經營科學
第20卷 第2號
2003年 11月

국내 소프트웨어 개발사업에 적합한 기능점수규모 예측방법에 관한 연구

박찬규* · 신수정* · 이현옥*

A Study on Estimating Function Point Count of Domestic Software Development Projects

Chan-Kyoo Park* · SooJeong Shin* · Hyunok Lee*

Abstract

Function point model is the international standard method to measure the software size which is one of the most important factors to determine the software development cost. Function point model can successfully be applied only when the detailed specification of users' requirements is available. In the domestic public sector, however, the budgeting for software projects is carried out before the requirements of softwares are specified in detail. Therefore, an efficient function point estimation method is required to apply function point model at the early stage of software development projects.

The purpose of this paper is to compare various function point estimation methods and analyse their accuracies in domestic software projects. We consider four methods : NESMA model, ISBSG model, the simplified function point model and the backfiring method. The methods are applied to about one hundred of domestic projects, and their estimation errors are compared. The results can used as a criterion to select an adequate estimation model for function point counts.

Keyword : Function Point, Software Sizing, Cost Estimation

논문접수일 : 2003년 9월 2일 논문게재확정일 : 2003년 10월 30일
* 한국전산원 정보기술감리팀

1. 서 론

정보시스템이 정부 및 민간분야 모든 업무의 필수적인 수단으로 확산됨에 따라 정보시스템을 구축·유지보수·운영하는데 소요되는 비용이 지속적으로 증가하고 있다. 1998년 이후 공공부문 정보화예산은 매년 평균 23% 증가하였고, 2001년 기준으로 우리나라의 정보화지출 총액은 약 48조원으로 GDP의 8.9%를 차지하고 있다([6]). 특히, 정보화지출 중 소프트웨어부문이 차지하는 비중이 매년 증가하여 2001년에는 전체 정보화지출 중 24%를 차지하여 하드웨어부문보다 그 비중이 높게 나타나고 있으며, 국내 소프트웨어산업 시장규모도 연간 100억 달러 이상으로 추정될 정도로 급속히 성장하고 있다([6]). 이처럼 국가예산에서 정보화관련 비용이 차지하는 비중이 높아짐에 따라, 정보화예산 관리의 효율화를 위해 정보화사업의 소요비용을 사전에 정확히 예측하는 것이 중요한 문제로 대두되고 있다.

정보시스템 구축사업의 경우 전체 사업비용은 크게 소프트웨어개발비와 하드웨어/소프트웨어 구입비로 구분할 수 있다. 하드웨어/소프트웨어 구입비는 시장가격 조사를 통해 쉽게 예측할 수 있으므로 본 연구에서는 논외로 한다. 소프트웨어개발비는 요구사항의 불확실성과 모호성, 소프트웨어 생산성 측정의 어려움, 과거 데이터의 부족, 신기술에 의한 위험 및 개발환경 의존성 등으로 인해 예측이 쉽지 않다([31]). 소프트웨어개발비를 예측하기 위해서는 먼저 개발하고자 하는 소프트웨어의 규모를 측정하고, 소프트웨어 규모와 사업의 특성 등을 고려하여 소프트웨어개발비를 추정한다. 소프트웨어 규모를 측정하는 방법으로 가장 널리 사용되는 방법은 프로그램 라인수(line of code) 방식과 기능점수모형(function point model)이 있고([28, 29]), 근래에 Kumar[21]는 은행 프로젝트의 생산성 분석에 유용한 객체점수(object point)방식을 제안하였다.

우리나라에서는 소프트웨어개발 및 유지보수사업의 규모 산정에 주로 라인수 방식을 사용해 왔는데, 라인수 방식은 개발자 중심적으로 사업 초기에는 라인수 예측이 어렵고 개발환경에 영향을 받는 등 여러 가지 단점을 갖고 있다([11, 18, 24]). 라인수를 이용한 소프트웨어 비용산정 모형으로는 Boehm [8], Putnam & Myers[29] 등의 연구가 있다. 반면, Albrecht[7]가 제안한 기능점수모형은 정보시스템이 제공하는 기능의 수를 사용자 관점에서 측정하여 소프트웨어 규모를 산정하는 방법으로, 사용자가 이해하기 쉽고 개발환경에 독립적이라는 장점을 갖고 있다([11, 24]). 기능점수를 이용한 비용산정 모형으로는 COCOMO II[8], SPQR/20[16], ISBSG [13] 등이 있다.

미국을 비롯한 호주, 일본, 유럽 등에서는 1980년대부터 소프트웨어 생산성 측정과 비용/납기 예측을 위해 기능점수방법을 적용하고 문제점을 보완하기 위한 연구가 꾸준히 진행되어 왔다. Symons [30]은 Albrecht가 제안한 기능점수모형에서 내부논리파일(internal logical file) 대신 개체(entity)를 사용할 것과 기능점수 계산을 보다 간편화하고 오차를 줄일 수 있는 Mark II모형을 제시하였다. Low & Jeffery[23]는 조직, 계측자에 따라 기능점수값이 일관성이 있는가를 실험적으로 비교하였고, Kemerer & Porter[18]는 계측자간 기능점수 산정에 편차가 발생하는 이유를 제시하였으며, Jeffery, Low & Barnes[14]는 Albrecht의 기능점수방법과 Jones[15]에 의해 제안된 수정된 기능점수 방법에 의한 비용예측 결과를 비교하였다. 또한, 기능점수 방법에서 사용되는 5가지 기능유형(function type) 간의 상관성(correlation)에 관한 연구가 Kitchenham & Känsälä[19], Lokan[22] 등에 의해 이루어졌고, Kitchenman[20]은 기능점수 적용 과정에서 나타나는 문제점을 분석하였다. Orr & Reeves[27]은 실제 프로젝트 사례를 통해 소프트웨어 수명단계마다 기능점수값이 어떻게 변화하는가를 추적·분석하였다.

기능점수를 이용한 소프트웨어 비용산정에 관한 국내 연구로 김현수[1]는 클라이언트/서버 환경, 대

형 소프트웨어 프로젝트, 사무처리용 및 알고리즘 중심적인 소프트웨어 개발 등에 적합한 수정된 기능점수 모형을 제시하고, 설문조사를 통해 모형간의 적합성, 활용성, 예측정확성 등을 비교하였다. 이양규[3]는 34개의 공공부문 프로젝트를 대상으로 설문을 통해 기능점수와 소요인력을 조사하여, 기능점수와 투입인력간의 회귀식을 제시하였다. 또한, 박찬규 등[2]은 사업초기단계에 소요인력 예측을 위한 간이기능점수모형을 제시하고 간이기능점수와 투입인력간의 회귀식을 제시하였다.

기능점수에 관한 대부분의 기존 연구에서는 프로젝트의 모든 기능유형과 데이터 요소들이 식별되어 정확한 기능점수 산정이 가능하다는 것을 전제하고 있다. 특히, 국내에서는 소프트웨어 사업대가기준[4]에 의해 소프트웨어 개발비가 결정되는데, 소프트웨어사업대가기준은 기능점수로부터 소프트웨어개발비용을 산정하는 절차와 단가를 정하고 있다. 그러나, 실제 프로젝트 사례에서는 분석/설계단계가 완료되기 이전에는 개략적인 기능유형과 개체들만 파악할 수 있을 뿐이다. 특히 국내의 경우 정보화사업의 예산은 사업이 착수되기 일년전에 결정되는데, 예산신청 및 예산결정시 소프트웨어에 대한 요구사항분석이 기능점수방식을 적용할 만큼 상세하지 않는 경우가 많다. 따라서, 정보화사업의 예산신청 및 예산결정, 예정가격산정 등 사업초기단계에서도 기능점수를 활용하기 위해서는 적은 노력으로 기능점수를 정확하게 예측할 수 있는 방법이 요구된다.

사업초기단계에 기능점수방식을 적용하기 위해 Meli[25], NESMA[26], ISBSG[13] 등이 기능점수 예측방법을 제시한 바 있다. 그러나, 각 방법을 국내 프로젝트 적용하여 그 적합성을 분석한 연구는 이루어지지 않고 있다. 특히, 소프트웨어 개발비용 예측은 각 국가마다 갖는 고유한 특성 때문에 다른 국가에서 개발된 예측모형이 잘 맞지 않는 경우가 많다. 이러한 이유로 소프트웨어 비용예측을 위한 상용도구나 모형을 사용할 때 각 나라의 상황에 맞게 여러 가지 매개변수들을 적절히 설정하는 것이 필수적으로 요구된다.

본 연구는 기능점수예측에 사용되는 여러 가지 모형들을 국내 소프트웨어사업에 적용하고 그 정확도를 비교·분석하는데 그 목적이 있다. 이를 위해 국내 2년에 거쳐 100여개의 소프트웨어개발사업의 결과물을 수집·분석하여 비용에 관한 자료를 체계적으로 수집하였다. 기능점수예측방법으로는 NESMA, ISBSG에 의해 제시된 해외모형과 더불어 간이기능점수모형과 프로그램 본수를 활용한 예측방법을 고려하였다. 본 연구결과는 사업초기단계에 기능점수예측을 보다 용이하게 함으로써 기능점수방식의 소프트웨어 사업대가기준이 국내에 안정적으로 정착하는데 기여할 수 있을 것이다. 또한, 공공부문 정보화사업을 기획·심의하거나 직접 수행하는 사람이 정보화사업 착수 이전에 사업비용을 쉽고 정확하게 예측하는데 도움을 줄 수 있을 것이다.

