유스케이스 점수 기반 소프트웨어 비용 추정

박 주 석[†]

요 약

소프트웨어 개발은 구조적기법에서 객체지향기법으로 전환되고 있다. 객체지향 소프트웨어 개발은 폭포수 프로세스가 아닌 반복적 프로세스 적용을 보다 선호하고 있으며, 유스케이스에 기반하여 요구사항을 도출하고, 이에 기반하여 분석, 설계와 코딩이 이루어지고 있다. 따라서, 유스케이스에 기반하여 개발될 소프트웨어의 규모가 추정되고 이에 기반한 개발노력, 비용과 개발기간이 추정되어야만 프로젝트 성공을 위한 관리가 가능해진다. 기존의 유스케이스 점수 관련 개발노력 추정 모델들은 선형과 비선형 모델들이 제안되었지만 유스케이스 점수의 규모에 따른 개발노력을 적절히 추정할 수 있는 모델이 없는 실정이다. 본 논문은 성장곡선을 적용해 유스케이스 점수에 대한 개발노력을 추정하는 모델을 적용한 결과 기존의 통계적 모델들보다 월등한 성능향상을 보였다. 따라서, 본 모델을 적용하여 개발노력을 추정함으로서 프로젝트 개발관리를 적절히 수행할 수 있을 것이다.

Software Cost Estimation Based on Use Case Points

Ju-Seok Park[†]

ABSTRACT

Software Development is converting from structural to object oriented method. The later software development prefers the iterative process applications, not waterfall process and based on use case model, the requirements are expressed and based on this, analysis, design and coding are accomplished. Therefore, size of the software to be developed is estimated basing on use case and it is only possible to maintain the project success by estimating development effort, cost and development period. Even though development effort estimation models related current use case point, there is no appropriate development effort estimating. This paper shows, as a result of applying the development effort estimating model about UCP to the growth curve, a superior performance improvement to current statistical models. Therefore, estimation of development effort by applying this model, project development maintenance can be appropriately carried out.

키워드 : 유스케이스점수(Use Case Point), 개발비용(Development Cost), 회귀모델(Regression Model), 성장곡선(Growth Curve), Gompertz 곡선(Gompertz Curve)

1. 서 론

소프트웨어 개발비용 추정 분야는 20년 전부터 관심의 대상이 되어 왔으며, 비용에 영향을 미치는 인자에 기반을 두고 개발비용을 설명하기 위한 많은 방법들이 연구되고 실제 적용되고 있다. 그러나 이 분야에 대한 집중적인 연구활동에도 불구하고 현재까지도 일반화된 결론을 유도하지 못하고 있는 실정이다[1, 2]. 소프트웨어 비용 추정은 소프트웨어 시스템을 구축하는데 소요되는 시간인 노력의 양을 예측하는 과정이다[3].

소프트웨어 프로젝트에 소요되는 노력, 비용과 개발일정을 적절히 추정하기 위한 핵심적인 인자 중 하나로 소프트웨어 규모에 대한 정량화가 일반적으로 인식되고 있다[4]. 소프트웨어 규모는 길이, 기능성과 복잡도 인자들에 의해 정의될 수 있으며[4], 라인 수와 기능점수가 일반적으로 사용되고 있다[3-6]. 기능점수 방법[6]은 데이터 관리 위주인 경영정보시스템 소프트웨어에 적합하도록 설계되었다. 이에 따라 기능점수 기법을 제어 위주인 실시간 시스템과 내장형 시스템으로 적용 범위가 확장되었고 이를 완전기능점수라 부르고 있다.

기능점수나 완전 기능점수는 구조적 개발 방법에 적합한 기능성에 기반을 두고 있어 트랜잭션 처리나 파일 형태에 적합하며, 자료흐름도, 계층적 프로세스 모델 또는 데이터베이스 구조와 같은 전통적인 구조적 설계기법으로부터 유도된다[4].

대부분의 소프트웨어 개발의 실패 원인이 개발 초기에 사용자의 요구사항 도출 어려움에 기인하여 이를 해결하려는 연구가 계속되었다. 1992년 Jacobson[7]에 의해 요구사항 도출방법으로 유스케이스가 제안되었고, 1998년 UML(Unified Modeling Language)에 유스케이스도(Use Case Diagram)

† 정 회 원 : 국방대학교 직무연수부 교수
논문접수 : 2004년 11월 15일, 심사완료 : 2004년 12월 13일

가 포함된 이후 소프트웨어 산업계에서는 유스케이스-구동(Use Case-driven), UML 기반의 단일화된 프로세스(Unified Process)로 객체지향 소프트웨어를 개발하고 있다[4].

Karner[8]는 Albrecht[6]의 기능점수 계산방법을 수정(Modify)하여 유스케이스 점수(Use Case Point, UCP)를 계산하는 방법을 제시하였다. 이후에도 유스케이스 점수에 대한 연구는 계속되고 있으나 유스케이스 점수에 기반하여 개발노력을 추정할 수 있는 모델 연구는 미미한 수준이다.

따라서 본 논문은 소프트웨어 개발에 소요되는 노력을 추정함에 있어, 객체지향 개발방법론에 적합한 유스케이스 점수로 측정된 소프트웨어 규모에 따른 개발노력을 추정할 수 있는 통계적 모델을 제시한다.

