韓國 OA學會 論文誌
第4卷 第2號, 1999. 6.

99-4-2-1-7

소프트웨어유지보수 프로젝트의 투입인력 규모예측 모형

안 연 식*

An Effort Estimation Model for Software Maintenance Project

Yeon Sik Ahn*

요 약

본 연구에서는 소프트웨어 유지보수에 영향을 주는 생산성 요소들을 투입인력 특성, 소프트웨어의 기술적 특성 및 유지보수 환경특성 등 3개의 영역으로 구분하였다. 또한 실제 유지보수 프로젝트의 데이터를 수집하여 소프트웨어 유지보수 규모와 생산성 요소를 결합한 투입인력 규모예측을 위한 다양한 회귀모형을 통계적 분석에 의해 검증하였다. 결론적으로 소프트웨어유지보수 프로젝트의 투입인력 규모예측을 위해 가장 유의한 모형으로 요인점수에 의한 선형모형이 최종 모형으로 제시되었다.

Abstract

In this study, the productivity factors of software maintenance are categorized into the three areas such as maintainer characteristics, software technical characteristics and maintenance environment characteristics. And the various regression analyses are performed for searching the most significant model by using actual project's data. At conclusion, an linear model including new variables derived from factor analysis to estimate the effort for software maintenance project is suggested.

* 경원전문대학 비서과 조교수
논문접수 : 1999. 5.25. 심사완료 : 1999. 6.21

Ⅰ. 서론

대부분의 조직에서 소프트웨어 유지보수에 소요되는 비용이 증가하고 있으며, 유지보수에 투입되는 인력의 규모가 누증되고 있다. 따라서 이를 효율적으로 관리해야 하는 조직과 특히 프로젝트관리자들에게 정확도와 활용성이 높은 유지보수 비용예측 모형이 필요하다 하겠다.

그러나 대부분의 관련 연구는 신규 소프트웨어개발 프로젝트의 비용예측에 비해 소프트웨어의 유지보수부문에는 다소 소홀히 되어 왔다. 따라서 본 연구에서는 관련 연구를 분석하여 소프트웨어 유지보수의 특성을 포함한 생산성요소들을 도출하고, 이를 유지보수 프로젝트에 투입되는 인력의 규모예측에 활용할 수 있는 모형을 제시하고자 한다.

Ⅱ. 관련 연구

본 장에서는 소프트웨어 유지보수에 영향을 미치는 생산성 요소에 관한 기존의 연구결과를 분석하고, IFPUG에서 제시한 최신의 기능점수모형에서의 유지보수 프로젝트 소요공수 예측모형에 관해 고찰한다.

1. 유지보수 생산성 요소

Sommerville[14]은 유지보수 생산성 관련 요소로서 비기술적 요소 5개 항목과 6개 항목의 기술적 요소를 도출하여 〈표 1〉과 같이 제시하였다.

표 40. 유지보수 생산성 요소
Table 1. Maintenance Productivity Factors

비기술적 요인	기술적 요인
애플리케이션 영역 요원의 안정성 프로그램의 나이 외부 환경 하드웨어 안정성	모듈의 독립성 프로그램 언어 프로그램 스타일 프로그램 검증 문서화 형상관리

Jorgensen의 연구[6]는 유지보수 원인, 변경유형, 모듈형태와 자신감 등은 유지보수에 영향을 미치며 프로그램언어, 유지보수자 경험, 과업의 우선순위, 프로그램 나이, 전체시스템의 크기 등은 생산성과 관련이 없다는 결과를 밝히고 있다.

유성렬의 연구[17]에서는 노력보정계수로서 유지보수자의 경력, 경험, 프로그램언어, 대상기종 및 운영체제 등 4개 항목을 유지보수자의 속성으로, 파일크기, 소프트웨어의 복잡도, 신뢰도, 처리유형 등 4개 항목을 소프트웨어의 속성으로, 일정, 도구사용, 개발유형, 유지보수 조직 및 업무의 중복성 등 4개 항목을 시스템의 환경속성으로 분류하고 있다. Oman의 연구[10]에서는 요원과 프로세스의 관리, 기존 환경과 변환환경, 그리고 성숙도와 원시코드 및 문서화를 고려하고 있다.