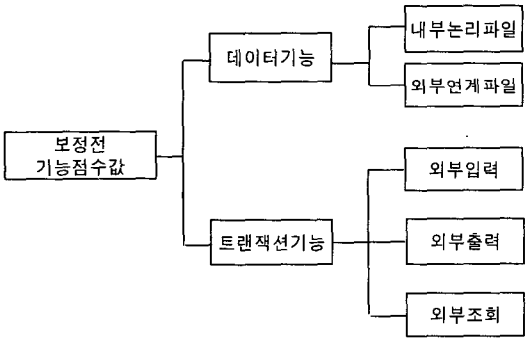
본 논문의 구성을 살펴보면, 2장에서 기능점수방식의 특징과 소프트웨어 사업대가기준을 간략히 소개하고, 3장에서 사업초기단계에서 활용할 수 있는 여러 가지 기능점수예측방법에 대해 논의한다. 4장에서는 국내 소프트웨어사업에 적용한 결과를 제시하고, 5장에서는 적절한 기능점수예측방법을 선택하는 기준을 제시한다. 마지막으로, 6장에서는 본 연구의 결론과 추후 연구과제를 제시한다.

2. 기능점수모형

Albrecht[7]가 기능점수모형을 제안한 이후, 이를 보완하고 조직에 맞게 수정한 여러 가지 변형된 기능점수모형이 제안되었다. 미국, 호주, 일본 등에서 사용되고 있는 기능점수모형은 국제기능점수사용자그룹(IFPUG)이 제시한 모형이지만, 영국에서는 MkII 모형이 활용되고 있으며([30]), 최근 실시간(real-time), 다계층소프트웨어(multi-layered software)의 기능을 측정할 수 있는 COSMIC-FFP[9] 모형이 국제표준으로 승인되었다. 본 연구에서는 국제적으로 가장 널리 활용되고 있는 국제기능점

수사용자그룹에 의해 제시된 기능점수모형을 사용하였으며, 이에 관한 자세한 내용은 [12]를 참고하기 바란다.

기능점수방식은 사용자의 관점에서 소프트웨어 규모를 산정하는 방법으로, 논리적 설계를 기초로 사용자에게 제공되는 소프트웨어 기능의 수를 정량화하여 소프트웨어의 규모를 산정한다. 기능점수는 구매하고자 하는 응용패키지의 규모 산정, 소프트웨어의 품질과 생산성 분석, 소프트웨어 개발과 유지보수를 위한 비용과 자원 소요 산정 등에 사용된다. 기능점수분석(function point analysis)에서는 먼저 데이터기능, 트랜잭션기능의 기능점수값을 구한 다음, 시스템 특성에 따라 그 값을 보정함으로써 최종적인 소프트웨어의 기능점수값을 산정한다. 보정전 기능점수값은 데이터기능과 트랜잭션기능이라는 두 가지 유형을 갖는다. 이러한 두 가지 유형을 보다 세분화하면 다음 그림과 같다.



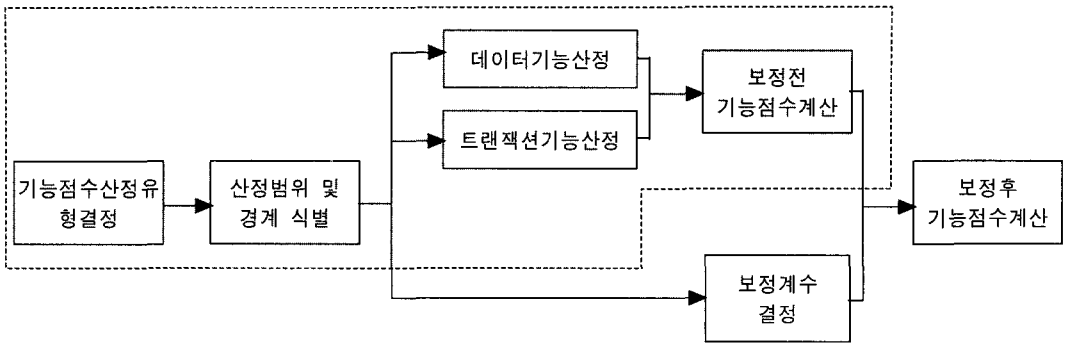
〈그림 1〉 기능점수 유형

데이터기능(data function)은 내부 및 외부 자료요구사항을 만족시키기 위해 사용자에게 제공되는 기능을 말한다. 데이터기능에는 내부논리파일(Internal Logical File, ILF)과 외부연계파일(External Interface File, EIF)이 있다. 내부논리파일은 시스템 영역 안에서 유지(maintain)되고 논리적으로 연관된 자료 및 제어정보(control information)의 그룹을 말하고, 외부연계파일은 시스템이 참조하지만 타시스템에서 유지되는 자료 및 제어정보의 그룹을 말한다. 트랜잭션기능(transactional function)은 데이터를 처리하는 기능을 말하는데, 외부입력(External Input, EI), 외부출력(External Output, EO), 외부조회(External inQuiry, EQ) 등 세 가지 종류의 기능이 있다. 외부입력은 시스템 외부에서 들어오는 데이터 및 제어정보를 처리하는 기본프로세스이고, 외부출력과 외부조회는 시스템 밖으로 데이터나 제어정보를 출력하는 보내는 기본프로세스이다. 외부출력은 외부조회와는 달리 데이터 및 제어정보의 조회 외에 처리로직을 통해 유도되는 데이터를 생성한다. 내부논리파일과 외부연계파일은 데이터요소유형(Data Element Type, DET)과 레코드요소유형(Record, Element Type, RET)의 개수에 따라 복잡도를 결정하고 그 복잡도에 따라 보정전 기능점수를 <표 1>과 같이 산출한다. 외부입력, 외부출력 및 외부조회는 참조파일유형(File Type Referenced, FTR)과 데이터요소유형(Data Element Type, DET)의 개수에 의해 복잡도를 결정하고 복잡도에 따라 기능점수가 <표 1>와 같이 결정된다.

<그림 2>와 같이 기능점수 산정절차는 총 7단계로 이루어진다. 첫 번째로 기능점수산정유형(type of count)을 결정하는데, 본 연구에서는 소프트웨어 개발사업을 대상으로 하므로 개발유형을 선택한다. 두 번째로 산정범위와 산정경계를 식별한다.

〈표 1〉 기능유형별 기능점수

복잡도 등급	보정전 기능점수				
	내부논리파일	외부연계파일	외부입력	외부출력	외부조회
낮음(low)	7	5	3	4	3
보통(average)	10	7	4	5	4
높음(high)	15	10	6	7	6



〈그림 2〉 기능점수 산정 절차

다음에는 데이터기능과 트랜잭션기능을 각각 산정한 다음 그 결과를 합산하여 보정전 기능점수를 계산한다. 보정전 기능점수가 계산되면, 시스템의 특성을 반영하는 보정계수를 계산하여 기능점수를 보정하게 된다. 보정요소에는 데이터 통신(data communications), 분산 데이터 처리(distributed data processing), 성능(performance), 사용환경(heavily used configuration), 처리율(transaction rate) 등 14개가 있다. 최종적으로 보정후 기능점수가 산출되지만, 본 연구에서는 보정계수 결정과 보정후 기능점수계산을 제외한 음영으로 표시된 5단계만을 수행한다. 이는 국내의 소프트웨어사업대가기준이 보정전 기능점수를 기초로 하기 때문이다.

