논문 2020-1-4 http://dx.doi.org/10.29056/jsav.2020.06.04

소프트웨어 개발비 감정을 위한 유스케이스 점수 추정

권기태*†

Use Case Points Estimation for the Software Cost Appraisal

Ki-Tae Kwon*†

요 약

소프트웨어 개발비 감정은 프로그램 완성도 감정과 함께 소프트웨어 공학 방법론을 적용하고 있다. 특히 소프트웨어 비용산정 기법을 적극적으로 준용해왔다. 다수의 감정 사례에서 소프트웨어 개발비 감정을 위해 소프트웨어 비용산정에 기반을 두는 “SW사업 대가산정 가이드”를 참조하여 감정이 이루어져 왔으나, 이러한 방법은 본질적인 한계를 가진다. 개발비 감정을 위한 “SW사업 대가산정 가이드” 자체의 문제점과 함께 소프트웨어 규모 산정의 기본이 되는 기능점수가 가지는 단점으로 인해 감정의 정확성과 일관성이 유지되기 어렵다. 본 연구에서는 규모추정의 정확성과 일관성 유지를 위한 방안으로 유스케이스 기반의 규모 추정 방안을 제시한다. 평가 대상 프로젝트는 개발비 감정 사례들과 유사한 유형의 소프트웨어공학 교과목의 프로젝트로 진행하였으며, 공수 추정 시에 감정 사례들의 상황과 유사하도록 제공되는 문서와 정보를 최소화하였다. 기능점수 기반의 기존 소프트웨어 개발비 산정 방식과 유스케이스 기반으로 제안한 방안의 성능 평가를 실시한 결과, 기존 방식보다 정확도가 향상되었고 통계적으로 유의함이 입증되었다.

Abstract

The software development cost appraisal is treated as a part of the program completion appraisal, and the software engineering methodology is applied. In particular, software cost estimation techniques have been actively applied. For more information about the software development costs calculation, we can refer to the "SW cost estimation guide". Although successful appraisal of a number of development costs based on the guide has been processed, but a number of cases requiring discussion of appraisal results have been discovered. In this study, we propose a use case-based size estimation method to maintain the accuracy and consistency of size estimation. As a result of performing performance evaluation of the proposed method in an environment similar to the development cost appraisal case, it was proved that the accuracy was improved over the existing function points method.

한글키워드 : 완성도 감정, 대가산정, 개발비 감정, 기능점수, 유스케이스 점수

keywords : completeness evaluation, cost estimation, development costs evaluation, function points, use case points

1. 서 론

컴퓨터 프로그램 저작물 감정은 유사도 감정, 완성도 감정, 개발비용 감정의 세 가지 분야로 나누어진다. 특히 이 세 가지 유형 중에서 완성된 프로그램의 적정 개발비용을 감정하거나 미완

* 강릉원주대학교 컴퓨터공학과
† 교신저자: 권기태(email: ktkwon@gwnu.ac.kr)
접수일자: 2020.05.21. 심사완료: 2020.06.10.
게재확정: 2020.06.19.

성된 프로그램이 적절하게 완성되었을 경우의 적정 개발비용을 추정하는 것이 개발비 감정이다. 여기에서는 주로 소프트웨어 공학 분야의 초기 단계에서부터 연구되어 오고, 다수의 상용 비용산정 모델을 가지고 있는 비용산정 기법을 적용하는 것이 일반적이다[1]. 그러나 소프트웨어공학의 비용산정은 엄밀하게 말해서 개발 전 혹은 개발 초기 단계에서 개발 비용을 추정하는 것이다. 이에 반해 개발비용 감정은 완성된 프로그램의 적정 개발 비용을 감정하거나, 여러 가지 원인으로 완성되지 못한 프로그램의 적정 개발비용을 감정하는 것이 일반적이다. 또한 완성된 프로그램이라 하더라도 발주자의 요구조건을 만족하지 못하는 프로그램의 경우, 특정 프로그램의 개발비를 추정하는 것이 목적으로 소프트웨어공학의 비용산정과는 근본적으로 관점의 차이가 존재한다. 즉, 개발비 감정은 주로 분쟁 대상의 프로그램의 적정 개발 비용을 추정하는 것을 근본적인 목적으로 한다.