또한 Jones[5]는 COBOL과 같은 프로그램언어를 적용하고 구조성이 우수한 소프트웨어에서 유지보수프로그래머 1인당 1500기능점수라는 생산성을 나타낸데 비해서, 오래된 소프트웨어로서 구조성과 문서화수준이 낙후된 경우에는 500기능점수의 생산성으로서 약 3배의 차이를 보여주는 결과를 발표했다. 마찬가지로 구조성과 문서화수준이 떨어지는 소프트웨어(COBOL언어)에서 1KLOC의 추가하는데도 3배까지의 생산성 저하를 보였으며, 이것은 1KLOC의 새로운 프로그램 작성시보다 2배이상의 시간을 소요한다고 하여 결과적으로 모듈의 구조성과 문서화수준을 중요한 요소로 지적하였다.

Vessay와 Weber[16]는 약 400본의 COBOL 프로그램을 대상으로 한 연구에서 프로그램의 복잡도와 프로그램의 스타일을 교정유지보수(corrective maintenance)에 유의한 요인임을 밝혔다.

위의 여러 연구결과에서는 서로 상충되는 부분도 적지 않으며, 대부분의 연구는 생산성 요소를 실제 데이터를 이용하여 검증을 실시하지 않았거나 포함하고 있더라도 프로그램의 원시코드수를 기준으로 한 것이 약점이라 하겠다.

2. 기능점수모형과 유지보수 소요공수 예측

IFPUG의 최신 기능점수모형[4]에서는 소프트웨어 개선(enhancement)프로젝트의 규모를 다음 (식1)에 의해 기능점수로서 예측한다. 기능점수는 사용자의 관점에서 파악하는 외부입력, 외부출력, 외부조회, 내부파일과 외부연계파일 등 5개 기능유형별 기능수에 각각의 난이도에 의한 가중치를 곱한 점수로서 표현되는 값이다.

$$EFP = [(FP_A + FP_{CH} + FP_{CV}) * VAF_A] + (FP_D * VAF_B) \ldots\ldots (식1)$$

여기서 EFP=개선프로젝트 자체의 기능점수, FPA=개선프로젝트에 의해 추가된 기능점수, FPCH=개선프로젝트에 의해 수정된(modified) 보정되기 전의 기능점수, FPCV=변환(conversion)에 의해 추가된 기능점수, FPD=개선프로젝트에 의해 삭제된 기능점수를 각각 나타낸다.

또한 기능점수모형에서는 데이터통신의 필요정도, 분산처리, 성능, 특수하게 고려되는 운영환경, 처리율, 온라인 데이터 입력, 최종사용자 효율성, 온라인 갱신, 복합처리, 재사용성, 설치 용이성 그리고 운영의 용이성 항목 등 총 14개 항목의 기술적 복잡도(technical complexity)에 의한 보정을 적용하는데 VAFA=개선프로젝트 이후에 애플리케이션의 보정계수, VAFB=개선프로젝트 이전의 애플리케이션의 보정계수로 정의되며, 각 항목당 0~5의 가중치를 계산하여 모두 합한 값인 총영향도(Total Degree of Influence)를 다음 (식2)에 의해 보정계수로 환산한다.

$$VAF = (TDI * 0.01) + 0.65 \ldots\ldots\ldots (식2)$$

그러나 이 모형에서는 기능점수를 보정하는 기술적 복잡도를 개발프로젝트와 동일시하고 있어서, 소프트웨어 유지보수의 생산성과 관련된 복잡도 항목으로의 대체 가능성에 대한 연구가 필요하다 하겠다.

Ⅲ. 모형의 설정 및 검증방법

본 연구에서는 앞에서 고찰한 소프트웨어 유지보수에 관련된 생산성 요소에 관한 연구결과를 바탕으로 생산성 요소의 후보를 고려하고, 유지보수 규모의 예측은 기능점수모형을 이용함으로써, 유지보수 프로젝트에 소요되는 투입인력의 예측을 위해 필요한 최적의 모형을 제시한다. 또한 설문조사를 통해 수집된 실제 유지보수 프로젝트 데이터를 수집하여 통계적 검증을 통한 최적의 모형을 도출하기로 한다.