2장에서는 소프트웨어 규모 추정 기법의 관련 연구 및 문제점을 살펴보고, 3장에서는 성장곡선을 이용한 개발 노력 추정 모델을 제시하고 적합성을 평가한다. 4장에서는 결론 및 향후 과제를 제시한다.

2. 관련 연구 및 문제점

본 장에서는 유스케이스 점수에 기반하여 소프트웨어 개발노력을 추정하기 위한 기존 연구들과 문제점을 살펴본다.

2.1 통계적 개발노력 추정 모델

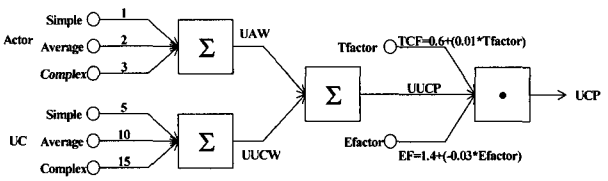
추정된 개발노력 (시간)에 대해 개발업체의 단위 시간당 평균 소요 비용을 곱하면 개발비용이 산출된다. 이와 같은 이유로 인해 일반적으로 개발노력 추정 모델을 개발비용 추정 모델이라 부른다[3]. 소프트웨어 노력추정 모델들은 일반적으로 주요 노력인자인 규모와 부수적인 조절인자로 구성되어 있다. 전형적인 통계적 노력 추정 모델은 과거 개발된 소프트웨어 프로젝트로부터 수집된 데이터에 대한 회귀분석으로부터 유도되며, 모델의 구조는 식 (1)의 선형과 비선형 형태로 표현된다[9].

$$E = aS + b \text{ 또는 } E = a + bS^c \qquad (1)$$

여기서 E는 개발노력 (Person-months 또는 Person-Hours), a, b, c는 경험적으로 유도된 상수이며 S는 주요 노력인자인 소프트웨어 규모로 LOC, FP, FFP 또는 UCP가 될 수 있다.

2.2 유스케이스 점수 계산 방법

유스케이스 점수(Use Case Point)의 개략적인 계산 과정은 (그림 1)에 제시되어 있다. 먼저, 각 액터(Actor)의 복잡도(Simple, Average, Complex) 가중치가 합해져 UAW (Unadjusted Actor Weights)가 계산되고, 각 유스케이스의 복잡도 가중치가 합해져 UUCW(Unadjusted Ucs Case Weights)가 구해진다. 이어서 UAW+UUCW = UUCP (Unadjusted Use Case Point)가 계산된다. 여기에 기술 복잡도 요인(TCF, Technical Complexity Factor)과 환경요인(EF, Environmental Factor)이 곱해져 AUCP(Adjusted Use Case Point)가 최종적으로 구해진다. AUCP를 일반적으로 UCP라 부른다.



(그림 1) UCP 계산과정

2.3 유스케이스 점수 기반 개발노력 추정 모델

2.3.1 선형 모델

식 (1)과 같은 일반적인 선형이나 비선형 개발노력 추정 모델에 비해 UCP를 이용하여 개발노력을 얻기 위해 식 (2)와 같이 단순한 선형 형태의 연구가 다양하게 진행되었다.

$$E = AUCP \times \text{유스케이스 점수 당 개발 소요시간} \qquad (2)$$

유스케이스 점수 당 개발 소요시간은 개발조직의 생산성에 의존한다[10, 11]. Banerjee[10]는 실제 적용 경험에 의하면 UCP당 15~30시간의 범위를 갖고 있다고 제안하였으며, Probasco[11]는 20시간부터 시작하여 개발조직의 생산성에 따라 조절해야 함을 제시하였다. Karner[8]는 20시간을, Schneider와 Winter는 $EF \leq 2$인 경우 20시간, $3 \leq EF \leq 4$인 경우 28시간, $EF \geq 5$인 경우 위험요소가 허용이 불가능할 정도로 크므로 EF를 조절하는 것이 필요함을 제시하였다[10]. Ribu[4]는 36시간을 적용하는 특별한 경우도 경험하였으며, 학생들이 수행한 프로젝트에는 10시간을 적용하였다. Sun사는 30시간을 적용하였다[10].

유스케이스 점수에 기반을 둔 개발노력 추정 기존 연구들은 AUCP에 상수를 곱하여 개발노력을 계산하였다. 즉 개발노력은 AUCP에 선형적인 관계를 갖는 식 (2)는 프로젝트의 규모에 상관없이 UCP 당 일정한 시간이 투입됨을 의미하나 이는 현실적으로 적합하지 않다. 왜냐하면 프로젝트의 규모(소형, 중형, 대형)가 커지면 프로그램의 복잡도(단순, 중간, 복잡)가 기하급수적으로 증가하는 경향을 띠고 있어 개발에 투입되는 시간도 프로젝트 규모에 따라 기하급수적으로 증가하는 비선형적인 관계를 나타낸다. 따라서 선형회귀모델 보다는 비선형 회귀모델이 보다 적합할 수 있다.

