국방C4I 비용추정 사례를 통한 기능점수 개선 연구

임득수',김현수

한국원가원, 국민대학교 정보관리학부

A Case Study on Improving Function Point Method in Defence C4I Cost Estimation

Lim, Deuksoo, Kim, Hyunsoo
Korea Cost Institute, School of MIS, Kookmin University
E-mail : idsn21@hanmail.net, hskim@kmu.kookmin.ac.kr

요 약

기능점수 방법은 소프트웨어 비용산정에서 최근에 가장 많이 연구되고 있는 방법이다. 국내에서는 기능점수 방법을 소프트웨어 비용산정에 적용한 사례가 매우 적고, 실무 중심의 연구 성과 또한 적은 상황이다. 본 연구에서는 비용추정에 기능점수 방법을 적용하였던 대표적인 사례를 통하여, 기능점수 방법의 문제점을 찾아내고 체계적으로 분석하여, 이를 개선할 수 있는 방안을 모색하였다.

기능점수방식의 본질적인 문제점은 발견하기 어려웠으나, 기능점수 방법을 비용추정 실무에 적용하는 과정과 방법에 대해서는 많은 문제점이 도출되었다. 난이도 구분 방식의 문제점, 측정 도메인의 문제, 측정자의 개인차에 관한 문제 등이 중점적으로 발견되었다. 향후 적용 사례를 추가로 수집하여 보다 체계적인 연구를 수행할 필요가 있을 것이다.

1. 서론

소프트웨어 비용산정을 위해서는 정확한 규모산정이 선행되어야 한다. 소프트웨어의 정확한 규모산정을 위해서는 소스코드(Source Code) 규모와 같은 물량중심의 방식은 적합하지 않고, 사용자 관점에서의 경제적 가치를 중심으로 산정하는 방식이 보다 적합한 것으로 인식되고 있다. 사용자 가치 관점으로 규모를 산정하는 대표적인 방법이 기능점수(Function Point) 방법이다. 소스코드는 소프트웨어개발프로젝트의 최종 산출물이 되기는 하지만, 프로그래밍 언어와 기법의 발전으로 모든 프로젝트에서 작업량을 정확하게 추정해주기는 어려운 상황이다. 그러나 소스코드 방식은 측정이 쉽고, 일부 프로젝트에서는 작업량을 측정하는 바람직한 대안이 되고 있다.

소프트웨어의 규모 산정을 용이하게 하기 위하여 개발소프트웨어의 본수를 추정하고 이를 적절한 매개변수를 이용하여 스텝수로 환산하는 방법(이하 본수스텝수방식이라 함)을 사용하는 경우도 있다[한국소프트웨어산업협회, 2001]. 숙련자들에 의하여 직접 스텝수를 산정하게 하는 방법은 직접스텝수법이라하여 위 방식과 구분하여 용어를 사용한다.

본 연구에서는 이들 측정방법을 기능점수를 중심으로 특정 소프트웨어개발집단에 각각 적용하여보고 이들 3가지의 측정결과에 어떤 차이가 있으며 그 차이는 어떻게 해석될 수 있는가와

기능점수법의 취약점은 무엇인가를 분석하며, 개선방안을 도출한다.

아래 제 2 장에서 그동안 연구된 기능점수의 문제점을 요약하여 제시하고, 제 3 장에서 국방 C4I 적용사례를 통하여 기능점수의 추가적인 문제점을 도출한다. 제 4 장에서는 사례를 중심으로 기능점수 방법 및 규모산정 방법을 개선하는 방안에 대하여 토의하며, 제 5 장에서는 요약 및 결론을 제시한다.

2. 기능점수 적용의 문제점

기능점수(Function Point:FP)법은 1979년에 IBM의 Albrecht가 고안하여 발표한 이래, 현재 IFPUG(International Function Point User Group)가 기법의 개선을 주도하고 있으며, 북미, 유럽, 아프리카, 아시아 등에서 널리 사용되고 있다. 기능점수법은 MarkⅡ법 등 몇 개의 파생된 수법이 있지만IFPUG법이 가장 많이 보급되어 있다[정보통신부, 2001].

기능점수(Function Point)는 소프트웨어를 사용자의 관점으로 보아 사용자가 필요로 하는 기능의 수를 중심으로 소프트웨어의 규모를 측정한다.