1. 소프트웨어 유지보수의 생산성 요소

본 연구에서는 소프트웨어 유지보수의 생산성 요소로서 다음과 같은 10개 항목을 선정하였다.

① 적용(application) 업무에 대한 실무지식

소프트웨어의 유지보수는 소프트웨어의 개발과 마찬가지로 유지보수 대상업무에 대한 요구정의 및 시스템 분석을 필요로 하며, 이 과정에서 유지보수에 투입되는 인력이 적용업무에 대한 실무지식의 보유정도에 따라서 유지보수 생산성에 영향을 미칠 것이다.

② 해당프로그램 언어에 대한 지식

소프트웨어 유지보수에서는 프로그램을 수정, 삭제, 추가 등의 작업이 발생하게 되며 이때 해당 프로그램 언어(program language)에 대한 지식의 정도에 따라서 유지보수 생산성에 영향을 미칠 것이다.

③ 해당 소프트웨어의 시스템환경 지식

소프트웨어의 유지보수에서 수정된 프로그램의 설치 및 시험과 관련하여, 프로그램이 동작하는 플랫폼 즉

운영체제(OS)나 데이터베이스관리시스템(DBMS)에 대한 지식의 정도에 따라 유지보수 생산성에 영향을 미칠 것이다.

④ 모듈의 구조성(structuredness)

모듈이 개발당시의 구조화 상태를 유지하고 있는 경우에는 모듈의 가독성(understandability)과 수정용이성이 우수하여 생산성에 좋은 영향을 보이는 반면에 프로그램의 나이가 듬에 따라서 구조성이 저하되고 이것이 유지보수 생산성의 저하를 가져올 것이다.

⑤ 모듈의 독립성

소프트웨어는 다수의 모듈로 구성되며 모듈들은 개발완료시점에는 서로 높은 독립성을 유지하고 있을 것이지만, 유지보수가 지속됨에 따라 최초의 구조성이 점점 저하될 것이다. 따라서 서로 얽히고 설키는 상태가 됨으로서 독립성이 점점 저하될 것이며, 한 모듈의 수정시 파급효과가 커짐으로써 유지보수 생산성의 저하를 가져올 것이다. 따라서 모듈의 독립성 정도에 따라서 유지보수 생산성에 영향을 미칠 것이다.

⑥ 프로그램언어의 종류

소프트웨어는 이를 구현하는 프로그램 언어에 따라서 복잡도가 다르다. 즉 제3세대언어로부터 제4세대언어, 객체지향언어 등 프로그램언어의 종류에 따라서 구현의 난이도가 다르므로 유지보수 생산성에 영향을 미칠 것이다.

⑦ 프로그램의 재사용성

소프트웨어의 유지보수 과정에서 기존 프로그램모듈을 재사용하는 경우가 있다. 이처럼 기존 모듈을 재사용하는 정도에 따라서 유지보수 생산성에 영향을 미칠 것이다.다.

⑧ 문서화(documentation) 상태

소프트웨어를 설명하는 문서가 최신의 상태로 유지되어 소프트웨어 유지보수에서 이를 활용하면 유지보수 생산성을 높일 수가 있을 것이다. 또한 문서화 상태가 유지보수에 전혀 활용되지 못하면 그 만큼 생산성의 저하를 가져올 것임으로, 문서화 상태에 따라서 유지보수 생산성에 영향을 미칠 것이다.

⑨ 자원의 표준화준수 상태

시스템 자원(table 명, field명, module명 등)이 각종 표준화 규칙을 준수하고 있는 경우에는 시스템상황의 파악이 용이하고 유지보수 작업에 효율적일 것이다. 반대로 무질서한 자원관리 상태에서는 생산성이 저하될 것이므로 이러한 자원의 표준화준수 정도가 소프트웨어 유지보수에서 생산성에 영향을 미칠 것이다.