소프트웨어 공학에서는 이상적인 개발환경, 즉 충분한 요구사항이 제시되고 표준 개발 프로세스에 따라 진행되는 프로젝트를 전제로 하지만, 분장 대상 프로그램이 주된 대상이 되는 개발비 감정은 대부분의 경우 감정의 근거가 되는 요구사항이 존재하지 않거나 부정확하다.

한국저작권위원회에서 발표한 “완성도 감정 가이드라인”은 완성도와 개발비 감정에 적용하는 절차 및 기법을 제시하고 있다[2]. 여기에서 개발비 감정은 기능점수와 투입공수 방식으로 나누어 제시하고 있으며, 상세한 절차는 “소프트웨어 대가산정 가이드[3]”를 준용하도록 한다. 기능점수는 소프트웨어 규모의 국제 표준으로, 상기 가이드에서는 기능점수로 추정한 소프트웨어의 크기를 단가를 곱하는 방법을 적용한다. 즉, 아파트 분양가를 산정할 때 평당 분양가에 전체 면적을 곱하는 것과 동일한 방식이다. 그러나 동일 면적의 아파트라도, 설계 특화, 자재, 인테리어, 커뮤니티 설비에 따라 번체 분양가가 조정되는 것과 같은 원리로 소프트웨어의 전체 크기인 총 기능점수에 단가를 단순하게 곱하는 것 외에 소프트웨어의 특성에 따라 총 개발비를 보정하는 방식이다. 보정은 다양한 특성에 따라 보정계수를 곱한다. 기능점수는 ISO 국제 표준의 소프트웨어 규모 척도로 산정 절차는 IFPUG의 CPM 문서로 정의되어 있고, 국내 공공사업에서는 “국제표준 기반 기능점수 산정 안내서[4]”를 주로 참조한다. 또 한 가지 개발비 감정 방식으로 준용되는 투입공수 방식은 이전에 수행한 유사 프로젝트에 투입된 공수를 근거로 개발비용을 추정하는 방식으로, 그동안 제한적으로 사용되어 왔지만, 최근 공공분야의 소프트웨어 개발 비용산정에서는 적용하지 않는 것으로 의무화되어 있다.

그동안 여러 개발비 감정 사례에서 완성도 감정 가이드라인에서 제시한 것처럼 한소협 발간 “소프트웨어 대가산정 가이드”를 근거로 감정이 실시되었다. 그러나 대가선정 가이드와 기능점수 자체가 문제점을 가지고 있는 것은 물론, 상기 가이드의 기초가 되는 기능점수는 이상적인 요구분석을 전제로 하고 있는 개발 프로세스 초기에 소프트웨어 개발 비용추정을 목적으로 하는 반면, 개발비 감정은 분쟁 대상이 되는 프로그램을 대상으로 수발주자가 동의하는 개관적인 요구분석이 실행되지 않은 본질적인 한계를 가지고 있다. 본 논문은 개발비 감정 시 기능점수 기반의 대가산정 가이드를 준용할 때 일어나는 문제를 해결하기 위해 유스케이스 기반의 규모 추정 방안을 제안하고자 한다. 본 논문의 구성은 다음과 같다. 1장은 서론으로 연구방향을 제시하고, 2장에서 기존의 개발비용 감정의 기초가 되었던 대가가정 가이드의 한계를 제시한다. 3장에서 유스케이스 기반의 규모 추정 방안을 제안한다. 4장에서는 기존 기능점수 기반의 방안과 비교한 성

능 평가 결과를 정리한다. 마지막으로 5장에서 결론과 함께 문제점을 제시한다.

2. 기존 대가산정 가이드의 문제점

2.1 소프트웨어 대가산정 가이드 정리

소프트웨어 대가산정 가이드의 목적은 공공기관이 소프트웨어를 개발할 때 전체 생명주기 단계에서 적정 예산을 수립하고 프로젝트 발주 시에 적절한 대가를 산정할 때 지침으로 활용하는 것이다[3].

적절한 개발비용 추정은 소프트웨어 전체 생명주기에서 정형화된 프로세스 활동으로 반복적으로 진행되지만, 수발주자를 포함한 다양한 이해당사자들에게 영향을 미치고 있다.