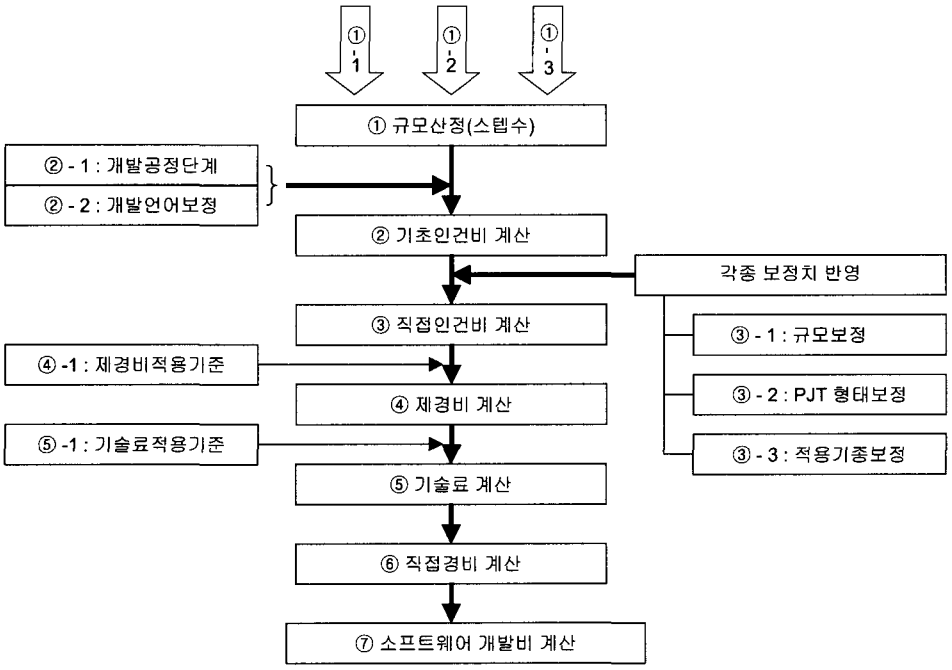
이상의 기능점수모형이 사용자 중심이고, 프로그래밍 언어 등에 독립적인 일관성 있는 척도라는 장점이 있지만, 여러 가지 보완되어야 할 점도 발견되었다. 첫 번째로, 소프트웨어 수명주기 중에서 요구사항 분석단계 이후부터 적용될 수 있는 규모산정 방법이라는 점이다. 대부분의 정보화사업이 정보전략계획(Information Strategy Planning, ISP) 없이 시작되거나 또는 ISP를 통한 개략적인 기능과 개체만 결정된 상태에서 시작된다. 따라서, 사업제안과 예산심의 시점에서 기능점수모형이 실질적으로 활용될 수 있기 위해서는 기능점수를 쉽게 구할 수 있는 방법이 필요하다.

두 번째로, 기능점수모형의 데이터기능, 트랜잭션기능들이 기능점수에 기여하는 정도가 절대적으로 평가되지 않는다는 것이다([20, 30]). 예를 들어, 내부로직파일의 복잡도는 '낮음', '보통', '높음'의 등급으로 평가될 뿐이며 연속적인 값으로 평가되지 못한다. 또한, 각각의 복잡도에 따라 주어지는 기능점수가 잘 맞지 않을 수도 있다. '낮음'인 내부로직파일과 '높음'인 내부로직파일의 기능점수비가 7 : 15인데 이는 프로젝트 유형이나 조직의 프로세스 수준 등에 따라 달라질 것이다.

세 번째로, 5가지 기능유형(ILF, EIF, EI, EO, EQ)간에 상관관계가 있다는 점이다. Kitchanham & Känsälä[19], Lokan[22]에 의하면, EI와 EO, EI와 EQ, EI와 ILF, EO와 EQ, EO와 ILF, EQ와 ILF 간에 상관관계가 있는 것으로 나타났다. 특히, 개발프로젝트의 경우, ILF, EI, EQ, EO간의 강한 상관관계가 존재하는 것으로 나타났으며, EIF는 다른 기능유형과 상관관계가 없는 것으로 나타났다.

네 번째로, 보정계수의 결정은 다소 주관적 판단에 의존하도록 되어 있으며, 보정요소에 대한 복잡도도 절대적인 값이 아닌 등급으로 결정된다. 실제로, Kemerer[18], Low & Jeffery[23]에 의하면 동일한 조직의 사람들도 동일한 프로젝트에 대한 보정요소 등급을 다르게 판단하고 있으며, 보정요소로 인한 비용모델의 설명력 향상도 크지 않은 것으로 보고되고 있다.

기능점수모형에 위와 같은 몇 가지 문제점이 있긴 하지만 기능점수모형은 지금까지 개발된 소프트웨어의 규모를 측정하는 방식 중에 가장 합리적인 모형으로 국제표준으로 선정되어 해외 선진국에서 널리 활용되고 있다. 이러한 추세를 감안하여



〈그림 3〉 소프트웨어사업대가기준 적용 절차([5])

국내의 소프트웨어사업대가기준을 기능점수방식으로 개선하는 작업이 진행중이다.

소프트웨어사업대가기준[4]은 공공기관이 국내에서 발주하는 소프트웨어개발, 유지보수, 데이터구축 등 소프트웨어사업의 비용을 산정하기 위한 기준으로 1989년 제정된 이래로 공공부문뿐만 아니라 민간부문 소프트웨어개발사업에도 적용되어 왔다([4]). 소프트웨어사업대가기준에 명시된 소프트웨어개발비 계산절차를 정리하면 <그림 3>과 같다.

소프트웨어사업대가기준을 적용하기 위해선 먼저 소프트웨어의 규모를 산정해야 한다. 현재는 스텝수(라인수)방식으로 소프트웨어 규모를 산정하지만, 개정될 기준은 기능점수방식을 사용하게 될 것이다. 소프트웨어 규모가 산정되면 언어와 수행공정에 따라 기초인건비가 계산되고, 여기에 규모, 프로젝트형태, 적용기종에 따른 보정을 통해 직접인건비가 계산된다. 제경비 및 기술료는 직접인건비에 일정비율을 곱하여 계산하고 직접경비는 프로젝트에 직접 사용되는 경비들을 말한다. 직접인건비, 제경비, 기술료, 직접경비를 모두 합하여 소프트웨어 개발비가 계산된다. 소프트웨어사업대가기준에는 기능점수당 단가와 각종 보정치도 함께 명시되므로, 소프트웨어 규모인 기능점수값이 결국 프로젝트 비용을 결정하게 된다. 따라서, 사업비를 적정하게 산정하기 위해서는 기능점수값의 정확한 예측이 필요하다.

3. 기능점수 예측 방법

본 장에서는 기존에 제시된 기능점수예측방법을 알아본다. NESMA 모형, ISBSG 모형, EFP 모형은 본래 기능점수예측을 위해 고안된 방법이고, 간이기능점수모형이나 프로그램 본수를 이용한 방법은 본래 사업비 예측을 위한 방법이었으나 본 연구에서는 기능점수예측에 적용하고자 한다.

3.1 NESMA 모형[26]

NESMA에서 제시한 기능점수예측방법은 크게 두 가지이다. 첫 번째 방법은 기능유형별로 기능의

개수는 도출될 수 있으나 RET/DET 또는 FTR /DET을 도출할 수 없어 복잡도 산출이 어려운 경우에 적용될 수 있는 방법이다. 이 방법에서는 내부논리파일과 외부연계파일의 복잡도를 '낮음'으로 두고, 외부입력, 외부출력 및 외부조회의 복잡도는 '보통'으로 설정하여 기능점수를 예측한다. 두 번째 방법은 기능유형별 기능의 개수조차 도출되기 어려운 경우 데이터기능만으로 전체 기능점수를 예측하는 방법이다. 이 방법은 하나의 내부논리파일에 평균적으로 외부입력 3개, 외부출력 2개, 외부조회 1개가 존재하며, 하나의 외부연계파일에 평균적으로 외부출력 1개, 외부조회 1개가 존재한다는 가정에 의해 제안된 방법이다. 두 방법의 기능유형별 가중치를 정리하면 <표 2>와 같다.

〈표 2〉 NESMA 모형의 가중치

기능유형	NESMA모형의 기능유형별 기능점수	
	방법 1	방법 2
내부논리파일	7	35
외부연계파일	5	15
외부입력	4	-
외부출력	5	-
외부조회	4	-

3.2 ISBSG 모형[13]

소프트웨어개발사업에서 사용자의 요구분석이 이루어지지 않은 사업초기단계에 모든 기능유형의 개수와 복잡도를 산출하는 것은 현실적으로 불가능하다. ISBSG는 이러한 현실을 감안하여 일부 기능유형의 기능개수만 산출된 경우에 전체 기능점수값을 예측하는 방법을 제안하였다. 예를 들어, 내부논리파일의 개수를 파악할 수 있을 경우에는 내부논리파일의 개수에 내부논리파일의 평균가중치를 곱한 다음 이를 내부논리파일이 전체 기능점수값에서 차지하는 비율로 나누어 전체 기능점수값을 예측한다. 마찬가지로 외부입력의 개수를 파악할 수 있을 경우에는 외부입력의 개수에 외부입력의 평균가중치를 곱한 다음 이를 외부입력이 전체 기능점수값에서 차지하는 비율로 나누어 전체 기능점수값을 예측한다. 전체기능점수값에서 각 기능유형이 차지하는 비율과 기능유형의 평균기능점수는 <표 3>과 같다.