⑩ 시험/설치의 용이성

소프트웨어 유지보수에서 수정완료된 소프트웨어를 설치하고 시험하기에 용이한 환경에서는 더 높은 유지보수 생산성을 보일 것이다. 반대로 여러 플랫폼에서 다양한 사례에 대해 설치 및 시험이 필요한 경우에는 생산성이 저하될 것이다. 따라서 설치 및 시험의 용이성 정도에 따라서 유지보수 생산성에 영향을 미칠 것이다.

표 41. 응답 프로젝트의 일반사항
Table 2. General Descriptions of Responded Project' Data

구분	평균치	단위
시스템의 규모	169.1	본수
프로그램 언어	3(4GL), 20(3GL)	
시스템 평균수명	6.24	년
프로젝트당 유지보수 규모	37.7	기능점수
프로젝트당 유지보수 투입인력	9.5	man · weeks

2. 투입인력 예측 모형

본 연구에서 선정한 생산성 요소를 이용하여 유지보수 프로젝트의 투입인력을 예측하기 위한 모형을 회귀분석을 통해 검증한다. 또한 유지보수 대상 소프트웨어의 규모를 나타내는 척도로는 기능점수(function Point)를 기준으로 한다. 소프트웨어 개발 및 유지보수 규모와 생산성 요소간의 관계를 회귀모형에 의하여 검증(regression analysis)할 때, 많은 연구에서 다음의 (식3)에서와 같이 투입인력을 종속변수로 하고 기타의 변수를 독립변수로 하는 비선형의 관계식을 활

용하고 있다.

$$\text{투입인력} = a * \text{기능점수}^{\text{생산성지수}} \quad \ldots\ldots(\text{식}3)$$

본 연구에서는 선형 및 비선형의 다양한 회귀식을 대상으로 통계적 검증을 수행하여 최적의 회귀모형을 도출하고자 한다.

3. 설문의 내용

위의 모형의 검증을 위해 사용할 설문은 다음 내용을 포함하도록 구성하였다. 즉 유지보수 대상시스템의 명칭, 규모, 개발언어, 초기버젼 개발완료시점 등의 일반적 내용과 유지보수 프로젝트의 소프트웨어의 기술적 특성상태, 유지보수자원환경상태, 유지보수 인력의 상태 등을 5점 리커드 척도로 응답토록 하였다. 또한 유지보수 프로젝트에 투입된 인력을 산정하기 위한 기술자 등급별 규모와 유지보수 규모를 기능점수에 의해 산정하기 위한 작업의 기능별 제목, 작업구분별 평가항목 등이다.

Ⅳ. 연구결과

1. 설문조사의 개요

설문대상 기관으로는 국영기업의 SI전담 자회사 1개 업체와 국내 대기업계열 SI자회사 2개 업체 등 총 3개 회사를 선정하였으며, 소프트웨어 유지보수를 주요 업무로 하고있는 SM(System Management)부서의 중간관리자 및 프로젝트관리자(PM)급을 대상으로 설문의 취지와 내용을 설명하고 설문지를 작성하도록 하였다. 설문의 작성에 노력이 소요되기 때문에 약 1주일의 시간을 주고 난 뒤 이를 회수하였으며 일부의 상세한 내용은 직접 면담 및 전화를 통해 확인하였다. 소프트웨어 유지보수 프로젝트 1건에 1매를 기준으로 총 30매의 설문이 배포되어 28매가 회수되었으며, 불성실한 설문 등이 제외된 후 총 23매의 설문이 본 연구에 활용되었다. 또한 설문결과의 처리는 SPSS(version 8.0)을 사용하였다. 이들 23개 프로젝트에서 수집된 데이터에 대한 일반적 특성은 다음 〈표 2〉와 같다.

2. 유지보수 생산성요소에 대한 검증

1.1.1. 생산성요소의 신뢰도

본 연구에서 선정한 10개 항목의 소프트웨어의 유지보수 생산성 요소를 대상으로 한 응답치의 신뢰도 검정결과, 다항목의 신뢰도를 나타내는 크론바하 알파값이 0.8253으로서 통상적인 기준치인 0.6이상을 보여주고 있어, 신뢰성이 있는 것으로 분석된다.