이 가이드의 목적은 그동안의 개발현장에서 늘 제시되었던 적정 개발비용 보장을 지원하기 위한 객관적인 기준을 제시하기 위한 것이다. 특히 민간부문과는 달리 공공사업의 경우 수주자가 적정 개발비용을 받지 못한다는 불만을 해소하여 적절한 비용을 보장함으로써 소프트웨어 품질 향상뿐만 아니라 소프트웨어 개발업체의 수익 향상, 궁극적으로는 국가 경쟁력 향상을 목표로 하고 있다.

2012년 이전까지 공공분야의 소프트웨어 개발비용 추정의 근거가 되었던 소프트웨어 그리고 엔지니어링 사업대가 기준은 폐지되고, 한소협에서 발표한 “소프트웨어 대가산정 가이드”가 공공기관의 소프트웨어 개발 시에 적정 개발비용 산정을 위한 기준으로 적용되어 왔다.

상기 가이드에서는 ISO 국제 표준의 기능점수 기반의 개발비용 추정방안을 제시하고 있으나, 이 부분은 과거의 대가기준을 그대로 유지하고 있는 것으로 큰 변화는 없다. 다만, 세부적인 산정 기법의 설명과 사례 제시는 IFPUG의 CPM[7]을 기반으로 기존의 대가기준보다는 공공기관의 수수발주 업무를 담당하는 이해당사자에게 유용한 정보를 제공하고 있다. 또한 기존 대가기준에서 제시하지 않았던 기획단계와 운영단계의 대가산정 전 과정에 관한 비용추정 방안을 사례 제시와 함께 알기 쉽게 설명하여, 공공기관 수발주자가 편리하게 이용할 수 있게 하였다. 또한, 2012년 발표 이후 지속적으로 보정계수를 수정하여 정확도 향상에 노력해 오고 있다.

2.2 대가 산정 가이드 적용의 문제점

상기 대가산정 가이드에서 소프트웨어를 개발하는데 소요되는 비용은 프로세스 수행에 필요한 원가, 직접경비, 이윤으로 그림 1의 산정 절차를 따른다.

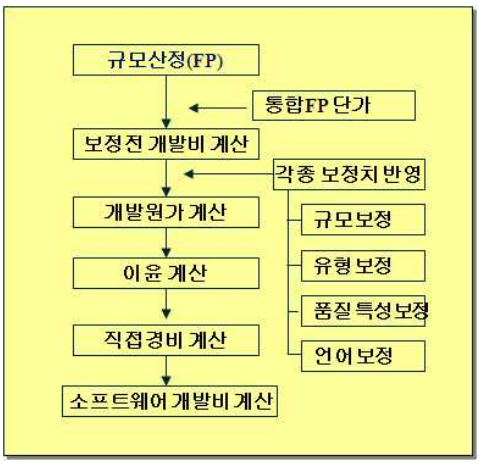

그림 1. 소프트웨어 비용 추정 프로세스
Fig 1. Process of development cost

앞에서 비유한 것처럼, 아파트의 분양가를 평당 분양가에 평을 기준으로 총 면적을 곱하여 산정할 때, 아파트의 크기인 평(坪)에 해당하는 것이 기능점수이다. 즉, 소프트웨어 개발비용의 가

장 기본적인 규모 척도가 기능점수로, 그림 2와 같이 데이터 기능(내부논리파일, 외부연계파일)과 트랜잭션 기능(내부입력, 외부입력, 외부조회)으로 구성된다[5-7].

그림 2. 기능점수 구성 요소
Fig 2. Components of Function Points

한소협 "소프트웨어 대가산정 가이드"에서 정의하는 기능점수는 국제 표준 기능점수 중에 개발 기능점수에 해당한다. 국제표준으로 제시하는 기능점수는 개발 기능점수, 개선 기능점수, 어플리케이션 기능점수의 세 가지 유형이 있다. 그러나 상기 가이드에서는 이 중에서 개발 기능점수로만 한정되어 있다. 실제로 그동안 수행되었던 개발비 감정 사례에서는 개발 외에도 고도화, 유지관리, 재구축, 맞춤식 수정 사업 등 다양한 영역에 걸쳐 있다. 극단적으로 써드 파티 개발 혹은 기 개발한 제품을 포함하더라도 감정 대상이 되는 전체 소프트웨어를 처음부터 개발하는 것을 전제로 하는 가이드 혹은 그 이전 감정 사례에서는 대가산정 가이드를 수정 없이 준용하므로 과다한 비용으로 감정하는 결과가 된다는 문제가 있다.

