야 개발노력의 양을 표현할 수 있음을 알 수 있다. 이에 반해, 시그모이드 모델은 시험노력의 분포 형태도 잘 표현할 수 있으면서 시험 시작 시점의 큰 노력의 양도 잘 표현하는 모델임을 알 수 있다. 따라서, DS1 데이터에는 시그모이드 모델이 가장 적합함을 알 수 있다.

DS2에 대한 모델의 추정 결과와 실측치는 (그림 9)에 제시하였다. 그림에서 DS2의 시험노력 (디버깅 시간)에 대해 지수 모델은 실측 데이터의 분포 형태를 표현하는데 실패하였으며, 와이블과 Rayleigh 모델은 시험노력이 최대가 되기 이전에 최대 값을 나타내면서 노력의 분포 형태를 적절히 표현하지 못하고 있다. 이에 비해, 시그모이드 모델이 시험 시작 시점에서의 노력의 양과 더불어 시간에 따른 노력의 분포형태 측면에서 볼 때 실측치에 가장 근접한 결과를 나타냄을 알 수 있다.



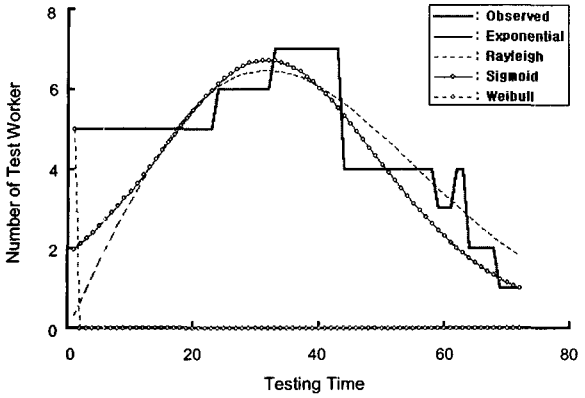
(그림 9) 시간에 따른 디버깅시간 실측치와 추정치(DS2)

DS3에 대한 모델의 추정 결과와 실측치는 (그림 10)에 제시하였다. 그림에서 DS3의 시험노력(Test Case 수)에 대해 와이블 모델은 시험노력 분포 형태를 과대추정함을 알 수 있으며, Rayleigh 분포는 실측 데이터를 표현하는데 실패하였다. 다만, 지수와 시그모이드 모델이 시험노력의 분포 형태와 시험 시작 시점의 노력의 양을 비교적 잘 표현하는 모델임을 알 수 있다.



(그림 10) 시간에 따른 Test Case 수 실측치와 추정치(DS3)

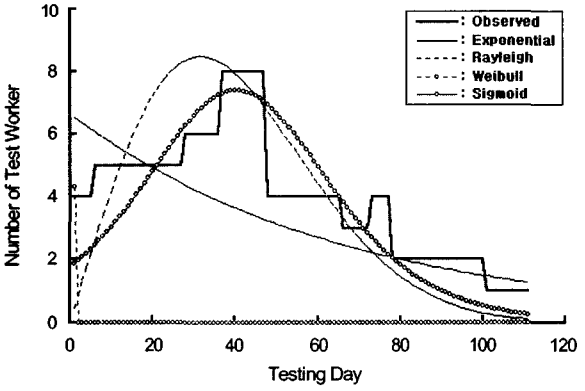
DS4에 대한 모델의 추정 결과와 실측치는 (그림 11)에 제시하였다.



(그림 11) 시간에 따른 시험인력의 실측치와 추정치(DS4)

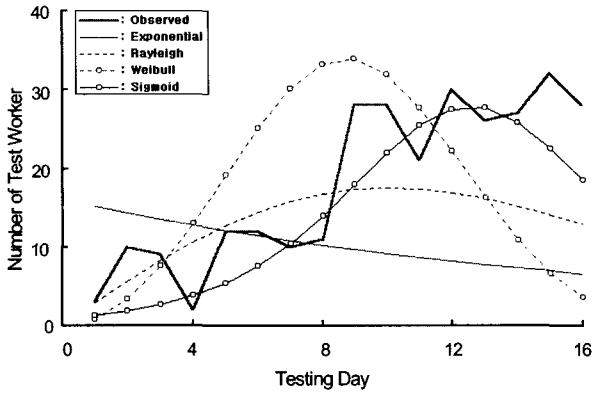
그림에서 DS4의 시험노력 (시험인력)에 대해 와이블과 지수 모델은 시험노력의 분포 형태를 추정하는데 실패하였으며, Rayleigh 모델은 시험노력의 분포 형태는 잘 표현하고 있으나, 시험 시작 시점의 투입 노력의 양을 일정 시점이 지난 후에야 적절히 표현할 수 있다. 이들 모델들에 비해, 시그모이드 모델은 시험노력의 분포 형태와 시험 시작 시점의 노력의 양도 적절히 표현하는 모델임을 알 수 있다.