또한 10개 항목을 연구자의 분류방식에 의해 3개 영역 즉 유지보수 소프트웨어의 기술적 특성, 투입인력의 특성 및 유지보수 자원환경 특성영역으로 분류한 상태에서 수집된 데이터를 이용하여 신뢰성 및 타당도 분석을 실시한 결과, 크론바하 알파값이 4개 항목의 소프트웨어의 기술적 특성(α=0.8816), 3개 항목의 투입인력 특성(α=0.8671), 3개 항목의 유지보수환경특성(α=0.6312) 등 세가지 영역에서 각각 0.6 이상의 비교적 큰 값으로 나타나서 신뢰성이 있는 것으로 분석되었다. 그러나 특이한 점은 투입인력이 해당 응용업무에 관련된 「실무지식」 항목과 유지보수환경 특성에서 「시험/설치의 용이성」 항목은 소프트웨어 유지보수의 생산성과 관련이 높을 것으로 고려하였으나, 이 2개 항목을 제거하는 것이 오히려 크론바하 알파값을 0.8671에서 0.9601로 그리고 0.6312에서 0.6856으로 각각 증가시키는 것으로 나타났다. 따라서 이에 대해서는 상세한 연구분석이 필요하겠지만 유지보수 시점에서는 이미 업무에 대한 지식이 크게 영향을 미치지 않는 것으로, 그리고 이미 개발환경에서 소프트웨어가 설치되어 있는 상태로서 시험/설치에는 큰 노력이 필요하지 않는다는 것을 암시하는 것으로 받아들여야 할 것 같다.

이와 같은 결과에 따라 본 연구에서는 앞에서 제시한 유지보수 생산성 요소 10개 항목 중에서 투입인력

표 42. 생산성요소의 통계분석 결과
Table 3. Statistic Analysis Result for Productivity Factors

영역	생산성 요소	요인부하량
투입 인력의 특성 (α=0.9601)	① 프로그램 언어에 대한 지식 ② 소프트웨어의 시스템환경에 대한 지식	0.976 0.953
소프트웨어의 기술적 특성 (α=0.8816)	① 모듈의 구조성 ② 모듈의 독립성 ③ 프로그램언어의 종류 ④ 프로그램의 재사용성	0.831 0.807 0.884 0.837
유지보수자원환경 특성 (α=0.6856)	① 문서화 상태 ② 자원의 표준화준수 상태	0.856 0.812

표 43. 보정후 기능점수에 의한 회귀분석 결과
Table 4. Regression Analysis Result using Adjusted Function Point

번호	보정율	모형	결정계수	a	b	회귀식
1	±0%	$FP = FPu$	0.815%	0.062	1.352	$M = 0.062 * FP^{1.352}$
2	±10%	$FP = FPu*[0.9 + (0.005 * TDI)]$	0.822%	0.070	1.314	$M = 0.070 * FP^{1.314}$
3	±20%	$FP = FPu*[0.8 + (0.001 * TDI)]$	0.826%	0.078	1.276	$M = 0.078 * FP^{1.276}$
4	±30%	$FP = FPu*[0.7 + (0.015 * TDI)]$	0.829%	0.086	1.247	$M = 0.086 * FP^{1.247}$
5	±40%	$FP = FPu*[0.6 + (0.002 * TDI)]$	0.829%	0.094	1.218	$M = 0.094 * FP^{1.218}$

의 특성 중 적용업무에 대한 실무지식항목과 유지보수 환경특성 중 시험/설치용이성을 제외하고 총 8개 항목만을 사용하기로 한다.

1.1.2. 생산성 요소의 분류

본 연구에서 사용할 8개 항목의 생산성 요소들을 응답치 특성에 의하여 통계적으로 항목의 요약 및 축소를 위해 요인분석을 실시하였다. 요인분석에서는 주성분분석법(principle components method)을 사용하였고, 요인수의 결정은 고유치(eigen value)가 1이상인 것을 선택하는 것을 원칙으로 하였으며, 요인을 회전시키는 방법은 직교회전법(varimax)을 사용하였다.