일반적으로 개발비 감정 대상 프로젝트의 경우 관련 문서가 충분하거나 정확하지 않아 모든 프로그램을 원점에서 개발했는지, 혹은 일부 프로그램이 써드 파티 개발 프로그램인지, 기존에 개발한 프로그램을 재사용했는지의 여부를 명확하게 제시하지 않고 있다. 이 경우 감정 결과는 실제 개발비용에 비해 과대추정될 우려가 있다.

앞에서 기술한 것처럼, 소프트웨어 대가산정 가이드의 개발비 단가는 개발 기능점수 당 단가이며, 이 단가는 2004년 사업대가기준을 본수 방식에서 기능점수 방식으로 바꾸는 혁명적인 과정에서 대형 SI 업체들이 기존에 수행했던 프로젝트 자료에 근거하여 유도된 기능점수 당 평균 단가로 대가산정 가이드의 개정 시에 수정을 거듭하고 있으며, 그림 3과 같은 요소를 포함한다[7].

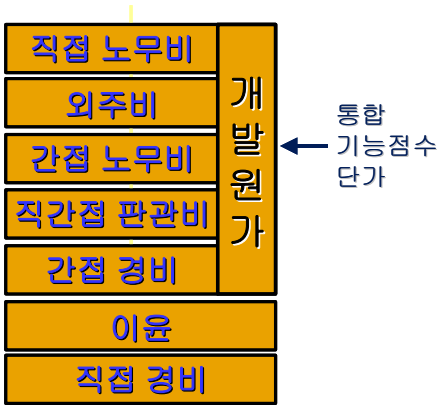

그림 3. 기능점수 당 단가 구조
Fig 3. Unit Cost per Function Points

2004년 대가산정 기준 개정 시에 기능점수 단가 유도에 필자가 참여하였으며, 데이터 수집의 어려움으로 제한된 몇몇 대형 SI 업체의 데이터만 적용되었고, 실제 유도된 단가에 대해 정치적 사회적 요소를 고려하여 상향 조정된 값이다. 그러나 이 기능점수 당 단가에 대해서는 발주기관의 입장에서는 지나치게 상향 조정된 값이라는 불만이 있고, 수주기관의 경우 그동안의 적자를 보전하지 못하는 낮은 값이라는 불만을 가지고 있다. 또한 대형 프로젝트가 아닌 중소규모 프로젝트의 경우, 다른 단가가 필요하다는 불만도 지속적으로 제기되었다. 이와 같은 한계를 가지고

있는 대가산정 가이드의 단가를 프로젝트 성격과 감정의 목적이 근본적으로 상이한 개발비 감정에 준용할 때 과대 추정될 위험성이 있다. 적정 개발비를 감정하기 위해서는 이러한 문제점을 해결할 방안을 제시해야 한다.

표 1. 간이법 기능점수 복잡도
Table 1. Estimation Form Simple Function Points

기능유형	가중치	합계
	평균복잡도	
내부논리파일	( ) × 7,5	
외부연계파일	( ) × 5,4	
외부입력	( ) × 4,0	
외부출력	( ) × 5,2	
외부조회	( ) × 3,9	
총 기능점수		

소프트웨어 대가산정 가이드를 기준으로 개발비를 감정할 때에는 표 1의 평균 복잡도, 표 2의 정통법 복잡도를 적용한다[3]. 기능점수는 요구분석이 정확하게 수행되는 것을 전제로 하고 있지만, 가이드가 적용되는 공공사업의 예산 수립 시, 혹은 요구분석 이전의 단계, 혹은 요구분석 정보가 부족한 경우 사용하는 것이 평균 복잡도로 정확도는 매우 낮은 문제점을 가지고 있다. 표 2의 정통법은 IFPUG CPM 기반의 국제 표준 기능점수 복잡도이다. 비용산정에서는 기능점수 당 단가의 모호함으로 비록 상세하고 정확한 규모추정이 이루어져도 개발비 추정 결과에는 많은 의문이 따른다.