DS5에 대한 모델의 추정 결과와 실측치는 (그림 12)에 제시되어 있다. 그림에서 DS5의 시험노력 (시험과 디버깅 시간)에 대해 와이블 모델은 적합이 불가하였으며, 지수와 Rayleigh 모델은 시험노력의 분포 형태를 추정하는데 실패하였다. 단지 시그모이드 모델이 시험노력의 분포 형태와 시험 시작 시점의 노력의 양도 모두 잘 표현하는 모델임을 알 수 있다.



(그림 12) 시간에 따른 시험과 디버깅시간의 실측치와 추정치(DS5)

모델의 추정 결과와 실측치는 (그림 13)에 제시하였다. 그림에서 DS6의 시험노력에 대해 지수 모델은 시험노력 분포 형태에 부적합하며, 와이블 모델과 Rayleigh 모델은 시험노력 분포 형태를 잘 표현하지 않고 있다. 단지 시그모이드 모델이 시험노력의 분포와 시험 시작 시점의 노력의 양도 모두 잘 표현하는 모델임을 알 수 있다.



(그림 13) 시험시간에 따른 시험노력의 실측치와 추정치(DS6)

지금까지 직관적으로 살펴본 바와 같이, 6개 데이터 집합 모두에서 제안된 시그모이드 모델이 시험단계 시작 시점에서의 큰 투입 노력의 양과 더불어 시험 수행과정에서의 시험노력 분포의 변동 형태를 가장 잘 표현하고 있음을 알 수 있어, 시험단계 투입노력 추정 모델로 적합함을 알 수 있다.

각 모델들을 이용해 6개 데이터 집합에 대해 추정된 총 시험노력 E_T는 <표 3>에 제시하였다. 표에서, 시험노력 분포 추정에 실패한 모델을 제외하고 나머지는 음영으로 표시되어 있다. 시험단계에 투입된 총 노력의 양을 비교한 결과 특정 모델이 가장 좋다고 결론을 내릴 수 없다.

직관적 모델평가와 더불어 모델의 성능을 보다 이론적 관점에서 평가해 보자. 모델의 성능을 비교하기 위해 MSE (Mean Squared Error)와 MMRE(Mean Magnitude of Relative Error)를 비교한다. n개의 데이터에 대한 $MSE = 1/n \sum_{i=1}^{n}(\text{실측치} - \text{추정치})^2$이다. 상대오차(Relative Error, RE)는 $RE = \frac{(\text{실측치} - \text{추정치})}{\text{실측치}} \times 100(\%)$로 구해지며, MRE(Magnitude of the RE) = $|RE|$이며, n개의 데이터에 대한 $MMRE$ (Mean MRE) $= \frac{1}{n}\sum_i MRE_i$, $i = 1, 2, \cdots, n$로 계산된다. $MMRE$가 작은 값이면 좋은 모델임을 알 수 있다.

〈표 3〉 총 시험노력 E_T 추정 결과

Data Set	실측치 (E_T)	추 정 치($\hat{E}_T$)			
		Weibull	Exponential	Rayleigh	Sigmoid
DS1	442	4.3279	355.6986	441.0809	409.6412
DS2	93	92.9990	60.5048	80.6750	82.1956
DS3	1523	1523.0000	1391.0513	1006.2756	1220.1125
DS4	337	4.9995	5.0413	312.6406	238.6871
DS5	1141	0.0000	635.8706	914.6798	1075.4490
DS6	289	285.8403	164.8609	214.7300	234.5302

6개 데이터 각각에 대해 모델들을 적용한 결과 얻어진 MSE와 각 데이터들에 대한 MSE가 적은 순서대로 순위를 정한 결과는 <표 4>에 제시되어 있다. <표 4>의 평균자승오차 측면에서 지수 모델이 DS3 데이터에서 우수한 성능을 보이 반면 나머지 5개 데이터에 대해서는 시그모이드 모델이 가장 좋은 결과를 나타내었다.