그 결과 〈표 3〉에서와 같이 8개 항목이 4개 항목의 소프트웨어 기술적 특성과, 2개 항목의 투입인력 특성 그리고 2개 항목의 유지보수 자원환경특성으로 적재되었다. 이와 같은 분석결과를 정리하여 표시하면 〈표 3〉과 같다.

3. 유지보수 규모와 생산성 요소의 관계

본 연구에서는 유지보수 프로젝트에 대한 설문조사를 바탕으로 생산성 항목 응답치, 실투입인력 그리고 유지보수 규모를 기능점수로 산정한 결과 등의 실제 프로젝트 데이터를 이용하여 생산성 예측을 위한 모형을 도출해보기로 한다.

대개의 경우 소프트웨어 유지보수 규모와 생산성 요소간의 관계를 회귀모형에 의하여 검증(regression analysis)할 때, 많은 연구에서 다음과 같이 투입인력을 종속변수로 하고 기타의 독립변수가 선형 또는 비선형의 관계식을 활용하고 있다. 물론 최적의 모형으로는 통상 F값을 통해 통계적 유의성이 검증된 전제에서 회귀분석의 결정계수(R2)가 큰 모형이 선정된다. 본 연구에서는 다음과 같이 선형모형, 지수모형 등 몇 가지의 가능한 모형을 대상으로 최적의 모형을 도출하기로 한다.

(1) 단순 선형모형

단순 선형모형은 실투입인력(M)이 보정전 기능점수(FP)와의 1차 선형관계를 가정한 모형이다. 회귀분석에서 실투입인력과 기능점수와의 관계가 다음 (식4)에서와 같이 상수 a, b는 -3.766, 0.349 등으로 산출

되었다.

$$M = a + b * FP$$
$$= -3.766 + 0.349 * FP \text{.....(식4)}$$

이때 결정계수(R2)는 0.786으로 산출되었으며, 따라서 이 결과에 의하면 기능점수가 투입인력의 78.6%를 설명하고 있다. 그러나 이 모형에서는 기능점수 외에 다른 생산성요소를 회귀식에서 제외하고 있다.

(2) 요인점수에 의한 선형모형

이 모형은 투입인력을 기능점수 및 요인분석의 결과로 생성되는 새로운 3개 요인점수(factor score) 항목들이 결합된 1차 선형관계를 가정한 모형이다.

투입인력=a + b*기능점수 + c*X1 + d*X2 + e*X3 .....(식5)

여기서 X1, X2, X3는 요인분석에서 선택기능에 의해 새로이 생성된 3개 요인점수 항목으로서 각각 소프트웨어의 기술적 특성 요인, 투입인력 특성 요인, 유지보수 자원환경 특성 요인이다.

회귀분석은 단계적 변수 투입방식(stepwise)으로 처리한 결과 유의수준이 낮은 제2요인점수 항목은 모형에서 제외되었고, 각 계수를 대입하여 다음 (식6)과 같이 표현할 수 있다. 또한 이 모형에 의해 설명되는 투입인력의 변동을 나타내는 결정계수(R2)는 0.877으로 산출되었다.

투입인력= -4.912 + 0.379*기능점수 + 2.488*X1 -2.117*X3.....(식6)

이 모형을 단순 선형모형과 비교해볼 때, 생산성 요인을 추가함으로써 약 9.1%의 설명력이 증가된 결과로 해석할 수 있다.

1.1.3. 기능점수에 의한 지수모형

이 모형은 보정전 및 보정후의 기능점수와 투입인력의 관계를 지수모형으로 표현한 것이다. 회귀분석은 Levenberg-Marquardt 방법을 사용하여 처리하였다.