기능점수는 기능수 계산, 기술적 복잡도 계산, 기능점수 계산 등의 3단계 절차에 의해 계산된다. 기능수는 개발 대상 소프트웨어가 명확하게 정의되고, 외부 시스템과의 인터페이스가 분명해진 상태에서 외부입력(External Input: EI), 외부출력(External Output: EO), 내부논리파일(Logical Internal File: LIF), 외부인터페이스파일(External Interface File: EIF), 외부조회(External Inquiry: EQ) 등의 5가지 유형의 기능을 추출하여 각각의 복잡도를 측정한다.

각 기능의 복잡도는 단순, 보통, 복잡의 3단계로 구분하여 평가한다. 기능에서 다루는 데이터 항목의 수, 입출력기능에서 참조하는 파일수와 파일이 포함하는 레코드의 종류수 조합 등 2차원 테이블에 의해 기능의 복잡도를 판단하고 있다.

기능수는 기술적 복잡도를 이용하여 기능점수로 변환된다. 기술적 복잡도는 데이터통신의 필요정도, 분산처리기능의 정도 등 14 개 항목을 5점 만점의 영향도로 평가하여 계산한다.

기능점수방식은 나름의 장단점을 가지고 있는데, IFPUG에서 말하는 강약점은 각각 다음과 같다.

1) 강점
 * 규모 산정방법이 프로그래밍 언어에 독립적임
 * 전 수명주기 분석에 탁월함
 * 벤치마크 및 경제성 연구에 좋음
 * 다른 비용추정 도구들이 다수 채택하고 있음
 * 논리코드로 쉽게 변환가능함

2) 약점
 * 아직 측정전문가가 부족함
 * 측정에 시간이 많이 소요되고 비용이 많음
 * 소규모(15 기능점수 미만) 프로젝트에서는 신뢰성 없음
 * 특수 FP와의 차이는 IFPUG FP로 변환 안됨 (예:3D function points, Mark II function points, Object points, Engineering function point, Full function points 등)

이러한 기능점수 방식의 문제점을 개선하기 위하여 국내외에서 많은 연구가 수행되었다. 대표적인 연구로는 입력, 출력, 처리를 중심으로 기능수를 추정하고, 20개의 복잡도 요소를 사용하는 Mark II모형[Symons, 1988], 화면, 대장 파일로 기능수를 산정하는 간편모형인 COSDES모형[田中淳, 1995], 내부에서 복잡한 계산이 이루어지는 시스템에 적용할 목적으로 개발된 특성점수(Feature Point)모형[Jones, 1996], 입력데이터 유형수, 엔티티타입수, 알고리즘수, 출력데이터 유형수 등을 이용하여 기능수를 산정하는 모형[김현수, 1997] 등이 있다. 이들 모형은 측정의 정확성을 유지하면서 간편성을 증대시켜 기능점수 방법의 실용성을 보완한 연구이다.

기술적복잡도를 개선한 대표적인 연구로는 Symons[1988]와 김현수[1999]의 연구가 있는데, 이들 연구는 요소를 추가 보완하고, 요소를 판단하는 세부적인 기준을 제시하여 방법의 실용성과 정확성을 높이고 있다.

실무차원에서 수행된 최근의 연구들에서는 기능유형별 평균 기능점수 수에서 많은 차이를 나타내고 있다고 보고되고 있다. 예를 들어 한국소프트웨어산업협회[한소협:2001]의 연구에서는 아래 [표 1]과 같이 ISBSG(International Software

Benchmarking Standards Group)의 기존 연구 결과와 매우 상이한 결과를 발표하고 있다.

[표 1] 기능유형별 평균 FP 수

FP유형	EI	EO	EQ	ILF	EIF	합계	평균*	차이율
한소협	3.8	4.9	3.7	8.1	5.3	25.8	4.15	100.0
ISBSG	4.3	5.4	3.8	7.4	5.5	26.4	5.99	169.3

*: 평균의 계산 : ISBSG의 FunctionPointMix를 이용한 FP 5유형의 구성율인 EI:33.5%, EO:23.5%, EQ:16.0%, ILF:22.1%, EIF:5.0%를 가중한 값

위 [표 1]은 우선 규모측정에 혼란을 가져올 가능성이 많음을 예시하고 있으며, 기능점수당 적용할 단가를 객관적으로 결정하기가 어려움을 나타내고 있다. 따라서 비용에 관한 연구를 별도로 수행하여야 하되 이와같은 규모 측정의 문제를 안고 있는 현실에서는 비용산정이 보다 어려운 과제가 된다.