〈표 3〉 ISBSG 모형의 가중치

기능유형	ISBSG 모형	
	전체기능점수에서 차지하는 비율	평균 기능점수
내부논리파일	22.1%	7.4
외부연계파일	5.0%	-
외부입력	33.5%	-
외부출력	23.5%	5.4
외부조회	16.0%	-

3.3 간이기능점수모형[2]

간이기능점수모형은 기능점수모형을 사업제안단계에 적용할 수 있게 하기 위해서, 기능점수모형을 보다 간략화한 모형이다. 간이기능점수모형은 원래 화면입력, 출력화면, 보고서출력, 내부엔티티 외부엔티티 등 5가지의 기능유형으로 구성되었으나, 배치입력과 배치출력이 추가되어 총 7개의 기능유형으로 구성된다. 기능점수모형과 간이기능점수모형의 차이점은 크게 세 가지로 요약할 수 있다. 첫 번째, 간이기능점수모형의 기능요소는 기능점수모형의 기능요소와 약간 다르다. 이는 사업초기단계에서의 획득 가능여부를 기준으로 기능요소들을 선정했기 때문이다. 두 번째, 간이기능점수 모형에서는 각 기능요소의 복잡도를 별도로 구하지 않는다. 왜냐하면, 사업제안단계에서는 내부엔티티 수준만 도출되고 분석/설계단계에서 비로소 엔티티의 속성까지 상세하게 도출되므로, 속성의 개수를 고려하여 엔티티의 복잡도를 결정하는 것은 사업초기단계에 현실적으로 무리가 있기 때문이다. 세 번째, 간이기능점수모형은 시스템 특성에 따른 보정 과정이 없다. 시스템 특성에 따른 보정이 소프트웨어 개발비 추정의 정확성을 어느 정도 향상시키지만, 향상 정도는 크지 않는 것으로 보고되고 있다

〈표 4〉 기능점수모형과 간이기능점수모형

기능점수모형(IFPUG 모형)		간이기능점수모형
데이터기능	내부논리파일(ILF)	내부엔티티(Internal Entity, IE)
	외부연계파일(EIF)	외부엔티티(External Entity, EE)
트랜잭션기능	외부입력(EI)	입력화면(Input Screen, IS) 배치입력(Batch Input, BI)
	외부출력(EO) 외부조회(EQ)	출력화면(Output Screen, OS) 보고서출력(Report Output, RO) 배치출력(Batch Output, BO)

([23]). 특히, 사업초기단계에서는 사업의 불확실성으로 인해 분석/설계 단계에 비해 요구되는 정확도가 낮다고 볼 수 있기 때문에, 시스템 특성에 따른 보정 과정이 없더라도 간이기능점수모형의 유용성이 크게 훼손되지는 않을 것이다.

기능점수모형과 간이기능점수 모형의 기능유형 차이점을 비교하면 <표 4>와 같다.

간이기능점수모형에서는 내부논리파일 대신에 내부엔티티를 사용한다. 내부엔티티는 대상 시스템에 의해 유지되며 사용자가 논리적으로 식별가능한(identifiable) 엔티티를 의미한다. 마찬가지로, 외부엔티티는 외부시스템의 의해 유지되며 대상 시스템에 의해 참조되고, 사용자가 논리적으로 식별가능한 엔티티를 뜻한다. 화면입력은 데이터를 입력/수정/삭제하기 위해 사용되는 화면을 의미하며, 동일한 화면에 입력/수정/삭제 기능이 함께 포함되어 있으면 기능점수모형과 같이 각각을 하나의 화면입력으로 간주한다. 화면출력은 데이터를 검색/조회/출력하는데 사용되는 화면으로, 입력기능과 조회기능이 한 화면에 구현되어 있으면, 각각 화면입력과 화면출력으로 산정한다. 또한, 검색 조건을 입력하는 화면과 검색 결과를 출력하는 화면이 별도의 화면으로 구분되어 있더라도 이 두 개를 합쳐 하나의 화면출력으로 산정한다. 보고서출력은 데이터에 대한 조회/처리 결과를 인쇄하는 보고서를 의미한다. 배치입력과 배치출력은 배치작업에 의해 수행된 입력과 출력작업을 말한다.

기능점수모형의 외부출력과 외부조회는 사용자에게 데이터를 전달한다는 점에서 공통점을 가지고 있으나, 유도된 데이터를 포함하거나 수학적 공식 또는 데이터의 변경을 포함하면 외부출력이고 그렇지 않은 경우에는 외부조회로 분류된다. 그러나, 간이기능점수모형은 출력이 화면상에서 이루어지면 화면출력, 출력이 인쇄물로 나타나는 경우는 보고서출력으로 분류한다. 이는 사업초기단계에서 출력되는 데이터가 내부적으로 저장된 자료인지 아니면 유도되는 자료인지를 결정하기가 현실적으로 어렵다는 점에 착안한 것이다.

3.4 EFP(Early Fucntion Point) 모형

EFP는 Meli[25]에 의해 제안된 방법으로, IFPUG의 기능점수방식과 일관성을 가지면서 사업초기단계에 적용할 수 있도록 고안되었다. EFP에서는 소프트웨어 기능을 그 상세화 정도에 따라 원시기능(functional primitive), 마이크로기능(microfunction), 일반기능(function), 매크로기능(macrofunction) 등 4가지로 구분한다. 개발할 전체 소프트웨어 중에 상세분석이 가능한 부분은 원시기능 또는 마이크로기능 수준까지 분석하고 그렇지 않은 부분은 매크로기능 또는 일반기능 수준까지 분석할 수 있다. 즉, 요구사항을 어느 정도 수준까지 명확히 할 수 있느냐에 따라 4가지 기능유형 중 하나를 택하여 기능량을 측정한다.

원시기능은 IFPUG의 기본프로세스(elementary process)와 같은 개념으로 사용자 관점에서 의미있는 가장 세분화된 기능을 의미한다. 원시기능에는 원시입력(primitive input), 원시출력(primitive

〈표 5〉 EFP의 기능유형과 가중치

기능유형	설 명
원시기능	원시입력, 원시출력, 원시조회
마이크로기능	4개의 원시기능(입력, 수정, 삭제, 조회)
일반기능	대(19~25개의 원시기능), 중(12~18개의 원시기능), 소(11개이하의 원시기능)
매크로기능	대(8~12개의 일반기능), 중(4~7개의 일반기능), 소(3개이하의 일반기능)

〈표 6〉 기능점수당 프로그램 라인수([17])

언 어	언어수준	기능점수당 라인수	언 어	언어수준	기능점수당 라인수
Assembly	1.5	213	HTML(3.0)	22.0	15
BASIC	3.0	107	JAVA	6.0	53
C	2.5	128	PowerBuilder	20.0	16
C++	6.0	53	SQL	25.0	13
DELPHI	11.0	29	Visual Basic	11.0	29

output), 원시조회(primitive inquiry) 등이 있다. 마이크로기능은 생성, 조회, 수정, 삭제 등 4개의 원시기능으로 이루어진 기능묶음을 말한다. 일반기능(function)은 11~25개 원시기능들로 이루어진 기능묶음을 말한다. 매크로기능은 3~12개의 일반기능들로 이루어진 가장 큰 단위의 기능집합을 나타낸다. <표 5>는 EFP의 기능유형과 가중치를 정리한 것이다.

또한, IFPUG는 데이터기능을 내부논리파일과 외부연계파일로 구분하는 반면에 EFP는 데이터파일을 내·외부 구분 없이 5개 척도로 그 복잡도를 평가한다.

EFP를 다양한 실제 사업에 적용하여 정확성과 사용가능성을 평가한 연구는 거의 없는 실정이다. Meli[25]에 의하면 약 15개 사업에 적용해 본 결과 오차범위가 ±25% 이내인 사업이 전체 대상사업의 75%인 것으로 조사되었다. 또한, EFP의 기능분류는 다소 주관적이며 동일한 기능유형에 포함되는 원시기능의 수에 차이가 많아 기능점수예측오차가 클 것으로 보인다.

3.5 프로그램 본수(라인수)를 이용한 기능점수 예측

프로그램 본수로부터 기능점수를 역으로 추정(backfiring)하는 방식은 일반적으로 정확성이 떨어져 기능점수값 예측에는 부적합한 방법으로 알려져 있다. 그러나, 유사한 사업의 프로그램 라인수가 알려져 있거나, 개발된 소프트웨어의 기능량을 간편하게 추정하고자 할 때 많이 사용되고 있다. PRICE-S®, KnowledgePLAN® 등 소프트웨어 비용예측을 위한 상용도구에서도 라인수와 기능점수간의 상호변환을 지원하고 있다. 특히, 국내에서는 소프트웨어사업대가기준에 의해 프로그램 본수(本數)에 따라 사업비가 결정되었기 때문에 1989년 이래로 프로그램 본수에 의해 사업규모를 산정해 왔다. 따라서, 국내의 유사한 사업의 규모로부터 기능점수를 추정하고자 하는 경우에는 프로그램 라인수 또는 본수로부터 기능점수를 추정하는 방식이 유용하게 사용될 수 있다. <표 6>은 언어수준에 따라 기능점수당 라인수 변환에 관한 예를 보여 준다.