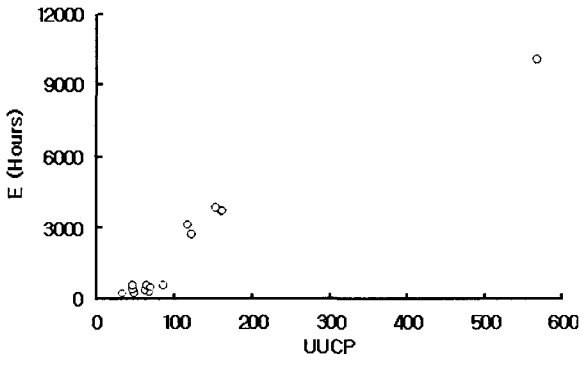
2.3.2 비선형 모델

박주석 et al.[12]은 Ribu[4]와 Nageswaren[13]의 연구결과로부터 얻은 <표 1>의 데이터를 이용하여 개발노력을 추정하는 비선형 모델을 제시하였다. 이 모델은 TCF와 EF를 고려하지 않고 직접 UUCP로부터 개발노력을 추정할 수

있는 통계적 모델이다. <표 1>에서 Project A는 (그림 2)에서 알 수 있듯이 다른 데이터들에 비해 UUCP와 Effort 값이 월등히 큰 이상점(Outliner)을 갖고 있어 이 데이터를 포함시켜 모델을 유도하면 편향된 결과를 얻을 수 있다.

〈표 1〉 유스케이스 점수 데이터

프로젝트		Actor UAW	Use Case UUCW	UUCP	AUCP	Staff-Hour per UCP	Effort
S. Nageswaren[19]		55	64	119	176.68	20	3,120
Ribu [4]	Project B-Subsystem 1	13	150	163	146.90	28	3,723
	Project B-Subsystem 2	11	150	161	145.10	28	3,665
	Project B-Subsystem 3	10	145	155	139.69	28	3,835
	Project B-Subsystem 4	13	110	123	110.85	28	2,710
	1-Q	8	60	68	47.97	10	294
	2-S	9	40	49	32.13	10	298
	3-S	9	40	49	32.13	10	232
	4-Q	7	55	62	43.74	10	371
	5-Q	7	40	47	33.16	10	420
	6-S	7	80	87	57.05	10	580
	7-Q	9	25	34	23.99	10	243
	8-S	11	55	66	43.28	10	595
	9-S	7	40	47	30.82	10	578
	10-Q	9	60	69	48.68	10	490
	Project A	6	560	566	541.55	10	10,043

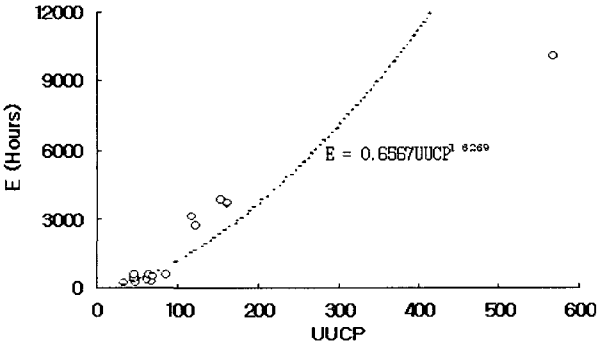


(그림 2) UUCP에 따른 개발노력 분포

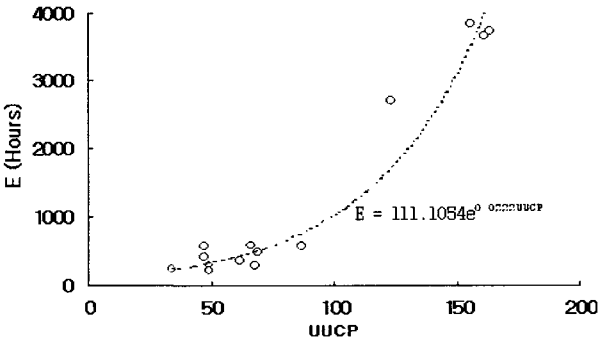
적합한 통계적 회귀모델을 선정하기 위해 결정계수(Coefficient of determination, R^2)[14]를 적용하며, 모델의 성능을 평가하기 위해 Briand et al.[1]이 적용한 MMRE (Mean Magnitude of Relative Error)를 적용하였다. 종속변수의 값(E)은 독립변수(UUCP)에 의해 결정되는 부분과 미지의 오차의 합으로 나타나며, 총 변동을 설명하는데 있어서 회귀직선에 의해 설명되는 변동 비율을 결정계수라 하며, 값이 클수록 쓸모 있는 회귀직선이 되지만 좋은 모델로 선정하기 위한 기준은 없는 실정이다. 모델의 정확도(Accuracy)를 평가하는 MMRE는 작은 값이면 평균적으로 좋은 모델임을 알 수 있다.

Project A를 포함한 경우에 대해 UUCP에 따른 개발노력 추정모델은 (그림 3)에, Project A를 제거하고 나머지 데이터들만을 이용하여 UUCP에 따른 개발노력 추정 모델은 (그림 4)에 제시되어 있다.

모델의 성능 분석 결과는 <표 2>에 제시되어 있다. Project A를 포함한 경우 누승 함수형 모델이 가장 적합한 모델로 선정되었고, Project A를 제거시 지수 함수형 모델이 MMRE가 가장 적은 결과를 나타내었다.