표 2. 정통법 기능점수 복잡도
Table 2. Estimation Form Standard Function Points

기능유형	가중치			합계
	낮음	보통	높음	
내부논리파일	( ) × 7	( ) × 10	( ) × 15	
외부연계파일	( ) × 5	( ) × 7	( ) × 10	
외부입력	( ) × 3	( ) × 4	( ) × 6	
외부출력	( ) × 4	( ) × 5	( ) × 7	
외부조회	( ) × 3	( ) × 4	( ) × 6	
총 기능점수				

이처럼 비용산정 시에도 한계를 가지고 있는 대가산정 가이드에 준용한 개발비 감정에서는 여러 가지 문제점을 지니고 있으므로, 경우에 따라 감정인이 수정한 복잡도 값을 적용한 사례도 있다. 이처럼 기능점수 당 단가가 과다하다고 생각하지만, 공표된 단가를 수정하는 것이 어려운 감정사례에서 별도의 복잡도를 적용하기도 한다.

“소프트웨어 대가산정 가이드”는 기능점수 산정을 위한 국제 표준 문서인 IFPUG CPM[7]을 기반으로 한다고 했지만, 다르게 모호하게 기술된 부분이 존재한다. 상기 매뉴얼에서는 트랜잭션에 관한 기능요소를 구분하기 위해 단위 프로세스를 제시하고 있고, 이 프로세스는 기본 의도를 기준으로 식별한다. 그러나 상기 가이드에서는 이러한 기본 의도를 전혀 기술하지 않아, 기능 식별 오류의 발생 가능성이 있고, 단위 프로세스 별로 데이터 기능점수를 산정하는 오류가 흔히 발생된다. 이와 같은 대가산정 가이드 자체가 가지는 문제점으로 인해 개발비 감정의 지준으로 적용되기에는 많은 한계점이 존재한다.

2.3 기능 점수의 문제점

앞에서 기술한 한소협 “소프트웨어 대가산정 가이드” 자체의 문제점[7] 중, 기능점수 당 단가는 조정 가능하며, 가이드의 모호성은 해결가능한 문제점이다. 그러나 상기 대가산정 가이드의 기본 전제는 소프트웨어 규모를 기능점수로 산정하는데 있다. 만약 기능점수 산정이 부정확하거나 모호한 경우 감정의 신뢰성이 크게 위협받을 수 있다.

전통적으로 소프트웨어 규모산정은 LOC, 본수 방식이 기본이었으나, LOC의 경우 동일한 요구사항을 기반으로 작성해도 LOC의 크기가 달라질 수 있다는 점과 어셈블리어로 작성된 프로그램의 규모가 고급언어로 작성된 프로그램의 규모

보다 크고, 고급언어의 경우에도 편차가 심하다는 생산성 모순 문제가 발생한다. 따라서 개발자 관점이 아닌 사용자 측면에서 요구사항을 기반으로 기능점수 방식이 제안되었다.

기능점수 방식은 해당 소프트웨어를 사용하는 이해당사자의 관점에서 규모를 추정하는 유용한 방법이기는 하지만, 실제 기능점수는 상세 설계가 진행되고 나서야 얻을 수 있는 정보를 기반으로 측정이 가능하므로, 요구사항만 제시된 경우 일관성 있는 규모산정이 어렵고, 규모산정과 추적 관리에 큰 오버헤드가 소요된다. 더군다나 대부분의 개발비 감정 사례에서는 관련 문서가 충분하지 않고, 발주자와 수주자 간의 논란이 벌어져서 규모산정의 기준점을 설정하기 쉽지 않은 상황에 있다. 기능점수 산정에는 일정 수준 이상의 지식과 경험이 필요하므로 전문적인 교육과 경험을 쌓은 기능점수 전문가 간에도 측정값의 오차는 상당한 것으로 보고되고 있다. 이러한 상황에서 개발비 감정에 기능점수를 적용하는 경우 오류나 편차가 발생할 가능성이 높다.