기능의 복잡도 유형을 단순(Low), 보통(Average), 복잡(High) 등의 3단계로 구분하는 척도에 대해서도 문제점이 제기되고 있다. 이들 각 유형의 가중치가 기능의 난이도 수준을 적절하게 반영하지 못하여 적정한 규모산정이 되지 않는 문제점이 있는 것이다. 예를 들어 외부입력의 경우 단순의 최대값 (파일수가 1개, 필드수가 15개)이 보통의 최소값 (파일수 0, 필드수 16) 보다 적다고 할 수 없는데, 단순 최대값의 가중치는 3으로서 보통의 최소값의 가중치 4보다 오히려 1이 적은 것으로 되어 있다. 같은 논리로 보통의 최대값 (파일 1개, 필드는 제한없음)이 복잡의 최소값(파일 2개, 필드 16개) 보다 적다고 하기 어려운데, 보통의 기능점수가 복잡의 점수보다 2점이 적다. 이와 같이 기능수의 정확한 산정은 어려운 것으로 나타나고 있다.

기술적 복잡도 계산의 경우 데이터통신, 분산처리, 성능, 트랜잭션량, 온라인처리, 데이터의 복잡성 등 과거 시스템의 특성을 주로 감안한 것으로 현재는 대부분 시스템에 공통적으로 들어가 있는 것들이고 최근에는 기술적 복잡도가 이들 보정치의 거의 최상위 수준에 있다 할 수 있다. 이들은 평가자들로 하여금 상위편중현상을 초래하였고 따라서 보정치의 의미가 많이 약화되었다. 이와같은 현상은 기존 연구에서도 반영되어 IFPUG에서 제시한 보정치를 배제하고 프로젝트 목표보정 등 5개로 나누어 별도로 주는 경우도 있다[정보통신부, 2001].

아래 제 3 장에서 사례를 통하여 기능점수 적용의 최신의 문제점을 도출한다.

3. 기능점수 적용사례

3.1 기능점수의 산정

본 연구에 참여한 연구원들은 기능점수 방식을 이용하여 소프트웨어의 규모를 산정(이하 기능점수방식이라 함)하였다. 규모의 정확한 추정을 위해 기능점수 방식과 함께 본수추정에 의한 스텝수 산정방식(이하 본수스텝수방식)과 스텝수의 직접 추정(이하 직접스텝수방식)을 사용하였다.

1) 연구에 참여한 연구원

규모 추정을 위해 40명(기존 경영정보시스템 개발 경험자, 공정관리 경험자, 도구 개발자 등이 참여함)의 연구원이 약 5개월간 활동하였으며, 참여 연구원은 소프트웨어에 대한 규모 추정을 할 수 있는 전문성을 확보한 과장급 또는 선임급 연구원들로 소프트웨어 분야 경력 5년 이상의 팀장급들이다. 이들은 각자의 연구분야(각 5개 하부시스템중 1개 하부시스템은 3개분야로 연구되었으며 따라서 7개분야에 대한 산정)에 대하여 각각 기능점수와 본수 및 스텝수를 추정하였다.

2) 추정대상 소프트웨어의 내용

본 연구는 국방 C4I 소프트웨어에 대한 연구로 1개의 종합시스템하에 5개의 하부시스템으로 구성되어있으며, 그 하부시스템하에 다수의 하위시스템 모듈이 구성되어있다. 각 하부시스템별 기능수와 기능점수는 산정된 기능수를 가중치를 적용하여 기능점수로 환산하였고, 이 가중치는 단순 보통 복잡별로 EI, EO, EQ, ILF, EIF에 따라 개별치를 부여하여 산정하였다(아래[표 2] 참조). 특히 본 연구는 개념연구에 의한 결과에 대하여 각각을

산정한 것으로 설계 또는 개발의 결과를 분석한 것이 아니며, 개발전에 소프트웨어 규모의 추정을 수행한 것이다.

3) 조사 내용

본 연구에서는 참여한 연구원들에게 설문하여 자신이 수행한 당 프로젝트를 위한 개념연구에서의 연구 영역별 응용 프로그램을 추정 예측하고 이들 응용프로그램별 기능점수를 개별로 산정하게 하였으며, 아울러 본수추정 및 직접 스텝수를 추정하게 하였다. 추정을 위한 사전지식을 이해시키기 위하여 기본개념에 대하여 설명하고 작성서식에 대한 설명 및 질의응답을 거쳤으며, 작성도중에도 비용분석 연구원이 상주하면서 이들의 질의에 응답하였다.