(그림 3) 프로젝트 A 포함시 개발노력 추정 모델



(그림 4) 프로젝트 A 제거시 개발노력 추정 모델

〈표 2〉 개발노력 추정 모델 성능

구 분	모델형태	모델	결정계수	MMRE
Project A 포함시	누승	$E = 0.6567 \cdot UUCP^{1.6269}$	0.8571	43.42
Project A 제거시	지수	$E = 111.1054 \cdot e^{0.0222 \cdot UUCP}$	0.9152	25.85

기존 연구들이 수행한 AUCP에 상수를 곱한 형태의 모델들과 비교한 결과 비선형 모델이 월등한 성능향상을 보였다. 따라서 이 연구 결과로부터 소프트웨어 규모와 개발노력간의 관계는 일반적인 개발노력 추정 모델과 같이 선형보다 비선형 형태가 보다 적합함을 알 수 있다.

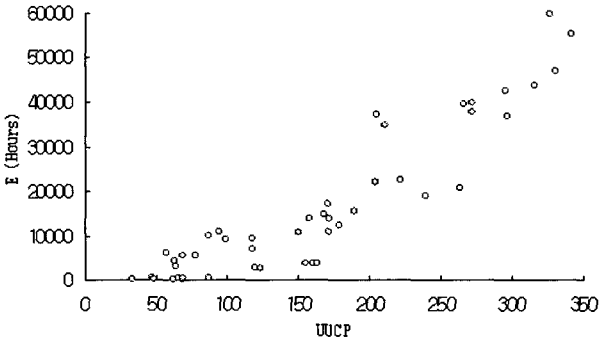
2.3.3 기존 모델의 문제점

박주석 et al.[12]이 제안한 비선형 모델은 적용된 데이터의 개수가 매우 적고 소프트웨어 규모가 매우 작은 부분에 한정되어 일반화된 모델 제시로는 불충분하다. 소프트웨어 규모가 큰 부분들에서도 이들 제시된 모델들이 적합하다면 일반화된 모델로 적용될 수 있을 것이다. 따라서, 이 부분을 고찰하여 보자. 국내 개발업체들로부터 <표 3>의 데이터를 추가로 획득하였다.

〈표 3〉 유스케이스 점수 관련 국내 데이터

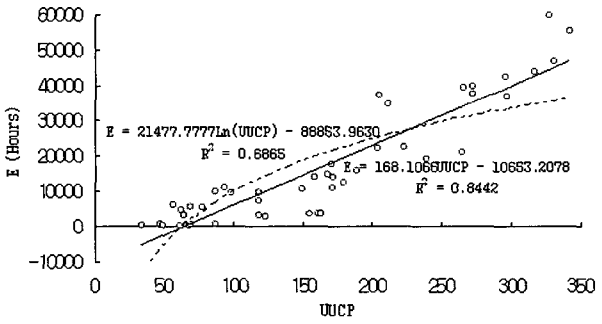
프로젝트		Actor (UAW)	Use Case (UUCW)	UUCP	AUCP	Staff-Hour per UCP	Effort
국내 개발 사례	P1	12	75	87	74.57	-	9,918
	P2	12	45	57	48.86	-	6,314
	P3	3	75	78	66.86	-	5,411
	P4	3	60	63	54.00	-	4,642
	P5	9	85	94	91.79	-	11,023
	P6	9	60	69	67.83	-	5,576
	P7	9	90	99	111.17	-	9,347
	P8	9	55	64	71.87	-	3,177
	P9	18	150	168	154.94	-	14,711
	P10	18	100	118	108.83	-	9,555
	P11	3	115	118	101.14	-	7,057
	P12	3	155	158	135.43	-	13,976
	P13	30	175	205	189.06	-	37,246
	P14	30	120	150	138.34	-	10,705
	P15	24	165	189	162.36	-	15,678
	P16	23	180	204	175.25	-	22,123
	P17	6	205	211	153.99	-	34,620
	P18	6	165	171	124.80	-	17,370
	P19	12	260	272	205.77	-	37,654
	P20	12	160	172	130.12	-	11,081
	P21	9	230	239	205.31	-	19,011
	P22	9	255	264	226.79	-	20,855
	P23	6	260	266	201.23	-	39,326
	P24	6	290	296	223.92	-	42,359
	P25	21	310	331	284.35	-	46,920
	P26	21	295	316	271.46	-	43,618
	P27	12	315	327	280.91	-	59,766
	P28	12	260	272	233.66	-	39,785
	P29	12	160	172	147.76	-	13,934
	P30	9	170	179	135.41	-	12,369
	P31	12	210	222	156.62	-	22,542
	P32	12	330	342	241.28	-	55,302
	P33	12	285	297	209.53	-	36,920

<표 1>과 <표 3>의 데이터를 종합하여 UUCP와 개발노력간 관계가 (그림 5)에 제시되어 있다.

