소프트웨어 규모산정을 위한 기능점수 측정 개선사례

박종모*, 김승권**
정보통신산업진흥원 소프트웨어공학센터
e-mail: jmpark@nipa.kr, sgkim@nipa.kr

A Case study of Improved Function Point Measurement for Software Size Estimation

Jongmo Park*, SeungGweon Kim**
Software Engineering Center, National IT Industry Promotion Agency

요 약

소프트웨어 규모산정 활동은 기획, 구현, 운영 등 소프트웨어 수명주기 동안 수행되는 활동으로 프로젝트를 추진함에 있어 예산수립, 사업발주, 사업관리 등에 중요하다. 소프트웨어 규모산정으로 사용되는 기능점수를 측정할 때 개인의 주관적인 판단으로 인해 오차가 발생한다. 본 논문에서는 기능점수 측정의 오차를 줄이기 위해 4가지의 검증로직을 제시하고, 진행되는 실제 프로젝트를 통해 검증하여 제시된 검증로직이 타당함을 보인다.

1. 서론

소프트웨어 규모산정 활동은 소프트웨어사업 전 수명주기 동안 반복적으로 수행되는 활동으로 발주자나 수주자를 비롯한 다양한 이해관계자들에게 큰 영향을 미치는 중요할 활동이다. 소프트웨어 규모산정에 있어서 사용자에게 제공되는 기능을 논리적 관점에서 식별하여 그 규모를 측정하는 척도로 기능점수방식이 있다. 기능점수방식은 구현기술에 종속되지 않고, 개발언어와 도구에 독립적이며 개발 생명주기의 초기단계인 요구분석 단계에서부터 측정이 가능하다[1].

1970년대 후반 IBM의 Allan J. Albrecht는 소프트웨어 개발 프로젝트의 규모를 측정하기 위해 기능점수분석(Function Point Analysis, FPA)방식을 정의하였다[2]. FPA방식은 1984년 "IBM CIS & A Guideline 313"을 통해 확장되었으며[3], 1986년 기능점수 사용자그룹(International Function Point Users Group, IFPUG)이 결성되면서 기능적으로 소프트웨어 규모를 측정하기 위한 방법으로 개선되며, 2002년 조정인자를 제외한 4.1버전이 ISO/IEC 14143 표준 규격을 획득하였다. 2010년 IFPUG는 FPA 프로세스와 규칙을 정리하고, ISO 표준포맷에 맞도록 기능점수측정 실무매뉴얼(Counting Practice Manual, CPM) 4.3.1을 발간하였다[4].

국내 공공부문 소프트웨어 규모산정은 소프트웨어 사업대가의 기준(지식경제부 고시 제2010-52호)과 엔지니어링 사업대가의 기준을 활용하였으나, 2012년 2월 소프트웨어 사업에 적용되는 사업대가가 민간자율로 결정되도록 유도하기 위해 소프트웨어사업대가의 기준 고시는 폐지되었다. 이후 국기기관 등에서 소프트웨어사업의 대가산정시 준용할 수 있는 대체방안으로 소프트웨어산업의 동적인 상황과 글로벌 표준에 입각한 ISO12207 기반의 소프트웨어 수명주기(기획, 구현, 유지보수 및 운영) 전반에 걸쳐 대가산정을 할 수 있도록 소프트웨어사업 대가산정 가이드가 발간되었다[5]. 소프트웨어사업 대가산정 가이드는 국가·지방단체·국가 또는 지방자치단체가 투자하거나 출연한 법인 또는 기타 공공단체 등에서 소프트웨어의 기획, 구현, 운영 등 수명주기 전 단계에 대한 사업을 추진함에 있어 예산수립, 사업발주, 계약시 적정대가를 산정하기 위한 기준을 제공하는 것을 목적으로 한다.

해외 주요 국가별 소프트웨어규모산정은 기능점수 방식을 기본으로 하여 사업의 유형, 환경, 발주자의 판단 등에 따라 해당 사업에 적합하다고 판단되는 방식을 채택하고 있다. 아래의 <표 1>은 소프트웨어규모산정 및 규모산정에 기초한 소프트웨어사업대가를 산정하는 주요 사례이다.