3. 유스케이스 기반의 규모 추정

3.1 유스케이스 점수

유스케이스 점수 기반의 규모산정 기법은 G. Karner가 제안한 이후로 많은 연구가 이루어졌다. G. Karner는 유스케이스와 액터를 이용하여 소프트웨어 시스템 규모를 산정한다. 해당 소프트웨어를 구성하는 유스케이스와 액터의 개수를 추정한 다음 이들 각각에 복잡도를 고려한다. 구체적으로는 환경 복잡도 요인과 기술 복잡도 요인을 고려하여 규모를 산정하였다. Ochodek 등은 유스케이스의 복잡도를 트랜잭션보다는 트랜잭션을 구성하는 단계를 기준으로 복잡도를 구했고, Periyasami 등은 유스케이스를 구성하는 문장에서 도출하였으나 실제 사례에서 적용되지 못했다. 유스케이스 개수는 추상화를 어느 수준까지 하느냐에 따라 상이한 결과가 나올 수 있다. 따라서 일관성 있는 결과를 유도하기 위해 단위 기능 중심의 유스케이스 점수 측정 방안[8]과 트랜잭션 중심의 유스케이스 점수 측정 방안[9]이 제시되기도 했다. 그러나 이 두 가지 방안 모두 요구분석이 상세하게 수행되어 요구명세서 외에 다양한 문서와 함께 소프트웨어개발 정보가 제공되는 경우에 가능한 방법이다.

본 연구에서는 일반적인 소프트웨어 비용산정 모델이 아닌, 개발비 감정 단계에서 적용할 수 있는 모델을 도출하는 것이 목적이므로, 가능하면 제한된 정보를 가지고 있는 감정 단계에서 사용할 수 있는 유스케이스 점수[9] 기반의 추정 방안을 제시한다.

유스케이스 기반 공수 추정은 다음 순서로 진행된다.

단계 1: 액터의 분류에 따른 크기(UAW)를 계산한다. 복잡도 수준에 따라 가중치를 각각 1, 2, 3으로 부여한다.

단계 2: 유스케이스의 크기(UUCW)를 계산한다. 복잡도 수준에 따라 가중치를 각각 5, 10, 15로 부여한다.

단계 3: 조정 전 유스케이스 점수(UUCP)를 계산한다.

$$UUCP = UAW + UUCW$$

단계 4: 보정 계수인 기술적 복잡도 인자(TCF)를 계산한다. 2018년 개정 “SW사업 대가산정 가이드”의 보정 항목을 반영하기 위해 기술적 요인(TF)은 분산 시스템(T1), 성능(T2), 이식성(T3), 보안성(T4)을 적용한다.

단계 5: 조정 후의 유스케이스 점수(UCP)는 기술적 복잡도 요인(TCF)을 반영하여 계산한다.

UCP = UUCP*TCF

단계 6: 생산성 인자(Person-Hour/SUCP)를 반영하여 공수를 계산한다.

Effort = UCP*생산성 인자

4. 성능 평가

4.1 산정 기법의 비교

본 논문에서 적용한 성능 평가 절차는 [10]의 성능 평가 방안을 적용한다. 평가 대상 프로젝트는 개발비 감정 사례들과 유사한 유형의 프로젝트로 진행하였으며, 공수 추정 시에 감정 사례들의 상황과 유사하도록 제공되는 문서와 정보를 최소화하였다.

표 3. 평가 대상 프로젝트
Table 3. Project to be Evaluated

프로젝트	언어	공수
1	Java	550.54
2	Java	560.43
3	Java	402.53
4	C#	546.75
5	C++	1014.91
6	C++	663.21
7	Python	503.53
8	Python	414.91
9	Java	473.46
10	Java	360.02

총 10개의 시스템으로 학부 및 대학원의 소프트웨어 공학 수업 과정에서 진행한 프로젝트로 완성되었다. 각 팀은 4명으로 구성되었으며, 총 개발 공수는 회의 시간을 포함하여 전체 인원이 개발에 소요된 총 시간을 포함한다. 각 프로젝트의 정보는 표 3과 같다.

각 기법의 정확도를 비교하고, 통계적 유의성을 판단하기 위해 MRE, MMRE, 결정계수(R2), t 검정을 사용하였다. MRE와 MMRE는 각각 상대오차의 절댓값과 평균값이다. 상대오차는 실제 공수와 추정 공수의 차이에 대한 절댓값을 실제 공수로 나눈 값이므로, MMRE가 작을수록 추정 모델의 정확도가 높다는 것을 알 수 있다. MMRE의 계산식은 다음과 같다.

$$MMRE = \sum_{i=1}^{10} MRE$$

$$MRE = \frac{|\text{실제 공수} - \text{추정 공수}|}{\text{실제 공수}}$$

4.2 실험 결과 및 분석

기능점수 기반의 규모산정 기법과 본 연구에서 제안하는 유스케이스 기반 산정 기법의 추정 정확도 비교한 결과를 표 4에 정리하였다.