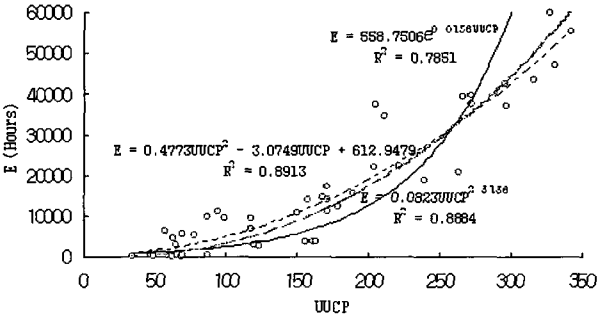


(그림 5) 종합 데이터의 UUCP와 개발노력 관계

본 데이터는 국내외의 유스케이스 점수에 기반한 개발노력 데이터를 종합한 것으로 소프트웨어 규모 (UUCP)가 (그림 4)보다 광범위하게 분포되어 있다. (그림 5) 데이터에 기존 연구 결과인 선형, 지수와 누승 모델이 적합할 것인가 아니면 다른 모델이 적합할 것인가가 관심의 초점이 되고 있다. 이들 모델에 대한 회귀분석 결과 실측 데이터와 모델의 추정 데이터는 (그림 6)에, 모델의 성능은 <표 4>에 제시하였다. 누승 모델이 MMRE 측면에서 볼 때 가장 적합한 결과를 나타내었으며, 누승 모델의 회귀분석 결과 분산분석표는 <표 5>에 제시되어 있다. 분산분석표로부터 $F \geq F(\alpha) = 53.0854 \geq 0.0000001$로 회귀분석이 유효함을 알 수 있다.



(a) 선형과 로그 모델 회귀분석 결과



(b) 누승, 다항식과 지수형 모델의 회귀분석 결과

(그림 6) 종합 데이터 회귀분석 결과 모델의 추정치와 실측치 비교

〈표 4〉 종합 데이터에 대한 개발노력 추정 모델 성능

모델 형태	모 델	결정계수	MMRE
선형	$E=168.1066UUCP-10653.2078$	0.8442	191.9276
지수	$E=558.7506e^{0.0156UUCP}$	0.7851	98.4394
누승	$E=0.0823UUCP^{2.3136}$	0.8884	71.3637
로그	$E=21477.7777\ln(UUCP)-88853.9630$	0.6865	379.1741
다항식	$E=0.4773UUCP^2-3.0749UUCP+612.9479$	0.8913	130.1043

〈표 5〉 누승 모델의 분산분석표

	자유도	제곱합	제곱평균	F	$F(a)$
회 귀	1	6104392676	6104392676	53.0854	0.000001
잔 차	29	3334768607	114992020		
계	30	9439161283			

(그림 6)으로부터 선형과 로그 모델은 MMRE가 가장 나쁜 결과를 보이고 있으면서 동시에 특정한 소프트웨어 규모에서는 개발노력이 "-" 값을 나타내는 모순을 보이고 있다. 즉, 선형 모델은 UUCP가 66부터, 로그 모델은 UUCP가 66이 되어야 개발노력이 "+" 값을 가진다. 지수모델은 소프트웨어 규모가 큰 부분에서는 실제 데이터를 잘 표현하지 못하는 단점을 보이고 있다. 반면에 다항식과 누승 모델은 실제 데이터를 비교적 잘 표현하는 성능을 보이고 있으며, 누승 모델의 MMRE가 가장 좋은 결과를 나타내고 있다.

3. 성장곡선을 이용한 개발노력 추정

본 장에서는 기존의 통계적 회귀모델들 보다 성능이 우수한 S자형 성장곡선을 제시하고자 한다.

3.1 성장곡선

현존하는 실세계의 많은 성장 현상은 초기에는 점진적으로 증가하다가 중간 구간에서 보다 빠르게 증가하고 한계점으로 도달하면서 성장 속도가 느려지며 일정한 시간 구간 후에는 최대 값에서 수평이 되는 S자 패턴을 보이고 있다. 따라서, 성장 곡선을 일명 S자(S-shaped) 곡선, 시그모이드(Sigmoidal) 곡선 또는 학습(Learning) 곡선이라 부르며 생명체의 성장이나 인간의 기술 성취도의 닮은 점을 표현하고 있다[15].

성장곡선을 표현하기 위해 다양한 수학적 함수가 제안되었다. 대표적인 성장곡선으로는 대칭 로지스틱 성장 곡선, 일반화된 로지스틱 곡선, Pearl 곡선과 Gompertz 곡선 등이 있다[15]. a를 성장함수 $y(t)$의 상한 점근선, t를 시간, b를 곡선의 위치를 결정하는 하한 점근선, c를 곡선의 모양을 결정하는 값인 성장률이라 하자.