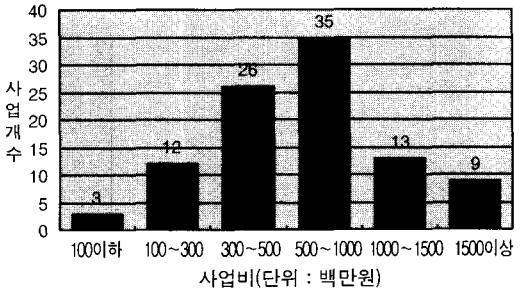
4. 국내 소프트웨어개발사업 적용 결과

본 장에서는 국내 소프트웨어사업에 관한 자료를 사용하여 여러 가지 기능점수예측방법의 정확성을 비교·분석해 본다. 정보화사업 비용자료 구축

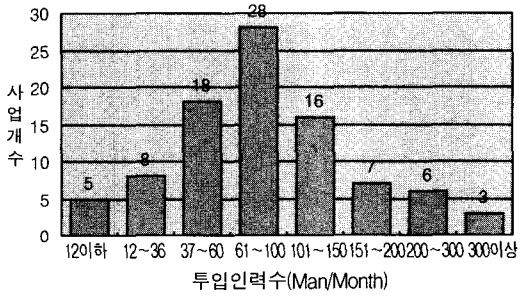
의 일환으로 2001년 이후부터 수집된 공공부분과 민간부분 소프트웨어개발사업의 구축결과물로부터 분석에 필요한 자료를 추출하였다. 자료수집을 위해 고급기술자 이상으로 약 20인/월(Man/Month)의 인력이 투입되었으며, 1994년부터 2002년 사이에 사업이 종료된 총 179개사업의 사업기간, 계약금액, 투입인력, 기능유형과 기능점수값, 프로그램 본수, 사업특성을 조사하였다. 이중 산출물이 완전하지 못하거나 정확하지 않아 조사결과의 신뢰성이 부족한 81개 사업을 제외한 총 98개 사업에 대해 기능점수예측 방법의 정확성을 비교 · 분석하였다. 기능점수는 <그림 2>에 설명된 절차에 따라 산정되었으며, 각 기능들은 [12]에 제시된 기능점수 산정매뉴얼에 따라 산정되었다. EFP 방식은 기능유형을 구분할 수 있는 정확한 기준이 없어 자료수집이 불가능하였다. 따라서, EFP 방식의 적용결과는 제외하기로 한다.

분석대상 사업 98개중 67개(68%)는 공공부문에서 수집되었고, 나머지 32개(32%)는 민간부문에서 수집되었다. 또한, 전체 사업중 73개(75%)는 사무처리용 소프트웨어이며, 13개(13%)는 지능정보형 또는 멀티미디어용 소프트웨어이고, 나머지 22개(22%)는 사무처리 · 멀티미디어 · 과학기술용 소프트웨어들이 혼합된 형태를 띠고 있다. 사업기간이 6개월미만인 사업이 14개(14%), 6개월 이상 10개월미만인 사업이 50개(51%), 10개월 이상인 사업이 34개(35%)로 나타났다. 분석대상사업의 총사업비와 투입입력 분포를 분석해 보면 <그림 4>, <그림 5>와 같다.

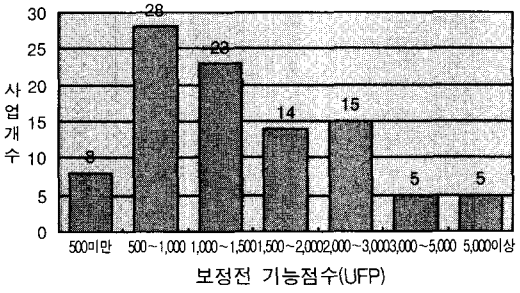
<그림 4>는 계약금액을 기준으로 사업비를 분석하였고, <그림 5>는 실제 투입된 총투입인력을 분석하였다. 각 그림의 막대위에 표시된 숫자는 해당구간에 속하는 사업개수를 나타낸다. <그림 5>의 투입인력은 인력등급을 무시한 전체 투입인력을 의미한다. 소프트웨어사업의 특성상 사업의 실제 원가를 파악하기는 현실적으로 어렵기 때문에 사업비는 계약금액을 기준으로 하였다. 사업비 평균은 7.3억원이며, 전체 사업비에서 하드웨어/소프트웨어 구입비를 제외한 소프트웨어개발이 차지하는 비중은 평균 72%이다. 평균 투입인력규모는 약 100M/M이며, 실투입인력 실적이 파악되지 않는 사업을 제외한 총 91개 사업에 대해 분석하였다.



〈그림 4〉 분석대상 사업의 사업비 분포



〈그림 5〉 분석대상 사업의 총투입인력 분포



〈그림 6〉 분석대상 사업의 보정전 기능점수(UFP) 분포

기능점수 측면에서 98개 사업의 소프트웨어 규모를 분석해 보면 <그림 6>과 같다. <그림 6>에서 알 수 있듯이 500~1,500 사이의 기능점수값을 갖는 사업이 다수를 차지하고 있다. 또한, 3,000 UFP 이상되는 대규모 소프트웨어사업도 10개 포함되어

〈표 7〉 기능유형별 평균 기능점수와 분포

기능유형	NESMA 복잡도	평균 기능점수	기능점수비율	복잡도 분포		
				낮음(Low)	보통(Average)	높음(High)
내부논리파일(ILF)	7	7.216	33.4%	94.4%	4.7%	0.9%
외부연계파일(EIF)	5	5.424	1.3%	86.3%	8.7%	5.0%
외부입력(EI)	4	3.991	28.7%	53.0%	20.9%	26.1%
외부출력(EO)	5	4.992	14.4%	44.6%	33.5%	21.9%
외부조회(EQ)	4	3.823	22.2%	54.9%	26.5%	18.6%

있다. 전체 98개 사업의 평균 보정전 기능점수값은 1,812이다.

각 기능유형별 복잡도의 분포와 평균 복잡도를 분석해 보면 <표 7>과 같다. 비율은 전체 기능점수값에서 각 기능유형이 차지하는 비율을 의미한다. <표 7>에서 알 수 있듯이 국내 소프트웨어사업의 기능유형별 평균 기능점수와 NESMA 방법 1의 기능유형별 기능점수는 매우 유사함을 알 수 있다. 특히, 외부입력 및 외부출력의 평균기능점수는 NESMA 방법 1의 기능점수와 거의 일치하고 있어, 국내에서 개발되는 소프트웨어가 국외에서 개발되는 소프트웨어와 기능점수 구성면에서 크게 차이가 없음을 알 수 있다.

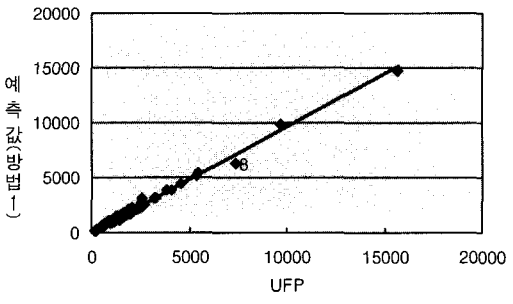
본 연구에서는 여러 가지 기능점수 예측식의 정확성을 비교하는 척도로 Conte 등[10]이 제안한 평균상대오차(Mean Magnitude Relative Error, MMRE)와 Pred(·) 통계량을 사용하였다. 평균상대오차는 다음과 같이 계산된다. 단, n은 분석대상 사업개수이다.

$$MMRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|\text{예측치} - \text{실제값}|}{\text{실제값}}$$

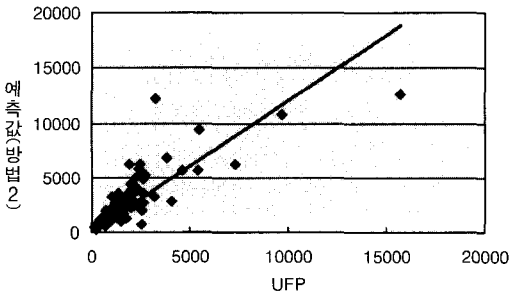
Pred(·)는 예측치와 실제값과의 상대오차가 일정범위내인 사업의 비율을 말한다. 예를 들어, Pred (0.10) = 0.75은 예측치와 실제값의 상대오차가 10 % 이내인 사업이 전체 사업의 75%임을 의미한다. MMRE가 작을수록 모형의 정확성이 높고, Pred (·)의 경우에는 값이 클수록 정확한 예측모형으로 볼 수 있다.

4.1 NESMA 모형 적용 결과

NESMA 모형을 국내 사업에 적용한 결과 실제 기능점수값과 예측된 기능점수값간의 차이를 분석해 보면 <그림 7>와 <그림 8>과 같다. NESMA 모형 중 방법 1은 거의 정확하게 기능점수값을 예측하지만, 방법 2에 의한 예측치는 상당한 오차가 있음을 알 수 있다. 이는 데이터기능의 기능점수값만으로 트랜잭션기능의 기능점수값을 예측하는 방식은 정확하지 않음을 의미하며, 그 이유는 하나의 데이터기능에 대응되는 트랜잭션기능의 수가 일정



〈그림 7〉 NESMA 모형(방법 1) 적용결과



〈그림 8〉 NESMA 모형(방법 2) 적용결과

〈표 8〉 NESMA 모형의 오차

	NESMA 모형		수정된 NESMA 모형
	방법 1	방법 2	
MMRE(오차의 표준편차)	7.9%(±5.5%)	72.1%(±55.5%)	**7.6%(±5.3%)**
Pred(0.05)	0.337	0.031	**0.398**
Pred(0.10)	0.694	0.071	**0.745**
Pred(0.15)	0.908	0.153	**1.000**

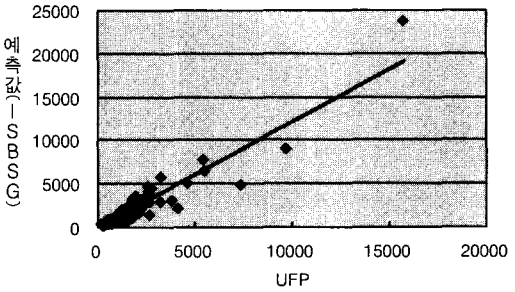
치 않기 때문이다.