한편, 측정의 정확성을 검증하기 위하여 델파이법에 의하여 프로젝트 전반에 걸친 추정을 하였는데, 당 추정에 참여한 연구원들과 추정결과치에 대한 합동토의를 수행하였고, 전 시스템에서 차지하는 개별 시스템의 규모를 기능점수나 스텝수에 의한 결과와 비교하기 위하여 연구과정에서 직관적으로 느끼는 각 하부시스템의 규모를 당 프로젝트의 통합담당자와 프로젝트관리자, 본 업무에 다소간의 경험을 가진자. 발주자중 실무자, 발주자중 관리자 등에게 설문 조사하였다.

설문은 최종적으로 자신이 검토한 해당 하부시스템의 개발을 자신이 수행할 경우 그 가격이 얼마정도이면 적정하겠는가에 대하여 상 하한 범위를 정하여 추정하게 하였다.

추정된 각 하부시스템별 기능수를 가중치를 적용하여 기능점수로 환산하였고, 이 가중치는 단순 보통 복잡별로 EI, EO, EQ, ILF, EIF에 따라 개별적으로 적용하였다(아래[표 2] 참조).

4) 기능점수 산정결과

하부시스템 각각에 대하여 기능수와 기능점수(여기서는 수정 전 기능점수 UFP를 가리킴)를 산정한 결과를 종합하여, 기능별, 복잡도별로 집계한 결과는 다음[표 2]와 같다.

[표 2] 기능점수 산정표

기능수 / 기능점수	단순	보통	복잡	계	구성비	한소협 연구	ISBSG 구성비
EI	122	215	483	820	28.5%	3.8	4.3
	366	860	2898	4124	18.5%	14.7%	16.3%
EO	5	69	152	226	7.8%	4.9	5.4
	20	345	1064	1429	6.4%	19.0%	20.5%
EQ	101	113	305	519	18.0%	3.7	3.8
	303	452	1830	2585	11.6%	14.3%	14.4%
ILF	68	558	308	934	32.4%	8.1	7.4
	476	5580	4620	10676	48.0%	31.4%	28.0%
EIF	39	72	272	383	13.3%	5.3	5.5
	195	504	2720	3419	15.4%	20.5%	20.8%
계	335	1027	1520	2882	100.0%	25.8	26.4
	1360	7741	13132	22233	100.0%	1	1
구성비	11.6%	35.6%	52.7%	1			
	6.1%	34.8%	59.1%	1			

비고: 본 시스템의 하부시스템 5개, 조사대상 7개분야에 대하여 각각 141, 369, 823, 463, 269, 158, 632 기능, 합 2,882 기능으로 산출되었으며 , 한소협연구와 ISBSG구성비는 [표 1]에서 제시된 비율을 사용하였음.

위 표와 같이 EI, EO, EQ, ILF, EIF의 각 기능별로 기능수가 얻어졌으며, 이들을 기능점수로 환산한 결과 총 22,233기능점수가 산출되었다. ILF가 32.4%로 가장 많이 측정되었으며, EQ, EIF 순으로 많은 기능수를 보이고 있다.

3.2 산정과정의 문제점

이와 같이 기능점수방법을 사용하여 규모 추정을 하는 과정에서 다음과 같은 문제점이 발견되었다.

1) 도출된 결과에 관련된 문제

① 단순, 보통, 복잡의 의미

기능의 난이도에 부여하는 단순, 보통, 복잡의 구분을 좀 더 세분화할 필요성이 나타났다. 본 사례에서 단순의 경우 기능수로 11.6%를, 이를 기능점수로 환산한 후에는 6.1%를 기록하고 있으며, 보통의 경우도 각각 35.6%와 34.8%를, 그리고 복잡에서 52.7%와 59.1%를 각각 기록하고 있다. 복잡으로 분류된 유형들을 분석한 결과 같은 가중치를 가지기에는 매우 난이도가 차이나는 기능들이 많이 발견되었다. 예를 들어, 국방업무의 작전을 지시 통제하는 소프트웨어의 경우 그 난이도가 대단히 높아 현행의 가중치인 3,4,6 또는 4,5,7로는 구분이 불가능하다.