대칭 로지스틱 성장곡선은 생물체의 성장 패턴을 모형화하는데 많이 사용되고 있으며 식 (3)으로 표현된다. 이 함수는 중간지점인 $a/2$를 중심으로 상호 대칭이 되며, $a/2$ 지점부터 성장률이 감소하기 시작하지만 최대 값 점근선인 a에 도달할 때까지 계속 성장한다.

$$y(t)=\frac{a}{(1+e^{b+ct})} \tag{3}$$

일반화된 로지스틱 곡선은 Richard 곡선이라고도 불리우며, 성장을 모형화하는데 널리 사용되고 있으며 식 (4)로 표현된다. 여기서 t_1는 최대 성장이 발생하는 점을 결정한다.

$$y(t)=b\frac{a}{(1+t_1e^{-c(t-M)})^{1/t_1}} \tag{4}$$

Pearl 곡선은 미국 통계학자인 Raymond Pearl[16]이 제안한 것으로 인구예측에 사용하였으며, 로지스틱 곡선으로 잘 알려져 있다. 표준화된 Pearl 곡선은 식 (5)의 형태를 취한다. 만약 초기 값이 0가 아닌 오프셋 값을 갖고 있으면 상수로서 위 공식에 더해진다. 이 곡선은 $t=\ln(b)/c$ 시점에서 $y(t)=a/2$가 중간지점이 되므로 이 지점을 중심으로 굴곡이 발생하며 위쪽과 아래쪽이 대칭이 된다.

$$y(t)=\frac{a}{1+be^{-ct}} \tag{5}$$

Gompertz 곡선은 영국의 공증인이자 수학자인 Benjamin Gompertz[17]에 의해 제안되었으며, 식 (6) 또는 (7)로 표현된다. 식 (6)은 식 (7)로 표현 될 수 있으며 식 (7)을 표준 Gompertz 성장곡선이라 불리운다. 이 곡선은 Pearl 곡선과 닮았지만, 곡선이 굴절되는 지점은 $t=\ln(b)/c$ 시점에서 비대칭이 되는 특징을 갖고 있다.

$$y(t)=ab^{c^t} \tag{6}$$

$$y(t)=ae^{-be^{ct}} \tag{7}$$

3.2 모델 선정과 모수 추정

성장곡선 들 중 대칭 로지스틱과 일반화된 로지스틱 곡선은 일반적으로 생명체 성장 분야에 적용하는데 비해 기술 예측 분야에는 Pearl 곡선과 Gompertz 곡선을 많이 적용한다. 본 논문의 주제는 기술예측분야에 속하므로 Pearl과 Gompertz 곡선에 한정하고자 한다. 이 두 곡선 중에서 적절한 것을 선택하는 기준은 다음 조건에 기반한다.

(1) 얻고자 하는 데이터의 성장 곡선이 대칭인가? 만약 대칭이 될 경우에는 Pearl 곡선을, 비대칭일 경우에는 Gompertz 곡선을 선택한다.

(2) 성장과정에서 이미 오프셋 (Offset) 요인이 있는가? 만약 오프셋요인이 있는 경우에는 Pearl 곡선을, 오프셋 요인이 없으면 Gompertz 곡선을 선택한다.

(그림 5)의 소프트웨어 규모에 따른 개발노력은 $y(t)=a/2$가 되는 시점에서 대칭이 되지 않는다. 왜냐하면 규모가 증가함에 따라 개발노력이 비선형적으로 증가하기 때문에 규모가 증가하는 후반부가 보다 크게 나타나기 때문이다. 또한 소프트웨어 규모가 0인 경우 개발노력은 0가 되기 때문에 오프셋 값이 존재하지 않는다. 따라서 성장곡선 중 Gompertz 성장곡선을 적용할 수 있다.

성장곡선을 이용하여 미래의 행위를 예측하는 과정은 성장 곡선 모델이 주어진 데이터에 적절한지 결정하고, 적절한 성장곡선 모델을 선택한 후 MSE(Mean Squared Error)를 최소화하거나 변수변환으로 얻는 선형회귀 방정식을 이용하여 모수(Parameter)를 결정한다. 식 (5)의Gompertz 성장곡선 모델들을 적용하기 위해 모수 a,b,c를 추정하는 방법은 최소자승법(LMS, Least Mean Squares)이나 최우추정법(MLE, Maximum Likelihood Estimation) 등 다양한 방법을 이용할 수 있다. 그러나 이 모델들은 비선형으로 초기 값을 적절히 설정하는데 어려움이 있다. 비선형 회귀 분석방법의 초기 값 선택은 쉬운 작업이 아니다[18]. 초기 값을 설정한 후 다시 Gauss-Newton 방법으로 모수를 조절하여 적절한 값을 얻을 수 있다. 이와 같은 이유로 인해 본 논문에서는 식 (6)의 Gompertz 성장곡선을 적용하지 않기로 한다.

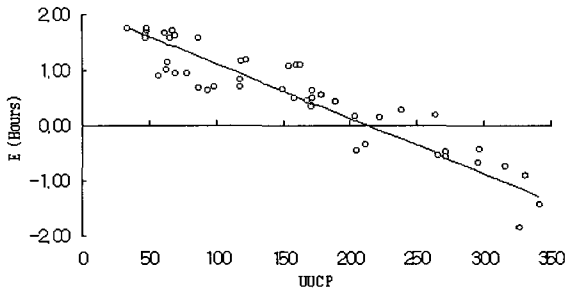
식 (7)의 표준 Gompertz 성장곡선은 식 (6)의 Gompertz 성장곡선 보다 간단한 방법으로 모수를 추정할 수 있는 장점을 갖고 있다. 식 (7)에서 시간 t를 소프트웨어 규모인 $UUCP$로, $y(t)$를 개발노력 E라 하자. 식 (8)과 같이 두 번의 자연로그 변수변환을 거치면 선형방정식이 얻어지며 이로부터 상용 통계 패키지 도구를 이용하여 선형회귀분석을 수행하면 모수들의 값을 쉽게 구할 수 있다.