우선 보정전의 기능점수를 회귀식에 대입하여 처리한 결과 상수 a, b는 0.062, 1.352로 각각 산출되어 회귀식은 다음 (식7)과 같이 표현될 수 있다. 또한 다음 표에서 보는 바와 같이 이 경우의 결정계수(R2)는 0.815로 산출되었다.

$$M = a * FP^{**}b$$
$$= 0.062*FP^{**}1.352 \text{.....(식7)}$$

이어서 〈표 4〉에서 보는 바와 같이 위의 보정전 기능점수(FPu)를 생산성요소에 대한 응답치의 합(TDI; Total Degree of Influence)를 고려하여 ±10%~±40%의 범위에서 보정한 후 새로운 기능점수(FP)로서 산출하고, 이 보정후 기능점수와 투입인력의 관계도 같은 방식으로 회귀분석에 의하여 처리하였다.

이와 같은 방식에 의해서 각 보정율의 범위에 따라 보정후 기능점수가 달라지고, 그에 따라 실시된 회귀분석의 결정계수도 〈표 4〉에서 보는 바와 같이 달라져서 0.815 ~ 0.829까지 변동되는 것을 알 수 있다.

V. 결론 및 토의

소프트웨어의 유지보수 비용예측을 위해 본 연구에서 다양한 회귀모형을 이용한 분석을 수행한 결과 F값을 통해 통계적 유의성이 있는 모형 중에서 결정계수(R2)가 가장 높은 모형으로 요인점수에 의한 선형모형 즉 투입인력을 기능점수 및 요인분석의 결과로 생성되는 새로운 3개 요인점수 항목이 결합된 선형관계 모형이 최적의 모형으로 선정되었다.

본 연구에서 제시된 모형은 소프트웨어개발 프로젝트에 적용되는 기능점수 모형의 확장과 개선을 통해 설정되었기 때문에 기능점수 모형의 과학성과 합리성을 그대로 유지하고 있다고 하겠다. 따라서 본 연구에서 제시된 모형의 장점을 들면 다음과 같다.

첫째, 소프트웨어의 규모 측면에서는 소프트웨어가

실행하는 기능을 나타내는 기능점수와 직접적인 관계를 갖는다는 점을 인정하고 있다. 따라서 소프트웨어의 규모를 측정할 때 스텝수를 적용하는 비과학성을 배제하고 있다. 또한 소프트웨어의 유지보수 규모를 예측할 때 과거의 유지보수 실적을 활용하는 방법도 있을 수는 있으나 유지보수 규모에 대한 과거 실적이 없거나 있더라도 신뢰성있는 자료가 수집되어 있지 않을 수도 있으며, 유지보수 규모가 과년도와 비슷하게 발생되지 않는 경우에 적용되는 불합리한 점을 배제할 수 있는 장점을 가진다.

둘째, 기능점수모형에서 제시하고 있는 기술적 복잡도를 평가하는 14개 항목자체의 문제, 평가의 객관성 확보 문제를 극복한 모형이다. 본 연구에서 제시한 모형에서는 이를 간단하게 평가하도록 하여 유용성을 높이면서도 평가에서 객관성을 높이는 방법으로 개선한 것이 특징이다.

셋째, 기능점수 모형은 IBM의 Albrecht를 중심으로 하여 소프트웨어 개발프로젝트에 적용하는 것을 전제로 개발된 모형으로서 국내 여건상 소프트웨어의 규모예측이나 예산확보가 시스템의 분석단계에 시행되는 것이 아니라 훨씬 앞선 단계인 시스템의 기본 개념설정상태에서 이루어지기 때문에 기능점수모형이 널리 정착되지 못하고 있다. 이러한 적용상의 난제가 있음에도 불구하고 소프트웨어의 유지보수 프로젝트는 이미 개발된 실체로서 소프트웨어가 존재하고 있다는 점에서 기능점수의 파악이 용이하고 특히 소프트웨어 유지보수의 규모도 소프트웨어의 기능점수차원에서 접근하는 것이 어렵지 않기 때문에 본 연구에서 제시된 모형의 활용성이 높을 것으로 기대된다.