<표 1> 소프트웨어규모산정 주요사례

국가	규모산정	대가산정	특징
호주	기초기능점수 계산방식	기능점수당 단가에 의한 개발비 산정 발주담당자가 2~3개의 유사사업 공수와 비용을 참조	정부주도의 Southern Scope 방법론 적용하며 E-government Repository 시스템 구축
이탈리아	IFPUG FPM (기능점수)	기능점수당 단가에 의한 개발비 산정, 소프트웨어사업실적 Repository를 통해 유사사업을 참조	공공행정 디지털공사 주관하에 소프트웨어사업실적 정보 관리

핀란드	FiSMA1.1 (기능점수)	개발프로젝트의 기능규모와 대가산정후 Scope Manager에게 전달하여 DB를 참조하여 값의 적정성 판단후 피드백 제공	FiSMA 핀란드 소프트웨어측정협회 주도로 정부의 사업대가 산정 및 관리제도 시행

기능점수측정에 대한 연구는 기능점수 산출시 적용하는 복잡성 및 가중치를 개선하려는 방향으로 진행되었다. 미조정 기능점수 측정에 대한 개선 방안에서는 소프트웨어의 복잡도에 따른 가중치의 반영이 적합하지 않아 각 기능 연관도를 추가요인으로 제시하여 적합한 가중치를 도출하였다[6]. 소프트웨어 규모산정을 위한 기능점수 측정 모델에서는 기존의 시스템 및 기능의 복잡도 가중치가 획일화되어 다양한 분야에 적용되지 못한 문제점을 제시하고, 다양한 조직의 특성과 기능의 복잡도를 반영할 수 있는 수학적 가중치 산출 모델을 적용하였다[7].

기존 연구에서는 개발 프로젝트의 대다수를 차지하는 업무처리용 소프트웨어에 대해 CPM에서 제시한 기능점수의 가중치가 미흡하다고 판단하여 이를 개선하려고 하였다. 본 논문에서는 기존 연구에서 간과한 기능점수 측정시 개인의 주관이 반영되어 발생하는 오차를 줄이고자 한다. 프로젝트 현장에서 기능점수를 산정한 경험을 바탕으로 실무관점에서 접근한다. 경험기반의 접근으로 본 논문에서 기능점수측정에 대한 가정 다음과 같다. 첫째, 실제 프로젝트에 대해 기능점수를 측정하면 개별 기능 즉, 단위프로세스에 대한 오차는 존재할 수 있지만 전체 프로젝트 규모의 오차는 미비하다. 둘째, 가중치를 개선하려는 기본 접근에는 해당 기능을 개발하는데 들어가는 노력(주로 투입공수)을 기반으로 한다. 예를 들어 단순한 레포팅 화면에 투입되는 공수와 복잡한 비즈니스 프로세스가 적용되는 레포팅 화면에 투입되는 공수는 다르다는 점이다. 이 부분에 대해서도 기능점수의 본래 목적은 사용자의 논리적인 관점에 기반을 둔다. 해당 기능을 구현하는데 얼마나 많은 시간을 투입하는지 보다는 사용자에게 의미와 가치를 주는 기능을 기본으로 측정한다. 따라서 기능구현의 난이도를 감안하여 복잡도와 가중치를 개선하는 것도 중요하지만 기능점수 자체의 산정오류를 줄이는 것이 필요하다.

본 논문에서는 이와 같은 가정에서 출발하여 기능점수 측정 자체의 신뢰성과 정확성을 높이는데 주안을 둔다. 기능점수 방식으로 수행된 공공기관의 프로젝트를 대상으로 기능점수 측정 오류를 분석하고, 기능점수 산정 자체의 측정 정확도를 높이고자 한다. 현재 진행되는 P 프로젝트의 기능점수 측정결과를 제시된 기능점수 검증로직을 통해 검증하여 기능점수를 보다 정확히 측정할 수 있음을 보인다.

2. 신뢰성 향상을 위한 기능점수 검증로직

기능점수의 측정의 정확도를 높이기 위해 국내 공공분야의 기능점수를 적용한 프로젝트를 분석하였다. 해당 프로젝트 데이터는 국내 소프트웨어 프로젝트의 대부분을 차지하는 개발 분야이며, 재개발분야와 유지보수 및 운영은 제외하였다. 분석된 결과로 총 4가지의 기능점수 검증로직을 개발하였고, 개발된 검증로직을 현재 진행 중인 P 프로젝트에 적용하여 비교분석한다.

프로젝트의 특성에 따라 제시된 검증로직에 맞지 않는 프로젝트가 존재할 수 있다. 즉, 검증로직은 절대적인 기준이기보다 기능점수 산정의 정확도를 높이기 위한 가이드로서의 역할을 제공한다.

검증1 : 국내 평균생산성과 비교한 편차분석

국내 평균생산성과 비교한 편차분석에 대한 기본접근은 기능점수당 투입공수를 기준으로 전체 기능점수의 오류가능성을 확인하는 것이다. 즉, 프로젝트의 투입공수 산정이 맞는다는 가정하에 생산성을 분석하여 평균생산성의 예외값을 벗어나면 전체 기능점수가 잘못 측정되었을 확률이 높다고 유추한다. 예외값은 일반적인 분포를 벗어난 상위 75%, 하위 25%를 예외로 추정하였다. 다음의 <표 2>는 2012년 소프트웨어공학백서의 데이터를 기준으로 추출한 국내 평균생산성과 예외값이다[8].