$$\ln[-\ln(\frac{E}{a})]=\ln b-c\cdot UUCP \qquad (8)$$

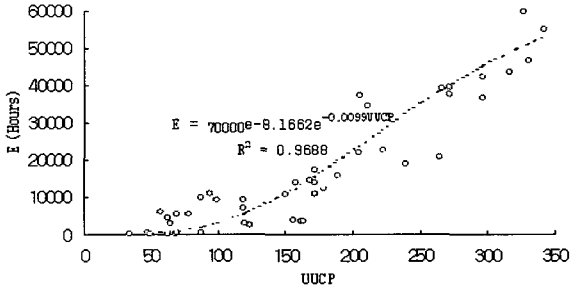
모수 a는 임의의 큰 값으로 초기값을 설정한 후 식 (8)의 좌변 식에 대한 주어진 데이터에 대해 그래프를 그리고 데이터들이 가능한 선형 형태를 취하도록 a의 값을 조절하면 된다. 모수 a의 값을 결정한 이후에는 모수 b,c는 통계 패키지를 이용한 선형회귀분석으로 얻을 수 있다. 결론적으로 본 논문에서는 개발노력을 추정함에 있어 식 (7)의 표준 Gompertz 성장곡선을 적용한다.

4. 적용결과 및 분석

<표 1>과 <표 3>에 대한 (그림 5)의 데이터에 대해 식 (8)에 따라 선형회귀분석을 수행한 결과는 (그림 7)에 제시되어 있다. $\ln(\hat{b})=2.1000$로부터 $\hat{b}=\log(\ln(\hat{b}))=8.1662$을, $\hat{a}=70000$, $\hat{c}=0.0099$을 얻었다. 이 모수들을 식 (7)에 적용하여 추정된 개발노력과 실측치를 비교한 결과는 (그림 8)에 제시되어 있다.



(그림 7) 표준 Gompertz 성장곡선을 적용한 모수 추정

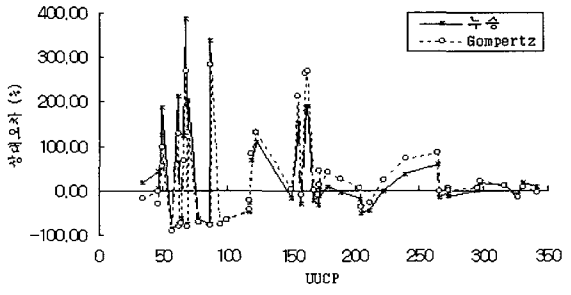


(그림 8) 표준 Gompertz 성장곡선 추정치와 실측치 비교

모델들의 상대오차는 (그림 9)에 제시되어 있다. 그림에서 소프트웨어 규모가 작은 부분 (UUCP < 100)에서는 누승 모델의 상대오차가 큰 반면 UUCP가 100 이상인 부분에서는 Gompertz 성장곡선 모델의 상대오차가 누승 모델보다 좋지 않은 결과를 나타내고 있다.

제안된 모델을 기존의 선형, 지수와 누승 모델들과 성능을 비교한 결과는 <표 6>에 제시되어 있다. 제안된 표준 Gompertz 성장곡선 모델은 소프트웨어 규모 전반에 걸친 개발노력을 잘 표현하는 능력과 더불어 모델의 성능(MMRE)도 가장 좋은 결과를 나타내고 있다.

따라서, 개발될 소프트웨어 시스템의 UUCP를 계산한 후, 제안된 모델을 이용하여 노력을 추정하고 개발업체의 시간당 평균 소요 노력을 곱하면 개발에 소요되는 직접노력을 구할 수 있을 것이다.



(그림 9) 누승과 Gompertz 성장곡선 모델의 상대오차

〈표 6〉 모델 성능 비교

모델 형태	모 델	결정계수	MMRE
선 형	$E=168.1066UUCP-10653.2078$	0.8442	191.9276
지 수	$E=558.7506e^{0.0156UUCP}$	0.7851	98.4394
누 승	$E=0.0823UUCP^{2.3136}$	0.8884	71.3637
로 그	$E=21477.7777\ln(UUCP)-88853.9630$	0.6865	379.1741
다항식	$E=0.4773UUCP^2-3.0749UUCP+612.9479$	0.8913	130.1043
Gompertz	$E=70000e^{-8.1662e^{-0.0001UUCP}}$	0.9688	66.3650

4. 결론 및 향후 연구과제

소프트웨어 개발에 적용되는 방법론과 언어는 보다 좋은 품질, 보다 빠른 개발과 보다 저렴한 개발노력을 투입하기 위한 방향으로 급격한 변화를 거듭하고 있다. 현재는 구조적 개발 방법론보다는 객체지향 방법론이 소프트웨어 산업계를 주도하고 있다. 소프트웨어 노력추정 분야도 이러한 변화에 적응하기 위해서는 기존의 구조적 개발방법론에 적합한 라인 수와 기능점수 기반의 추정모델을 적용하기보다는 객체지향 개발방법론에 적합한 유스케이스 점수 기반의 노력추정 모델로의 전환이 필요하다. 그러나 이 분야에 대한 연구가 거의 수행되지 않고 있다. 이러한 추세에 부응하고자 본 논문은 유스케이스 점수에 기반한 소프트웨어 노력추정 모델을 제시하였다.