〈표 4〉 MSE와 모델 순위

Data Set	구 분	모 델			
		Weibull	Exponential	Rayleigh	Sigmoid
DS1	MSE	-	3.0097	2,5970	1.5430
	순위	-	3	2	1
DS2	MSE	12.1176	14.6855	9.0059	8.6739
	순위	3	4	2	1
DS3	MSE	-	0.6799	1.2244	0.7150
	순위		1	3	2
DS4	MSE	-	-	1.9240	1.3490
	순위			2	1
DS5	MSE	-	3367.7986	1751.1249	397.1746
	순위		3	2	1
DS6	MSE	169.0852	213.7098	85.2987	33.2671
	순위	3	4	2	1

6개 데이터 각각에 대해 모델들을 적용한 결과 얻어진 MMRE와 각 데이터들에 대한 MMRE가 적은 순서대로 순위를 정한 결과는 <표 5>에 제시하였다.

〈표 5〉 MMRE와 모델 순위

Data Set	구 분	모 델			
		Weibull	Exponential	Rayleigh	Sigmoid
DS1	MMRE	-	26.73%	41.91%	32.85%
	순위	-	1	3	2
DS2	MMRE	93.44%	61.26%	38.21%	48.16%
	순위	4	3	1	2
DS3	MMRE	-	15305.29%	57785.76%	385.75%
	순위	-	2	3	1
DS4	MMRE	-	-	26.14%	20.11%
	순위	-	-	2	1
DS5	MMRE	-	298.87%	251.38%	57.88%
	순위	-	3	2	1
DS6	MMRE	101.97%	100.99%	59.70%	76.50%
	순위	4	3	2	1

평균 절대상대오차 측면에서 DS1은 지수 모델이, DS2 데이터에서는 Rayleigh 모델이 좋은 결과를 보이고 있다. 그러나 (그림 7)에서 알 수 있듯이 DS1의 지수 모델은 시험 시작시점에서의 투입 노력의 양은 충분히 표현하였으나 시간에 따른 노력의 분포 형태를 적절히 표현하는데 실패하였다. 또한, DS2의 Rayleigh 모델은 (그림 8)과 같이 시험 노력이 최대가 되는 시점을 정확히 표현하지 못하는 단점을 갖고 있었다. 시그모이드 모델은 DS1과 DS2 데이터의 분포를 적절히 표현하는 성능을 보임과 더불어 나머지 4개 데이터들에서도 실측 데이터의 분포를 적절히 표현함과 동시에 가장 작은 상대오차를 나타내는 성능을 보였다.

결론적으로, <표 4>와 <표 5>의 모델 성능 비교 결과 대부분의 데이터 집합에서 시그모이드 모델의 성능이 가장 좋음을 알 수 있다. 따라서, 시험단계 투입 노력을 추정하는 모델로 시그모이드 모델이 가장 적합함을 알 수 있으며, 기존 제안된 와이블 모델들의 대안 모델로 적용될 수 있을 것이다.

5. 결론 및 향후 연구과제

소프트웨어 시험단계에 투입되는 시험노력의 분포를 추정하기 위해 와이블(Rayleigh와 지수분포 포함) 모델이 제안되었다. 그러나 실제로 다양한 시험노력 패턴으로 인해 이들 모델들이 적합하지 않는 경우가 빈번히 발생하고 있다. 또한, 시험 시작시점에서 투입되는 큰 시험노력 현상을 이들 모델들이 정확히 표현하지 못하는 단점을 갖고 있다. 이들 문제점을 해결하기 위하여 시그모이드 모델을 제안하였다.

6개의 실제 수행된 시험노력 데이터들에 적합시켜 본 결과, 직관적으로 볼 때 기존에 제안된 와이블 모델들은 대부분이 적합하지 않음을 보인 반면, 제안된 시그모이드 모델은 다양한 시험노력 패턴에 적합함을 보였으며, 또한, 시험 시작 시점에서 투입되는 시험노력의 양도 적절히 표현함을 보였다. 또한, 이론적인 측면에서 MSE와 MMRE 평가 결과 제안된 모델이 다양한 데이터들에 대해 가장 좋은 성능을 보였다. 따라서, 제안된 모델은 소프트웨어 시험노력을 추정하는데 있어 기존의 와이블 모델들의 대안 모델로 적합하였다.