<표 7>에서 나타난 바와 같이 NESMA 모형의 방법 1에서 사용하는 기능유형별 기능점수와 국내 사업의 평균 기능점수간에는 약간의 차이가 있다. 따라서, 방법 1의 기능점수 대신 국내 사업에서 도출된 평균 기능점수를 사용하면 예측값의 정확도를 보다 높일 수 있다. NESMA 모형의 방법 1을 사용하되 기능유형별 기능점수를 국내 사업에서 도출된 평균 기능점수로 대체한 수정된 NESMA 모형의 예측 오차를 분석해 보면 <표 8>과 같다. 수정된 NESMA 모형은 평균오차와 오차의 표준편차가 개선되었다. 분석대상사업에 수정된 NESMA 모형을 적용해 보면 오차범위가 ±10%이내인 사업이 전체의 74.5%로 정확도가 매우 높음을 알 수 있다. 또한, 모든 분석대상사업에 대해서도 오차범위는 최대 ±15%이내이다.

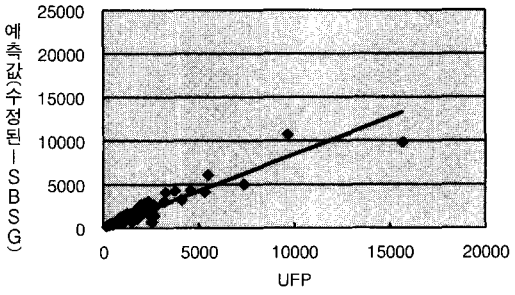
4.2 ISBSG 모형 적용 결과

ISBSG 모형은 <표 3>에서 알 수 있듯이 일부 기능유형의 기능점수값을 산정한 후에 전체 기능점수값에서 나머지 기능유형이 차지하는 평균 비율을 사용하여 나머지 기능유형의 기능점수값을 예측한다. ISBSG 모형은 데이터기능의 내부논리파일과 트랜잭션기능의 외부출력의 기능점수값을 구한 후에 나머지 기능유형의 기능점수값을 예측하고 있다.

<표 3>과 <표 7>의 기능유형별 비율을 비교해 보면 ISBSG 모형에서는 외부입력이 전체 기능유형중 가장 높은 비율을 갖고 있으나, 국내사업에서는 내부논리파일의 비중이 가장 높은 것으로 나타났으며 외부조회의 비율도 상대적으로 높게 나타났다. 이는 하나의 데이터파일에 대한 트랜잭션기능이 해외소프트웨어에 비해 적음을 의미한다. 또한, <표 7>에서 알 수 있듯이 트랜잭션기능 중에 가장 많은 비율을 차지하는 기능은 외부입력이다. 따라서, 트랜잭션기능의 외부입력을 사용하여 나머지 기능유형의 기능점수값을 추정하는 것이 기능점수 예측치의 정확도를 보다 향상시킬 수 있다. 이는 <그림 9>, <그림 10>의 예측오차 비교에서 확인할 수 있다. 여기서 수정된 ISBSG 모형은 원래의 ISBSG 모형에서 외부출력 대신에 외부입력을 사용하고, 국내사업에서 도출된 <표 7>의 기능유



〈그림 9〉 ISBSG 방법의 예측오차



〈그림 10〉 수정된 ISBSG방식의 예측오차

〈표 9〉 ISBSG 모형과 수정된 ISBSG 모형의 오차비교

	ISBSG 모형	**수정된 ISBSG 모형**
MMRE(오차의 표준편차)	30.1%(±21.7%)	**15.9%(±13.7%)**
Pred(0.05)	0.071	**0.224**
Pred(0.10)	0.184	**0.449**
Pred(0.15)	0.316	**0.602**

형별 비율과 평균 기능점수를 사용한 방식을 말한다.

ISBSG 모형과 수정된 ISBSG 모형의 평균오차와 Pred(·)값을 비교해 보면 <표 9>와 같다. 수정된 ISBSG 모형은 원래의 ISBSG 모형에 비해 오차가 크게 감소하였고, 오차의 표준편차, Pred(·)면에서도 크게 개선되었다. 그러나, <표 8>과 비교해 보면 수정된 ISBSG 모형이 NESMA 모형에 비해 오차가 큰 것으로 나타났다.

4.3 간이기능점수모형 적용 결과

간이기능점수모형은 <표 4>에서 볼 수 있듯이, 데이터기능에는 내부엔티티와 외부엔티티가 있으며, 이는 IFPUG 모형의 내부논리파일 및 외부연계파일과 일대일로 대응된다. 따라서, 내부엔티티와 외부엔티티의 개수가 주어지면, <표 7>에 제시된 내부논리파일과 외부연계파일의 평균기능점수를 사용하여 데이터기능의 기능점수값을 예측하면 된다. 반면, 트랜잭션기능의 기능점수를 구하기 위해서는 5가지 기능유형을 고려해야 한다. 그리고, 간이기능점수모형의 기능유형이 IFPUG 모형의 기능유형과 일대일로 대응되지 않는다. 예를 들어, 화면입력내에 있는 코드조회 기능을 IFPUG는 별도의 외부조회로 구분하지만 간이기능점수방식에서는 이를 구분하지 않고 입력화면 하나로만 간주한다. 이는 사업초기단계에는 화면에 있는 모든 세부기능을 식별하기 어렵기 때문이다. 따라서, 간이기능점수모형으로부터 IFPUG 모형의 트랜잭션 기능점수값을 예측하기 위해서는 두 모형의 데이터로부터 회귀식을 도출해야 한다.

트랜잭션기능을 크게 입력기능점수값과 출력기능점수값으로 구분할 수 있고, 입력기능에는 화면입력과 배치입력이 있고, 출력기능에는 화면출력, 보고서출력, 배치출력이 있다. 따라서, 다음과 같이 트랜잭션기능을 예측하는 회귀식을 구성할 수 있다.

$$\text{입력 기능점수값} = a \times \text{화면입력} + b \times \text{배치입력} \quad \text{(회귀식 1)}$$

$$\text{출력 기능점수값} = c \times \text{화면출력} + d \times (\text{보고서출력} + \text{배치출력}) \quad \text{(회귀식 2)}$$

(회귀식 2)에서 보고서출력과 배치출력은 작업수행시간이나 빈도가 다를 뿐 거의 동일한 성격의 기능을 수행하는 경우가 대부분이므로 두 기능의 회귀식 계수는 동일하다고 간주하였다. 통계분석도구를 사용하여 (회귀식 1)과 (회귀식 2)의 계수를 구하면 <표 10>과 같다.

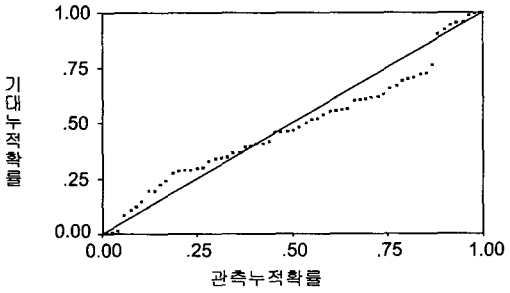
<표 10>에서 두 회귀식 모두 상관계수(R^2)가 0.947, 967로 매우 큰 것으로 나타났다. <표 10>에 제시된 회귀식이 중회귀모형의 기본가정을 만족하는가를 검사해 볼 필요가 있다. 먼저 독립변수간의 상관관계을 의미하는 다중공선성(multicollinearity)가 존재하는가를 확인하기 위해 공선성 통계량을 보면, 공차한계(tolerance)가 0.953, 0.953, 0.520, 0.520으로 (회귀식 1)과 (회귀식 2) 모두 다중공선성은 낮다고 볼 수 있다. 두 번째로, 잔차간에 자기상관성(autocollelation)이 존재하는가를 알아보기 위해 Durbin-Waton 테스트의 검정통계량을 보면, 그 값이 (회귀식 1)은 1.550, (회귀식 2)는 1.607로 임계치 1.54보다 크므로 자기상관성이 존재한다고 볼 수 없다. 세 번째로, 잔차가 정규분포를 따르는

〈표 10〉 간이기능점수 vs. UFP 회귀분석 결과

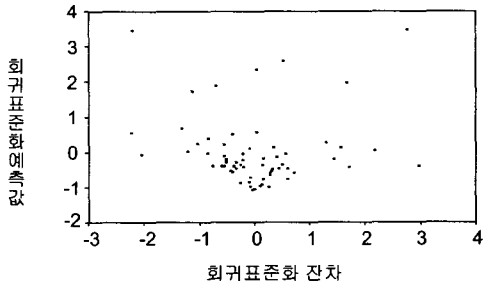
회귀식	계수	계수값	계수값의 신뢰구간(95%)		t값 (유의확률)	공차한계	Durbin-Waston	R^2
			하 한	상 한				
1	a	4.325	4.051	4.599	31.573 (0.000)	0.953	1.650	0.947
	b	5.317	1.535	9.098	2.811 (0.007)	0.953		
2	c	3.520	3.204	3.835	22.275 (0.000)	0.520	1.607	0.967
	d	5.769	4.666	6.872	10.460 (0.000)	0.520		

가를 알아보기 위해 잔차의 히스토그램과 산포도를 보면 <그림 11>, <그림 13>과 같으므로 잔차가 정규분포를 따라야 한다는 기본가정에 어긋난다고 볼 수 없다. 마지막으로, 종속변수 오차항의 분산이 모든 독립변수의 값에 동일해야 한다는 등분산성(homoscedasticity) 가정이 성립하는가를 알아보기 위해 잔차의 예측 산포도를 보면 <그림 12>과 <그림 14>와 같다. 특정한 패턴이 없으므로 등분산성 가정을 충족시킨다고 볼 수 있다.