② 각 기능별 구성비

본 연구에서 5가지 기능 유형기준을 적용하여 기능점수를 추정한 결과 기능점수의 각 기능별 구성비는 EI와 ILF에 집중되어 있으며 이들 두 기능의 합이 60%(기능점수로 보면 66.5%. 이는 ISBSG의 Function Point Mix로 볼 때 EI와 ILF의 합이 55.6%에 비하여 10% 이상 초과하는 수준임)를 넘는 것으로 나타났다. EO의 경우 6-7%대로 EQ에 비해서 상대적으로 낮은 비율을 보이고 있다. 이는 본 시스템의 개발단계가 아직 개념연구단계이므로 출력에 대한 개념이 아직 없음에 기인한다고 할 수도 있겠으나 본 시스템이 주로 작전용 들이므로 출력보다는 조회에 더 많은 비중을 두고 있기 때문으로 볼 수 있다.

국방C4I사업의 경우 지금까지 사용되었던 단순, 보통, 복잡의 구분이 보다 세분화될 필요가 있고, 특히 복잡의 구분에서 더 상세한 구분이 필요함을 말해주고 있다. 각 기능별 기능점수에 있어서도 EI와 ILF쪽에 비중이 많이 두어져 이 부분에 주의를 기울여 기능점수를 산정할 필요가 있다는 것을 의미한다고 할 수 있다.

2) 본수스텝수 방식과의 비교

기능점수에 의하여 산정된 소프트웨어 규모의 측정 신뢰성을 확인하기 위하여 기존의 전통적인 규모 측정 방법들인, 본수, 스텝수 방식과 비교 분석 하였다. 본수를 추정하여 산정한 스텝수를 기능점수에 의하여 산정한 결과와 비교하면 다음과 같다. 추정된 본수는 1,491본이었고, 현행 소프트웨어사업대가기준의 스텝수 환산기준에 의하여 환산한 결과, 22,233 기능점수로 환산되었다. 즉, 1 기능점수는 평균 27.56스텝이 되는 것으로 나타났다.

이는 기존 연구[한국소프트웨어산업협회,2001]에서의 1 기능점수당 평균 56스텝보다 매우 적은 수치로서, 스텝수와 기능점수의 관계에 대한 재검토가 필요함을 시사하고 있다.

[표 3] 추정된 본수로 스텝수를 산정한 결과

형 태	본수기준	본당스텝	환산스텝수
batch	15	470	7,050
on-batch	1,463	410	599,830
real time	13	460	5,980
total	1,491		612,860

비고: 정보통신부 고시 '소프트웨어사업대가 기준'의 '1본 당 스텝'수치를 사용함.

3) 직접스텝수 방식에 의한 산출과의 비교

참여한 연구원들에게 자신이 직접 프로그램을 작성할 경우 그 최종 결과 산출물이 몇 스텝 정도 이겠는가 라는 질문에 각 연구원들은 다음 [표 4]와 같이 추정하였다.

[표 4] 스텝수 산출과 FP와의 관계

하부시스템	본스텝수	직접스텝수	추정 FP	비율	FP당스텝수	
					본스텝	직스텝
Sub-1	30,070	32,400	716	3%	42	45
Sub-2	81,180	421,020	3,555	16%	23	118
Sub-3	134,890	375,500	6,620	30%	20	57
Sub-4	69,000	332,310	3,859	17%	18	86
Sub-51	78,370	76,000	1,567	7%	50	49
Sub-52	45,920	309,500	1,374	6%	33	225
Sub-53	173,430	824,670	4,542	20%	38	182
계	612,860	2,371,400	22,233	100%	28	107

위 표에서 보는바와 같이 소프트웨어사업대가기준에 의한 본수기준 스텝수는 612,860인데 비해 연구원들이 직접 추정한 스텝수는 2,371,400스텝으로 상당한 차이가 나고 있다. 이를 기능점수로 환산하면 전자가 1 기능점수당 평균 28스텝, 후자가 1 기능점수당 평균 107스텝으로 무려 3.8배나 차이가 나고 있다. 따라서 1 본당 평균 스텝수에 대한 재 검토가 필요할 것이며, 추정의 정확성을 제고하기 위한 별도의 방법에 대한 논의가 필요할 것으로 판단된다.