<표 2> 평균생산성

구분	생산성 (FP/MM)
평균 생산성	23.4
예외값 하한	21.8
예외값 상한	25.6

평균생산성과 P 프로젝트의 사례를 분석하면 아래의 <표 3>과 같으며 해당 프로젝트는 예외값의 범위를 벗어났음을 볼 수 있다. 현상파악 결과 프로젝트에 기획영역, 품질보증팀의 인원이 추가적으로 투입되었음을 확인할 수 있었다. 향후 직접 개발외의 간접인원이 투입되었을 경우를 고려한 생산성 분석이 필요하다.

<표 3> P 프로젝트의 생산성 적용사례

구분	평균생산성	P프로젝트
생산성(FP/MM)	23.4	17.6

검증2 : 기능별 평균비율과 비교한 편차분석

프로젝트에서 산정된 외부입력(EI), 외부출력(EO), 외부조회(EQ), 내부논리파일(ILF), 외부연계파일(EIF)의 각 기능별 비율과 수집된 25개 공공부문 프로젝트의 기능별 비율을 편차분석을 통해 비교한다. 별도로 2005년 Quality Plus Technologies에서 발표한 ISBSG의 기능별 기준을 제시하였다. ISBSG의 사례와 비교하면 국내의 경우 외부조회(EQ)의 비율이 높고, 외부연계파일(EIF)의 비율이 낮음을 알 수 있다. 편차 분석의 기준을 위한 예외 값은 다음의 <표 4>의 결과와 같다.

<표 4> 기능별 평균비율

구분	EI	EO	EQ	ILF	EIF
ISBSG	33.5%	23.5%	16.0%	22.1%	5.0%
국내	33.4%	14.2%	22.4%	27.3%	2.8%
예외값 하한	28.3%	6.5%	15.7%	22.0%	0.0%
예외값 상한	39.6%	20.9%	29.6%	34.5%	3.8%

P 프로젝트에서의 측정은 비중(P) = 각 기능점수/∑기능점수로 계산하였으며, 국내평균의 예외 값과 비교하면 다음의 <표 5>와 같이 외부출력(EO)이 높게 나타나고, 내부논리파일(ILF)이 낮게 나타남을 알 수 있다. 원인을 분석한 결과 외부출력(EO)는 통계 분석용 보고서의 비율이 높아서 높게 나타났고, 내부논리파일(ILF)은 아직 개발이 완료되어 않아 식별되지 않은 내부논리파일이 존재할 수 있을 것으로 추정된다.

<표 5> P 프로젝트의 기능별 비율 적용사례

구분	EI	EO	EQ	ILF	EIF	계
기능수	23	1	103	39	49	215
기능점수	173	5	308	203	191	880
비중(P)	35%	23%	22%	20%	1%	
국내	33.4%	14.2%	22.4%	27.3%	2.8%	

검증3 : 단위프로세스별 중복검증

공공부문 프로젝트의 기능목록은 총 17,354개의 단위프로세스로 구성되었으며, 이중 기능점수 검증을 통한 오류기능은 1,067개로 16%의 오류를 보였으며, 각 기능별로 오류의 비율은 다음의 <표 6>과 같다. 단위기능(EI, EO, EQ, ILF, EIF)의 오류와 산정되지 않아야 할 단위프로세스가 산정된 삭제오류, 중복 산정된 오류 및 사업초기에 산정되지 않았던 프로세스가 추가되는 경우가 있다.

이중 중복산정은 동일한 단위프로세스를 2번 이상 측정한 결과이며, 이것은 동일한 단위프로세스를 검증함으로 해결가능하다. 추가는 사업초기에 기능점수로 측정되지 않고 사업종료 후에 측정된 결과이며, 이 오류는 실제 프로젝트 완료 후 산출물을 기반으로 측정해야 하기 때문에 추정이 힘들다.

<표 6> 기능별 오류비율

구분	오류개수	비율
EI	255	24%
EO	38	4%
EQ	32	3%
ILF	17	2%
EIF	137	13%
삭제	165	15%
중복	183	17%
추가	240	22%
총합계	1,067	100%

검증 4: 용어사전을 이용한 오류 검색

기능점수 검증은 기능점수측정전문가(Certified Function Point Specialist : CFPS)를 이용하여 진행했으며, 검증된 결과를 확인하면 각각 오류별로 수정된 사유가 있지만 공통적으로 발생하는 오류가 존재하였다. 공통오류를 줄이기 위해 오류사유를 분석한 결과로 오류에 대한 용어사전을 <표 7>과 같이 제시한다. 해당 용어는 단위프로세스 명칭에 기반을 두어 만들어졌으며, 단위프로세스 산정시 적합하게 산정하도록 주의할 필요가 있다. <표 7>을 적용하면 단위기능(EI, EO, EQ, ILF, EIF)과 삭제에 대한 오류를 줄일 수 있을 것으로 기대된다. 해당 용어사전을 이용하여 전체 1,067개의 오류중 183개의 오류를 검출할 수 있었으며 이는 전체 17%의 오류를 줄일 수 있다. 결과적으로 [검증 3]의 중복검증과 [검증 4]의 용어검증을 통해 33%의 오류를 줄일 수 있게 된다.