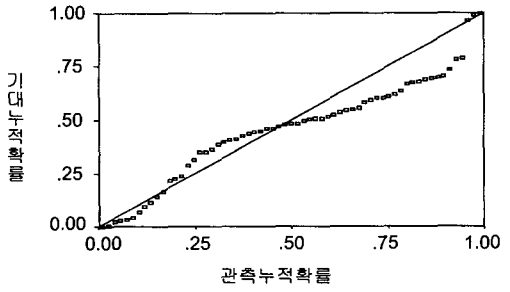
<표 10>의 회귀식 계수가 의미하는 바를 알아보기 회계식의 계수를 <표 1>과 <표 7>에 나타난 (평균)기능점수와 비교해 볼 필요가 있다. 화면입력과 배치입력의 계수는 외부입력의 기능점수 범위(3~6)내에 있고, 화면출력, 보고서출력, 배치출력의 계수값도 마찬가지로 외부출력과 외부조회의 기능점수 범위(3~7)내에 들어간다. 일반적으로 배치입력이 화면입력에 비해 복잡하므로 배치입력의 기능점수가 화면입력의 기능점수보다 높게 나타났



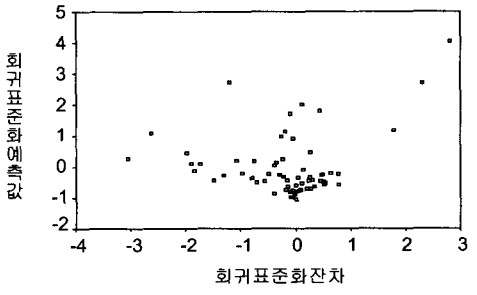
〈그림 11〉 잔차의 정규분포산포도(회귀식 1)



〈그림 12〉 잔차의 예측 산포도(회귀식 1)



〈그림 13〉 잔차의 정규분포산포도(회귀식 2)



〈그림 14〉 잔차의 예측산포도(회귀식 2)

다. 또한, 간이기능점수모형에서는 입력화면내에 있는 여러 개의 부분입력기능은 모두 묶어 하나의 화면입력으로 간주하므로 간이기능점수모형의 입력기능 개수가 IFPUG 모형의 외부입력 개수보다 작게 된다. 따라서, 입력화면과 배치입력의 회귀식 계수가 <표 7>의 평균기능점수값보다 커지게 된다. 회귀식 2에서 복잡한 출력기능은 대부분 보고서출력과 배치출력이고, 화면출력은 간단한 조회인 경우가 많으므로, 화면출력의 계수가 보고서·배치출력의 계수보다 작게 나타났다.

아래 식과 같이 데이터기능은 <표 7>의 평균기능점수를 사용하고 트랜잭션기능은 <표 10>의 회귀식을 이용하여 기능점수값을 예측한 후, 그 오차를 분석한 결과가 <표 11>에 나타나 있다. 간이기능점수모형의 예측오차는 수정된 NESMA 모형보다는 크지만, 수정된 ISBSG 모형보다는 정확하게 기능점수값을 예측할 수 있다.

기능점수 예측값
= 7.216×내부엔티티 + 5.424×외부엔티티
+ 4.325×화면입력 + 5.317×배치입력
+ 3.520×화면출력 + 5.769×보고서출력
+ 5.769×배치출력

〈표 11〉 간이기능점수를 이용한 기능점수값 예측 오차

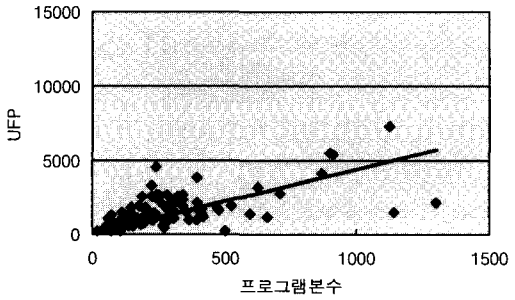
구 분	기능점수값 예측 오차
MMRE(오차의 표준편차)	11.9%(±15.8%)
Pred(0.05)	0.369
Pred(0.10)	0.600
Pred(0.15)	0.753

4.4 프로그램 본수를 이용한 기능점수 추정 결과

각 사업의 프로그램 본수를 조사하여 기능점수값과의 상관관계를 분석하였다. 프로그램 본수는 각 사업의 완료보고시 산출한 본수를 기준으로 하였다. 그러나, 프로그램 본수를 산정할 때 입력·수정·삭제기능이 하나의 프로그램내에 있을 경우 이를 하나의 프로그램으로 볼 것인지 아니면 각각의 기능에 따라 세 개의 프로그램으로 간주할 것인지를 정해야 프로그램 본수 산정의 오차를 줄일 수 있다. 본 연구에서는 위와 같은 경우 소프트웨어사업대가기준 해설서에 제시된 대로 세 개의 프로그램으로 산정하였다.

프로그램 본수와 보정전 기능점수값과의 상관관계를 알아보기 위해 먼저 두 데이터간의 산포도를 그리면 <그림 15>와 같다. 그림에서 알 수 있듯이 프로그램 본수와 UFP는 선형관계가 있으며, 회귀분석을 통해 프로그램 본수와 UFP와의 관계식의 계수를 구한 결과는 <표 11>과 같다.

보정전 기능점수값(UFP) = f × 프로그램 본수



〈그림 15〉 프로그램본수 vs. UFP의 산포도

<표 12>에 나타난 회귀식의 계수값은 4.40이다. 즉, 프로그램 1본이 기능점수 4.4에 해당함을 의미한다. <표 7>에 제시된 기능유형별 평균기능점수를 보면 프로그램 1본이 평균적으로 외부출력 또는 외부입력에 해당됨을 알 수 있다.

〈표 12〉 프로그램본수 vs. UFP 회귀분석 결과

계 수	계수값	계수의 신뢰구간(95%)		t 값 (유의확률)	R^2
		하 한	상 한		
f	4.40	3.839	4.961	15.78(0.000)	0.727

프로그램 본수를 이용한 기능점수값 예측 오차를 분석해 보면 <표 13>과 같다. 앞서 제시된 다른 방법에 비해 예측오차가 큼을 알 수 있다.

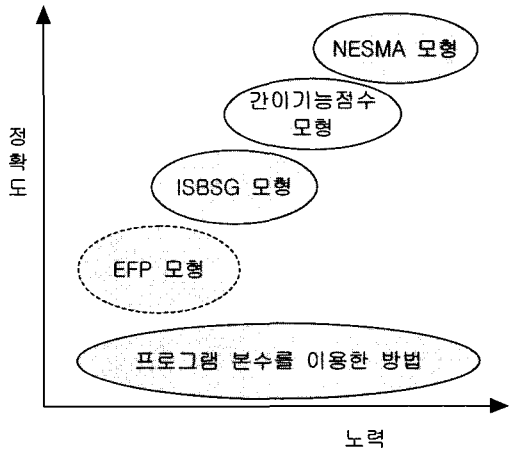
〈표 13〉 프로그램 본수를 이용한 기능점수값 예측 오차

구 분	기능점수값 예측 오차
MMRE(오차의 표준편차)	55.1%(±89.7%)
Pred(0.05)	0.071
Pred(0.10)	0.122
Pred(0.15)	0.183

5. 기능점수 예측모형 선정

3장과 4장에서는 여러 가지 기능점수예측모형과 국내사업에 적용한 결과들이 제시되었다. 본 장에서는 이러한 여러 가지 기능점수모형의 장단점을 정확도와 소요시간 측면에서 비교함으로써 적절한 기능점수예측모형을 선정하는 기준을 제시하고자 한다.