기능 점수 기반의 규모 산정 기법보다 본 연구에서 제안하는 유스케이스 점수 기반의 기법이 정확도가 더 뛰어나다는 것을 알 수 있다.

MMRE를 이용하여 확인한 정확도 향상이 통계적으로 유의한지 t 검정을 통해 검증하였다. 두 모집단의 표본 수가 동일하고, 이분산을 가정하여 두 모집단의 평균값을 비교하기 위한 독립표본 t 검정의 자유도와 검정 통계량은 각각 다음과 같다.

$$\phi = \frac{(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2})^2}{(\frac{S_1^2}{n_1})^2/(n_1-1) + (\frac{S_2^2}{n_2})^2/(n_2-1)}$$

$$\frac{\overline{X} - \overline{Y}}{\sqrt{\left(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}\right)}} \sim t(\varnothing)$$

표 4. 산정 모델 비교
Table 4. Comparison of Estimation Model

프로젝트	FP MRE(%)	UCP MRE(%)
1	80.84	10.21
2	60.87	42.92
3	89.72	22.41
4	72.28	13.91
5	34.82	23.18
6	12.13	10.81
7	80.32	52.22
8	81.07	32.19
9	48.02	32.22
10	90.09	18.39
MMRE	69.016	25.846

t 검정 결과 p 값이 유의수준보다 작으면 귀무가설을 기각할 수 있다. 여기에서 귀무가설은 두 집단의 평균이 동일하다는 것에 해당한다.

표 4의 산정 결과를 t 검정으로 비교한 결과 p 값은 0.05보다 작아 95% 신뢰수준에서 귀무가설이 기각된다. 즉, 본 논문에서 제안하는 유스케이스 기반의 규모산정 기법의 정확도가 기능점수 기반의 산정 기법보다 통계적으로 유의하게 향상되었다고 판단할 수 있다.

결정계수(R^2)는 본 논문에서 제안하는 기법의 설명력을 나타낸다. 만약 정확한 공수 추정이 가능하다면, 추정 공수와 실제 공수 사이에는 강한 상관관계를 나타낸다. 논 논문의 추정 공수와 실제 공수 사이의 R^2 값은 0.81로 비교적 강한 선형 상관관계를 나타내므로 본 논문에서 제안하는 유스케이스 기반의 추정 기법은 설명력이 있는 기법이라고 판단할 수 있다.

	FP	UCP
평균	69.016	25.846
분산	432.6224	195.0581
관측수	10	10
가설 평균차	0	
자유도	16	
t 통계량	5.448949	
P(T<=t) 단측 검정	2.68E-05	
t 기각치 단측 검정	1.745884	
P(T<=t) 양측 검정	5.36E-05	
t 기각치 양측 검정	2.119905	

5. 결 론

소프트웨어 개발비 감정은 소프트웨어공학 분야에서 그동안 연구를 진행한 소프트웨어 비용산정 기법을 적용하여 개발 비용을 추정하는 것이다. 개발비 감정은 주로 분쟁 대상이 된 소프트웨어 시스템을 개발하는데 소요된 적정 비용을 추정하는 것이 기본적인 목적이지만, 대부분의 경우 감정대상 프로그램은 수발주자 사이에 의견이 다르고, 요구분석이 제대로 진행되지 않았거나 관련 정보가 부실, 부정확한 상태이다. 또한 정상적으로 종료되지 않은 프로젝트도 많은 비중을 차지하고 있다.

한국저작권위원회에서 발표한 완성도 감정 가이드라인에서는 개발비 감정을 포함하는 감정 단계별 수행 업무를 제시하며 개발비 감정을 하는

전문가가 활용할 수 있는 감정 가이드라인과 사례들을 제공한다.

개발비 감정 가이드라인에서는 기능점수 및 투입공수 방식을 제시하고 있고, 구체적인 절차는 한소협 “소프트웨어 대가산정 가이드”를 참고하도록 했다. 기능점수는 국제 표준의 소프트웨어 규모 척도로 전체 소프트웨어 규모에 해당하는 기능점수에 단가를 곱해 개발비용을 추정하고 다양한 보정인자를 적용하여 조정하는 절차를 따른다.