P 프로젝트의 경우 검증을 통해 96개의 기능목록에서 검증로직 3과 검증로직 4를 적용하여 2개의 오류를 검출할 수 있었다.

<표 7> 오류용어사전

용어	적합	부적합	해설
데이터 적재	EI	EO,EQ	데이터적재는 EI로 산정하는 것이 타당함
발송	EQ	EO	단순 발송은 EQ로 산정하는 것이 타당함
그래프	EQ	EO	그래프는 일반적으로 EO로 산정하는 것이 타당함
다운로드	EQ	EI, EO	다운로드는 EQ로 산정하는 것이 타당함
로그인	EQ	EI, EO	암호검증 후 로그인은 EQ로 산정하는 것이 타당함
사용자 인증	EQ	EI, EO	사용자인증은 EQ로 산정하는 것이 타당함
통계	EO	EQ	통계기능은 EO로 산정하는 것이 타당함
전송	EQ, EO	EI	전송기능은 EQ 또는 EO로 산정하는 것이 타당함
코드	삭제	산정됨	코드데이터는 기능에서 제외하는 것이 타당함
임시	삭제	산정됨	임시파일은 기능에서 제외하는 것이 타당함
이력	삭제	산정됨	이력정보는 기능에서 제외하는 것이 타당함
첨부	삭제	산정됨	첨부는 단위프로세스를 완료하지 못하므로 제외하는 것이 타당함

4. 결 론

본 논문은 소프트웨어규모산정을 위한 기능점수 측정 개선 실무 사례 연구로 기능점수 자체의 신뢰성을 높이기 위한 검증로직을 제시하였다. 기능점수산정이 확산되지 못한 이유 중 하나로 산정하는 전문가마다 상이한 규칙을 적용한다는 점이 있다. 이러한 문제를 해결하기 위해 본

논문에서는 공공부문 25개 프로젝트 정보를 취합 후 분석하여 검증로직을 정의하였고, 정의된 검증로직을 진행되는 프로젝트에 적용하였다. 제시된 검증로직이 기능점수측정의 절대적인 가이드가 되지 못하지만 개별로 측정차이가 발생하는 부분을 줄일 수 있을 것으로 기대된다.

향후 연구는 프로젝트 표본을 확충하여 검증로직에 대한 신뢰성을 높일 필요가 있다. 추후 약 100여개 이상의 프로젝트 데이터를 추가하여 [검증로직 1]의 평균생산성 및 [검증로직 2]의 기능별 비율 데이터를 시뮬레이션 하여 신뢰성을 높일 계획이다. [검증로직 3]은 중복검증으로 그대로 사용가능하며, [검증로직 4]는 계속적으로 용어사전을 확장하고 전문가들의 의견을 취합하여 용어사전을 정형화시킬 예정이다. 본 연구가 향후 소프트웨어사업규모산정시 기능점수 측정을 적용하려는 조직에 이정표가 되고, 기능점수 측정 자체의 오류를 줄일 수 있을 것으로 기대된다.

참고문헌

[1] Matson J.E., Barrett B.E. and Mellichamp J.M., "Software Development Cost Estimation Using Function Points," IEEE Trans. of Software Eng., vol.20, 1994.

[2] Allan J. Albrecht, "Measuring Application Development Productivity," Proceedings of Joint SHARE GUIDE and IBM Application Development Symposium, 1979.

[3] IBM CIS & a Guidelines 313, "AD/M Productivity Measurement and Estimate Validation," 1984.

[4] IFPUG, "Function Point Counting Practices Manual, Release 4.3.1," 2010.

[5] 한국소프트웨어산업협회, "소프트웨어사업 대가산정 가이드," 2012.

[6] 이민태, "미조정 기능점수(UFP) 측정에 관한 개선 방안," 정보과학회논문지, 컴퓨터의 실제 및 레터 제17권 제7호, 2011.

[7] 정인용 외 3명, "소프트웨어 규모 산정을 위한 개선된 기능점수 측정모델," 한국인터넷정보학회 논문지, Vol10, No.04, 2009.

[8] 정보통신산업진흥원 소프트웨어공학센터, "소프트웨어공학백서 2012," http//www.software.kr