본 논문은 시험단계에 투입되는 노력의 분포를 적절히 추정할 수 있는 시그모이드 모델을 제안하였다. 본 모델의 확장성 여부를 검증하기 위해 소프트웨어 생명주기 전반에 걸친 투입노력 분포 형태에도 적합한지 평가해볼 필요가 있다. 따라서, 본 제안된 모델이 소프트웨어 생명주기 전체에 소요되는 노력을 추정할 수 있는 모델로서, Putnam[8]의 SLIM 모델을 대체할 수 있는 모델인지에 대한 연구가 필요하며, 추후 이 분야에 대한 연구를 수행할 것이다.

참 고 문 헌

[1] K. H. Möller and D. J. Paulish, 'Software Metrics - A Practitioner's Guide to Improved Product Development', Chapman & Hall Co., New York, 1993.

[2] B. W. Boehm, 'Software Engineering Economics,' Prentice Hall, 1981.

[3] B. W. Boehm, "Software Engineering Economics," IEEE Trans. on Software Eng., Vol.10, No.1, pp.7-19, 1984.

[4] A. J. Albrecht and J. E. Gaffney, "Software Function, Source Line of Code and Development Effort Prediction : A Software Science Validation," IEEE Trans. on Software Eng., Vol.SE-9, No.6, pp.639-648, 1983.

[5] A. J. Albrecht, "Measuring Application Development Productivity," Proceedings SHARE/GUIDE IBM Applications Development Symposium, Monterey, CA., 1979.

[6] C. F. Kemerer, "An Empirical Validation of Software Cost Estimation Models," Communication ACM, Vol.30, No.5, pp.416-429, 1987

[7] J. E. Matson, B. E. Barrett and J. M. Mellichamp, "Software Development Cost Estimation Using Function Points," IEEE Trans. on Software Eng., Vol.20, No.4, pp. 275-287, 1994.

[8] L. H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," IEEE Trans. on Software Eng., Vol.SE-4, No.4, pp.345-361, 1978.

[9] P. V. Norden, "Project Life Cycle Modeling : Background and Application of the Life Cycle Curves," U. S. Army Computer System Command, 1977.

[10] S. Yamada, H. Ohtera and H. Narihisa, "Software Reliability Growth Models with Testing-Effort," IEEE Trans. on Reliability, Vol.R-35, pp.19-23, 1986.

[11] S. Yamada, J. Hishitani, and S. Osaki, "Software- Reliability Growth with a Weibull Test-Effort : A Model & Application," IEEE Trans. on Reliability, Vol.42, No.1, pp. 100-106, 1993.

[12] P. V. Norden, "Curve Fitting for a Model of Applied Research and Development Scheduling," IBM J. Research and Development, Vol.3, No.2, pp.232-248, 1958.

[13] Y. Thoma and K. Tokunaga, "A Model for Estimating the Number of Software Faults," Inst. Electron. Commun. Eng.(IECE) Japan. Tech. Rep., FTS 86-14, pp.41-46, 1986.

[14] Y. Tohma, R. Jacoby, Y. Murata and M. Yamamoto, "Hyper-Geometric Distribution Model to Estimate the Number of Residual Software Faults," COMPSAC '89, Orland, Florida, pp.610-617, 1989.

[15] J. D. Musa, A. Iannino, and K. Okumoto, 'Software Reliability Measurement, Prediction, Application,' McGraw-Hill Book Company, 1987.

[16] Y. Tohma, K. Tokunage, S. Nagage, and Y. Murata, "Structural Approach to the Estimation of the Number of Residual Software Faults Based on the Hyper-Geometric Distribution," IEEE Trans. on Software Eng., Vol.15, No. 3, pp.345-355, 1989.

이 상 운

e-mail : sulee@sky.wonju.ac.kr

1983년~1987년 한국항공대학교 항공전자공학과(학사)

1995년~1997년 경상대학교 컴퓨터과학과(석사)

1998년~2001년 경상대학교 컴퓨터과학과(박사)

1992년~2003년 국방품질관리소 항공전자장비 및 소프트웨어 품질보증 담당

2003년 도립 강원전문대학 컴퓨터응용과 전임강사

2004년~현재 국립 원주대학 여성교양과 전임강사

관심분야 : 소프트웨어 공학(소프트웨어 시험 및 품질보증, 소프트웨어 신뢰성), 소프트웨어 프로젝트 관리, 신경망, 뉴로-퍼지, Use-Case, RUP, CBD