4) 기능점수의 신뢰성 검증

하부시스템별로 산정한 기능점수는 당해 시스템의 프로그램들이 대부분 난이도가 높은 프로그램으로 인식되고 있어 보정계수를 적용할 경우에도 각 시스템별 구성비율이 크게 변화되지는 않을 것이다. 각 하부시스템별로 보정계수를 별도로 적용하기도 쉽지 않아 거의 그대로 기능점수로 평가된다. 본 연구에서는 그 신뢰성의 검증을 위하여 델파이 기법에 의한 별도의 추정을 통하여 분석하였다.

델파이기법을 사용하여 본 설문에 응답한 연구원들 및 프로젝트관리자, 시스템 통합 담당자, 발주기관의 개발관리자, 관리책임자 등에게 하부시스템별로 규모를 추정하게 한 결과, 하부시스템

3(sub-3)의 규모는 전체시스템의 최저 10%, 최대 15% 비중을 가지는 것으로 나타났다. 따라서 기능점수를 이용하여 추정된 30%라는 비중은 과다 추정된 것으로 보여, 기능점수 방식의 신뢰성에 대한 재검토가 요구된다. 상세논의를 제4장에서 수행한다.

4. 토의

4.1 스텝수 산정에 관한 문제

현재의 소프트웨어사업대가기준에서 정한 배치 프로그램 1 본당 470스텝, 온라인 프로그램 410스텝, 리얼타임 프로그램 460스텝은 본 사례에서는 잘 맞지 않는 것으로 나타났다. 본 사례에서 추정작업을 수행한 연구원들의 주관적 판단이 크게 개입되어 기능점수당 스텝수가 더욱 크게 산출되었을 가능성을 배제할 수 없다. 자신이 검토한 하부시스템을 크게 보이게 하려는 과시효과가 상당히 작용하였다고 판단된다. 추후에 1본당 스텝수를 재산정하고자 하였을 때, 평균 스텝수에 대하여는 의견의 일치를 보이면서도, 자신의 추정을 수정하려는 시도는 하지 않았던 사실로 미루어 이와 같은 분석을 할 수 있다. 연구원들이 스텝수를 추정할 때 자신의 과거 경험에 비추어 1 본당 또는 1 기능점수당 스텝수를 추정하였기 때문에, 개인차이가 많이 나타났다. 즉 개발 언어는 물론이고, 모듈에 의한 구조적 접근방법을 적용하였는가 등에 따라 상당히 차이가 난다고 판단된다.

본 연구에서는 추정에 참여한 각 연구원들에게 1본당 평균스텝수를 단순 500스텝, 중급 1,000스텝, 고난이도 1,500스텝을 제시하여 의견 수렴 여부를 조사하였다. 그러나 각 연구원들은 위 수치를 사용하지 않고 각자의 판단에 의한 개별적인 수치를 사용하였으며, 이는 추정자의 과거 경험이 추정에 많은 영향을 미치고 있음을 입증하는 것이라 할 수 있다.

본수 추정치가 비교적 정확하다고 가정하면 현재 기준에서 정하는 본당 스텝수에 문제가 있을 수 있으며(감리 경험을 통하여 판단하면 본당 1000스텝정도 산정하는 것은 무리가 없을 것으로 판단됨), 스텝수가 다소 적게 책정된 것으로 볼 수 있다. 그러나 직접추정에 의한 스텝수는 과다한 것으로 나타났는데, 그 원인은 개인별 본당 스텝수를 산정한 것이 연구원별로 난이도의 상중하를 구분하여 500에서 3,500스텝까지 상당히 큰 격차를 나타내는데 있다.

4.2 기능점수에 의한 산정과 난이도 평준화의 문제

기능점수에 의한 규모산정은 [표 4]와 같은데, 여기서 각 5개 서브시스템별로 산정한 결과는 3, 16, 30, 17, 34%이다. 여러 관계자의 의견을 종합한 결과 하부시스템-1은 다소 과소하게 책정되었으며 하부시스템-5는 기능이 세부적으로 나뉘어짐에 따라 다소 과다하게 책정되었다고 보여진다. 그러나 앞에 언급한 바와 같이 하부시스템-3은 기능점수를 사용한 경우 전체의 30%로 추정된데 비해, 델파이기법을 이용한 전문가 판단에 의한 추정치는 전체의 10%내지 15%로 나타나, 기능점수 측정의 신뢰성을 재검토할 필요성이 대두되었다.