결론적으로 본 연구에서는 유지보수 대상 소프트웨어의 규모예측은 기능점수를 이용하고 소프트웨어 유지보수에서 유의한 생산성요소들을 도출하여 이를 생산성 예측에 활용할 수 있는 모형을 제시하였다. 이 모형은 소프트웨어 유지보수 프로젝트의 극히 일부 건수만을 대상으로 검증되었기 때문에 조직의 경험데이터를 이용한 검증 결과와 반드시 일치한다고 할 수는 없으며, 최소한 본 연구에서의 접근방식은 어느 조직에서도 활용될 수 있다고 판단된다. 향후 소프트웨어 유지보수의 특성을 고려한 생산성 요소에 관한 심도있는 연구와 실증데이터를 이용한 검증을 통해 다양한 소프트웨어 유지보수 생산성예측 모형이 제시되어야 할 것이다.

참고문헌

[1] Rajiv D. Banker, Srikant M. Datar, Chris F. Kemerer, "Factors Affecting Software Maintenance Productivity: An Exploratory Study," Proceedings of Information System, 1987. pp. 160-175

[2] L. Belady and M. Lehman, "An introduction to Growth Dynamics," in Statistical Computer performance Evaluation(W. Freiberger, ed.), Academic Press, 1972, pp. 503-511

[3] "Guidelines on Software Maintenance," Federal Information Processing Standards Publication(FIPS PUB 106), 1984

[4] "Function Point Counting Practices Manual (Release 4.0): Enhancement Project Function Point Calculation," IFPUG, 1994, pp. 8.10-8.17

[5] James Jones, "Applied Software measurement(2nd Ed.)," McGraw-Hill, 1997, pp. 95-96

[6] Magne Jorgensen, "Exprience With the Accuracy of Software Maintenance Task Effort Prediction Models," IEEE Transactions on Software Engineering, Vol. 21, No. 8, pp. 674-681

[7] Chris F. Kemerer, "An Empirical Validation of Software Cost Estimation Models," Communications of the ACM, Vol. 30, NO. 5(1987. 5) pp. 416-429

[8] B. P. Lintz, E. B. Swanson, "Characteristics of Application Software Maintenance," Communications of the ACM,

Vol. 21, NO. 6, pp. 466-471
[9] J. Martin & C. McClure, "Software Maintenance: The Problem and Its Solution," PH, 1983
[10] P. Oman, J. Hagemeister & D. Ash, "91-08 TR: A Definition and Taxonomy for Software Maintainability for Software Maintainability," Univ. of Idaho, SE Test Lab TR, 1991
[11] Thomas M. Pigoski, "Practical Software Maintenance," wiley, 1997
[12] Pressman, "Software Engineering: a Practitioner's Approach(3rd Ed)," McGraw-Hill, 1992, pp. 667-669
[13] H. Schaefer, "Metrics for Optimal maintenance management," Proc. Conf. Software Maintenance-1985, IEEE, November 1985, pp. 114-119
[14] Ian sommerville, "Software Engineering," addison-wesley, 1996, pp. 666-672
[15] George E. Stark, "Measurement to manage Software Maintenance," MITRE Corp, 1997, http://stsc.hill.af.mill/crostolk/1997/jul/maintenance.html
[16] I. Vessay, R. Weber, "The Dimensions of Maintainance," Proc. 2nd Int'l Conf. Software Engineering, IEEE, Oct. 1976, pp. 492-497
[17] 유성렬, "소프트웨어 유지보수를 위한 시스템 설계전략과 비용예측 모델에 관한 연구," 아주대학교 대학원 박사학위 논문, 96. 8

저 자 소 개

안 연 식

1977 - 1982 전북대학교 자연과학대학(이학사)
1987 - 1990 연세대학교 산업대학원 전자계산 전공(공학석사)
1997 - 현재 국민대학교 대학원 정보관리학과 재학중(박사과정)
1981 - 1992 한국전력공사 정보처리처 과장
1992 - 1996 한전정보네트웍(주) 컨설팅사업부장
1991 - 1996 동서울대학 전자계산과 강사
1996 - 현재 경원전문대학 비서과 조교수(정보처리기술사)