기능점수예측모형의 선정에 있어 가장 중요하게 고려할 것은 예측모형의 정확도와 예측모형에 필요한 입력자료를 얻는데 필요한 노력이다. 따라서, 지금까지 제시된 기능점수예측모형을 예측의 정확도와 모형을 적용하는데 필요한 노력을 기준으로 분류하면 <그림 16>과 같다. <그림 16>에서 NESMA, ISBSG 모형은 모두 수정된 NESMA모형, 수정된 ISBSG 모형을 의미한다. <그림 16>에서 정확도는 4장에서 제시된 각 방법의 MMRE을 참고로 하여 MMRE가 낮은 방법일수록 정확도는 높다고 판단하였다. 수정된 NESMA 모형의 MMRE는 7.6%(<표 13>), 간이기능점수모형의 MMRE는 11.9%(<표 11>), 수정된 ISBSG모형의 MMRE는 15.9%(<표 9>)이며, 프로그램 본수를 이용한 방법의 MMRE는 55.1%(<표 8>)이다. EFP 모형은 Meli [25]에 제시된 적용결과를 참고로 정확도를 추정한 것이다. 또한, 각 모형을 적용하는데 필요한 노력에 대한 정량적인 비교는 현실적으로 어려우므로, 3장에 제시되었듯이 각 모형이 필요로 하는 입력자료의 복잡도를 기준으로 모형적용에 필요한 노력을 개략적으로 추정하였다. NESMA 모형을 적용하기 위해서는 데이터 및 트랜잭션의 모든 기능유형을 식별해야 하므로 기능유형의 내부로직까지 상세하게 파악해야 한다. 따라서, NESMA 모형은 적용에 가장 많은 노력을 필요로 하는 모형이다. 반면, 간이기능점수모형은 상세한 내부로직을 파악할 필요 없이 입출력형태만을 기준으로 기능유형을 식별하기 때문에 NESMA 모형보다는 적용이 수월하다고 할 수 있다. ISBSG 모형은 일부 기능유형의 기능점수만으로 전체기능점수를 추정하기 때문에 NESMA, 간이기능점수보다는 상대적으로 적용이 쉽다고 할 수 있다. EFP 모형은 여러 가지 기능유형들의 묶음을 기준으로 기능점수를 추정하므로 적용이 가장 간편하다고 할 수 있다. 마지막으로, 프로그램 본수를 이용한 방법은 유사한 소프트웨어의 프로그램 본수가 주어진 상황에서는 기능점수를 쉽게 추정할 수 있는 방법이지만, 그렇지 않은 경우에는 전체 기능유형을 식별하는 것과 마찬가지의 노력이 요구되는 방법이라 할 수 있다.



〈그림 16〉 기능점수 예측모형 비교

정확도면에서는 수정된 NESMA 모형이 가장 바람직하지만 NESMA 모형을 사용하기 위해서는 모든 기능유형들을 도출해야 하므로 가장 많은 노력이 요구된다는 단점이 있다. 이에 비해 간이기능점수모형은 NESMA 모형에 정확도가 약간 떨어지지만, 모형적용에 요구되는 노력이 적다는 장점을 가지고 있다. 대체로 십억이상의 대규모 소프트웨어

개발사업은 NESMA 모형이나 간이기능점수모형을 사용하여 기능점수규모를 예측하는 것이 바람직할 것이다. NESMA 모형 또는 간이기능점수모형을 사용하는 경우 기능점수예측오차는 평균 10% 정도가 된다.

ISBSG 모형, EFP 모형, 프로그램 본수를 이용한 방법들은 평균오차가 15% 이상이고 오차의 분산도 매우 크기 때문에 소규모사업이나 예측시간에 제약이 있는 경우에 사용하는 것이 바람직하다. 특히, 프로그램 본수를 이용한 방법은 유사한 기존사업으로부터 프로그램 본수 산정이 가능한 경우나 이미 완료된 사업의 기능점수 규모를 평가하고자 하는 경우에 사용될 수 있는 기능점수예측방법이다. 그렇지 않은 사업의 경우에는 프로그램 본수를 산정하는 노력은 기능점수규모를 정확하게 산정하는 노력과 거의 비슷해지며 투입되는 노력에 비해 기능점수예측 정확도는 매우 낮다는 점에 유의해야 한다.

6. 결 론

본 연구에서는 소프트웨어개발비용의 정확한 산정을 위해 해외에서 널리 사용되고 있는 기능점수모형의 국내 적용을 모색하였다. 국내에서는 소프트웨어사업대가기준에 의해 기능점수값으로부터 사업비용이 결정되기 때문에 사업초기에 기능점수값을 예측할 수 있는 방법을 제시하였다. 국내 소프트웨어사업에서 수집된 자료를 기초로 기능점수예측방법의 정확성을 비교하여 적절한 기능점수예측방법을 선정할 수 있는 기준을 제시하였다.

소프트웨어개발비용 예측은 국가마다의 고유한 특성이 있어 해외에서 개발된 방법이 국내사업에 잘 적용되지 않는 경우가 많다. 본 연구결과에서 확인할 수 있듯이 해외모형을 국내사업 데이터에 맞게 수정함으로써 보다 정확한 기능점수예측방법을 구할 수 있었다. 향후 소프트웨어개발비용과 관련된 선진 연구결과를 국내 사업에 맞게 수정·발전시키는 연구가 지속적으로 필요할 것이다.

참 고 문 헌

[1] 김현수, "기능점수를 이용한 소프트웨어 규모 및 비용산정 방안에 관한 연구", 「경영과학」, 제14권, 제1호(1997. 5), pp.131-149.

[2] 박찬규, 구자환, 김성희, 신수정, 송병선, "공공부문 정보화사업의 소프트웨어 개발비용 예측에 관한 연구", 「경영과학」, 제19권, 제2호(2002), pp.191-204.

[3] 이양규, "기능점수모형을 이용한 소프트웨어 개발비용 산정", 「경영연구」, 제6권(1997), pp. 241-261.

[4] 정보통신부, http://www.mic.go.kr, 2003.

[5] 한국소프웨어산업협회, 「S/W사업대가기준 개선사업」, 2001.

[6] 한국전산원, 「국가정보화백서 2002」, 2002.

[7] Albrecht, Allan J., "Measuring application development productivity," *Proc. IBM Application Deveoplment Symp.*, GUIDE Int. and SHARE Inc., IBM Corp., Monterey, CA, (Oct. 1979), pp.83-92.

[8] Boehm, B.W. et al., *Software cost estimation with COCOMO II*, Prentice Hall PTR, 2000.

[9] Common Software Measurement International Consortium, *Measurement Manual (The COSMIC implementation guide for ISO/IEC 19761 : 2003)*, Version 2.2, 2003.

[10] Conte S.D. et al., *Software Engineering Metrics and Models*, Benjamin/Cummings Publishing Company, Inc., 1986.

[11] Furey, S., "Why we should use function points," *IEEE Software*, Vol.4, No.3(1997), p.28.

[12] IFPUG, *Function Point Counting Practices Manual (Release 4.1.1)*, International Function Point Users Group, 2000.

[13] International Software Benchmarking Stan-

dards Group, *Practical project estimation : A tool kit for estimating software development effort and duration*, ISBSG, 2001.

[14] Jeffery, D.R., G.C. Low, M. Barnes, "A comparison of function point counting techniques," *IEEE Transactions on Software Engineering*, Vol.19, No.5(1993), pp.529-531.

[15] Jones, T. Capers, "A short history of function points and feature points," *Software Productivity Research Inc.*, 1988.

[16] Jones, T. Capers, *Estimating Software Costs*, McGraw-Hill, 1998.

[17] Jones, T. Capers, "Programming Languages Table," Release 8.2, Software Productivity Research, 1996.

[18] Kemerer, C.F., B.S. Porter, "Improving the reliability of function point measurement : An empirical study," *IEEE Transactions on Software Engineering*, Vol.18, No.11 (1992), pp.1011-1024

[19] Kitchenham, B., Kari Kansala, "Inter-item correlations among function points," *Proceedings of the 15th International Conference on Software Engineering*, Baltimore, MD, CA, USA, (May 1993), pp.477-480.

[20] Kitchenham, B., "The problem with function points," *IEEE Software*, Vol.4, No.3 (1997), p.29.

[21] Kumar, R., "Object-based estimation of software development effort : An investigation in ICASE environments," *Graduate School of Business Administration*, New York University, 1994.

[22] Lokan, Chris J., "Empirical study of the correlations between function point elements," *Proceedings of the 1999 6th International Software Metrics Symposium*, (Nov. 1999), pp.200-206.

[23] Low, G.C., D.R. Jeffery, "Function points in the estimation and evaluation of the software process," *IEEE Transactions on Software Engineering*, Vol.16, No.1(1990), pp. 64-71.

[24] Matson, J.E., B.E. Barrett, J.M. Mellichamp, "Software development cost estimation using function points," *IEEE Transactions on Software Engineering*, Vol.20, No.4(1994), pp.275-287

[25] Meli, R., "Early and extended function point : a new method for function points estimation," *IFPUG Annual Conference*, Scottsdale, Arizona, USA, September 1997.

[26] NESMA, http://www.nesma.nl.

[27] Orr, G., T.E. Reeves, "Function point counting : one program's experience," *Journal of Systems and Software*, Vol.53. No.3(2000), pp.239-244.

[28] Pressman, R.S., *Software engineering : A practitioner's approach*, 4th ed., McGraw-Hill, 1997.

[29] Putnam L., W. Myers, *Measures for excellence*, Yourdon Press, 1992.

[30] Symons, C.R., "Function point analysis : difficulties and improvements," *IEEE Transactions on Software Engineering*, Vol.14, No.1(1988), pp.2-11.

[31] Vidger, M.R., A.W. Kark, *Software cost estimation and control*, Institute for Information Technology, National Research Council Canada, 1994.