각각의 개발규모를 추정하기 위하여 개별 연구원들에게 본인이 검토한 시스템에 대하여 개발비를 설문조사한 결과, 하부시스템-3의 경우 최저 8%에서 최대 9%에 머물렀다. 즉 하부시스템-3이 기능점수로 전체에서 차지하는 비율은 30%인데 전체적으로 개발규모는 10%이하인 것으로 나타난 것이다.

여기서, 만약 조정계수를 하부시스템-3에서 최저로 반영하였을 경우 본 시스템은

VAF = (TDI * 0.01) + 0.65

에서 VAF는 0.65가 될 것이고 타 시스템들을 최고치(1.35)로 가정할 경우, 하부시스템-3이 차지하는 비율 30%는 최저 17.1%로서 델파이기법을 적용한 경우의 최대한에 근접하게 된다.

그러나 국방프로젝트의 경우 어떤 특정한 프로젝트라 해도 보정계수를 최저로 가정하기는 어려울 것이므로 본 사례를 통하여 기능점수 산정에는 아직 문제가 상존한다고 볼 수 있다.

4.3 국방프로젝트에서의 기능점수 산정방법

소프트웨어개발의 특수한 분야인 국방프로젝트의 경우 일반적으로 개별 프로그램의 스텝수는 그 논리의 복잡성으로 인하여 업무용 소프트웨어보다 많다고 할 수 있을 것이다. 본 연구에서도 이 사실이 나타나 있으며, 각 기능별로 단순 보통 복잡

의 각 단계에서 복잡으로 판정한 부분의 비율이 60%에 가까운 것으로 보아 난이도가 높은 시스템을 정확하게 규모산정하기 위해서는 기능점수 방법에 대한 보완이 필요한 것으로 보여진다.

규모 산정과는 별도로 비용산정에 관한 많은 이슈들이 제기되었는데, 본 연구의 범위가 아니므로 추후 연구에서 언급한다.

5. 요약 및 결론

소프트웨어 규모산정에는 현행 소프트웨어사업대가기준에서 정하는 본수스텝수방식과 기능점수방법 및 직접 스텝수를 산정하는 방법이 있다. 본 논문에서는 이중 기능점수법에 집중하여 특정 사례를 대상으로 개념연구 단계에서 소프트웨어의 규모를 사전 추정하여 보았다. 그 결과 3가지 방식에 의한 규모추정에서 동일인에 의하여 추정함에도 편차가 많으며 특히 일부의 프로그램에서는 큰 편차가 나타났다.

스텝수를 직접 산정하는 방법에 있어서는 개인별로 차이가 크므로 이를 처음 도입할 시 개별 프로그램 본이나 개발되는 프로그램의 스텝수에 관하여 상당한 교감을 갖는 것이 필요하다. 본 연구의 경우 사전 지침이나 합의가 약한 상태에서 개인별로 추정하였기 때문에 이러한 큰 차이가 있었던 것으로 보아진다.

결론적으로, 기능점수 방법을 소프트웨어 예가산정에 도입할 경우 추정하는 인력들에게 충분한 교육과 가이드라인을 제공한 후 추진해야 할 필요가 있다. 향후 다른 도메인에서 기능점수방법 적용에 대한 복수개의 사례를 수집하여 본 연구의 기초 결과를 심화시키고 확인할 필요가 있을 것이다.

[참고 문헌]

[1] 김현수, 기능점수를 이용한 소프트웨어 규모 및 비용산정 방안에 관한 연구, 경영과학, 제14권 제1호, pp.131-149, 1997

[2] 김현수, 소프트웨어 비용산정을 위한 기능점수 모형 개선 연구, 한국정보처리학회논문지, 제6권 제9호, pp.2403-2413, 1999

[3] 정보통신부, 소프트웨어사업대가기준 개선 연구, 연구보고서, 2001.12

[4] 한국소프트웨어산업협회, 소프트웨어사업대가기준해설, 2001.2

[5] 田中 淳, 클라이언트/서버의 어려운 문제 - 개발공수 견적에의 도전, 일경컴퓨터 1995.3.20, pp.114-125

[6] IFPUG, International Function Point Users Group(IFPUG), Function Point Counting Practices Manual, Release4.1., Westerville, Ohio, 2000

[7] Jones, C., A Short History of Function Points and Feature Points, Software Productivity Research Inc., Burlington, MA, 1986

[8] Symons, C.R., "Function Point Analysis: Difficulties and Improvements", IEEE Transactions on Software Engineering, Vol.14, No.1, 1988, pp.2-11