본 논문에서는 유스케이스 점수 계산 방법을 살펴보고, 유스케이스 점수를 기반으로 하여 개발노력을 추정하는 기존의 선형과 비선형 모델의 문제점을 제기하였다. 다양한 실측 데이터를 획득하여 이에 적합한 모델을 고찰하였다. 다양한 성장곡선을 살펴보고 유스케이스 점수 와 소프트웨어 개발노력 관계를 적절히 표현할 수 있는 성장곡선을 선택하고, 모델의 모수를 추정하는 방법을 고찰하였다. 선택된 표준 Gompertz 성장곡선 모델은 유스케이스 점수와 개발노력 관계를 적절히 표현하는 능력과 더불어 상대오차도 기존 모델들 보다 작은 결과를 나타내 좋은 모델로 선택이 가능함을 보였다.

소프트웨어 개발 방법론은 전형적인 폭포수 모델에서 반복적이고 점진적인 단일화된 프로세스로 전환되었으며, 현재는 폭포수 모델과 단일화된 프로세스의 엄격한 절차 중심에서 탈피하여 기민한 프로세스(Agile Process)로 발전되고 있는 실정이다. 이에 따라 기술적 요구사항을 표현하는 형태도 유스케이스에서 스토리(Stories)나 형상(Features)들을 적용하고 있다. 이러한 발전 추세까지도 반영하여 모든 소프트웨어 규모 정량화 방법에 적용할 수 있는 일반화된 소프트웨어 개발노력 추정 모델을 제시할 필요가 있다. 추후에는 이 분야에 대한 연구가 수행될 것이다.

참 고 문 헌

[1] L. C. Briand, K. E. Elmam, D. Surmann, I. Wieczork, and K. D. Maxwell, "An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques," International Software Engineering Research Network, Technical Report, ISERN-98-27, 1998.

[2] L. C. Briand and I. Wieczorek, "Resource Estimation in Software Engineering," International Software Engineering Research Network, Technical Report, ISERN 00-05, 2000.

[3] K. Johnson, "Software Cost Estimation: Metrics and Models," Department of Computer Science University of Calgary, Albreta, Canada, http://sern.ucalgary.ca/courses/ seng/621/W98/johnsonk/cost.htm, 1998.

[4] K. Ribu, "Estimating Object-oriented Software Projects with Use Cases," University of Oslo Department of Informatics, Master of Science Thesis, 2001.

[5] 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.

[6] A. J. Albrecht, "Measuring Applications Development Productivity," Proceedings of IBM Application Dev., Joint SHARE/GUIDE Symposium, Monterey, CA., pp.83-92, 1979.

[7] I. Jacobson, M. Christerson, et al., "Object-oriented Software Engineering. A Use Case Driven Approach," Addison-Wesley, 1992.

[8] G. Karner, "Metrics for Objectory," Diploma Thesis, University of Linköping, Sweden, No. LiTH-IDA-Ex-934421, 1993.

[9] C. Larman, "Applying UML and Patterns. An Introduction to Object-oriented Analysis and Design and the Unified Process," Prentice-Hall, 2002.

[10] G. Banerjee, "Use Case Points-An Estimation Approach," http://java.isawix.com/whitepapers/1035194512861.pdf, 2001.

[11] L. Probasco, "Dear Dr. Use Case: What About Function Points and Use Cases?," http://www.therationaledge.com/content/aug_02/t_drUseCase_lp.jsp,Rational Software Canada, 2002.

[12] 박주석, 정기원, "소프트웨어 개발비용을 추정하기 위한 사용사례 기반 모델," 정보처리학회논문지D, 제11-D권 제1호, pp.163-172, 2004.

[13] S. Nageswaren, "Test Effort Estimation Using Use Case Points," Quality Week 2001, San Francisco, California, USA, 2001.

[14] A. Abran, C. Symons, and S. Oligny, "An Overview of COSMIC-FFP Field Trial Results," ESCOM 2001, London, England, 2001.

[15] C. Henry, "The Growth Curve," http://www.anzpug.org/jsp/index.jsp, PRIMAVERA Users Groups, Technology and Operations Management, California Polytechnic

and State University.
[16] R. Pearl, "The Biology of Population Growth," New York: Knopf, 1978.
[17] B. Gompertz, "On The Nature of The Function Expressive of The Law of Human Mortality, and on a New Mode of Determining the Value of Life Contingencies," Phil. Trans. Roy. Soc. London. Vol,123, pp.513-585, 1832.
[18] Weibull.com, "Software Reliability Growth Model," http://www.weibull.com/relgrowthwebcontents.html

박 주 석
e-mail : iage2k@dreamwiz.com
1984년 해군사관학교 전자공학과(공학사)
1995년 국방대학원 전산계산학과 (전산학석사)
2004년 숭실대학교 컴퓨터학과(공학박사)
2001년~현재 국방대학교 직무연수부 교수
관심분야 : 비용산정, 표준 및 프로세스, 정보체계 사업관리, 소프트웨어품질보증, 정보전