그동안 많은 개발비 감정 사례에서 상기 가이드를 준용하여 감정이 진행되어 왔다. 그러나 가이드 자체의 본질적인 문제점과 소프트웨어 규모 산정의 기본이 되는 기능점수가 가지는 본질적인 한계 때문에 개발비 감정은 한계를 가지고 있다. 본 논문은 이러한 개발비 감정의 문제점을 해결하기 위해, 유스케이스 기반의 규모추정 방안을 제시한다.

유스케이스 기반의 규모추정 방안의 평가 대상 프로젝트는 개발비 감정 사례들과 유사한 유형의 프로젝트로 진행하였다. 학부 및 대학원의 소프트웨어 공학 수업 과정에서 진행한 프로젝트를 대상으로 한 성능평가 결과 본 논문에서 제안하는 유스케이스 기반 규모 추정 방안이 기능점수 방법에 비해 정확도가 향상되었으며, 통계적으로 유의하다는 것을 검증하였다. 추후 연구에서 본 논문에서 제안한 유스케이스 기반의 규모추정 방안을 실제 다양한 사례에 적용하여, 적절한 생산성 인자와 단위 유스케이스 점수 당 단가를 유도하여 “SW사업 대가산정 가이드”에 적용하는 방안을 제시하고자 한다.

참 고 문 헌

[1] 한국저작권위원회, “저작권 관련 감정사건 판례집(4)”, p.417, 2018. ISBN: 978-89-6120-399-2

[2] 권기태, “컴퓨터 프로그램 완성도 감정 개요”, 한국 소프트웨어 감정평가학회 논문지, vol.11, no.2, pp.1-8, Dec. 2015. http://www.i3.or.kr/html/paper/2015-2/(2)2015-1.pdf

[3] 한국소프트웨어산업협회, “SW사업대가산정 가이드(2017년 개정판)”, p.451, 2017. https://www.sw.or.kr/site/sw/ex/board/View.do?cbIdx=276&bcIdx=37312

[4] 한국소프트웨어산업협회, “국제표준기반 기능점수 산정 안내서”, p.209, 2017. https://www.swit.or.kr/IS/web/isCbmRefView.jsp?ref_sq=655&schCode=13

[5] 권기태 외 공역, “기능점수와 소프트웨어 측정”, p.372, 도서출판 그린, 2004. ISBN: 978-89-5727-017-2

[6] IFPUG, “Counting Practice Manual”, Release 4.2.1 International Function Point Group, p.2-3, 2008. https://epmc2.monsite-orange.fr/file/d6ab0a1755c60de1840c1337f50b64d2.pdf

[7] 권기태, “개발비 감정 시 SW 사업 대가산정 가이드 적용 문제점”, 한국 소프트웨어 감정평가학회 논문지, vol.12, no.1, pp.11-20, June 2016. http://www.i3.or.kr/html/paper/2016-1/(2)2016-1.pdf

[8] 허령, 서영덕, 백두권, “유스케이스 점수 측정의 신뢰도 향상을 위한 단위기능 중심의 유스케이스 정제 방법”, 정보과학회논문지 vol.42, no.9, pp.1117-1123, September 2015. http://kiise.or.kr/e_journal/2015/9/JOK/pdf/06.pdf

[9] Roy K. Clemmons, “Project Estimation With Use Case Points”, CrossTalk The Journal of Defense Software Engineering, pp.18-22, February 2006. http://www.royclemmons.com/articles/docs/0602Clemmons.pdfB.

[10] 이선경, “유스케이스 트랜잭션 기반의 소프트웨어 공수 예측 기법”, KAIST 석사학위논문, p.23, 2010. https://koasas.kaist.ac.kr/handle/10203/34923

저 자 소 개

권기태(Ki-Tae Kwon)

1986년 서울대학교 계산통계학과 졸업
1988년 서울대학교 계산통계학과 석사졸업
1993년 서울대학교 계산통계학과 박사졸업
1996년 미국 Univ. of Southern California, 전산학과 Post-Doc.
현재 강릉원주대학교 컴퓨터공학과 교수
<주관심분야 : 소프트웨어공학, 데이터 사이언스, Statistical Learning>