요구사항 불확실성, 통제 표준화, 상호작용이 소프트웨어 품질에 미치는 영향

김기윤°, 양동구

광운대학교 경영학과

The Effects of Requirement Uncertainty, Control Standardization, Interactions on Software Quality

Kim, Ki-Yoon°, Yang Dong-Gu

Dept. of Business Administration, Kwangwoon University

E-mail : min1203@daisy.kwangwoon.ac.kr, dong99@hanmail.net

요 약

소프트웨어 개발 프로젝트에 대한 위험관리(risk management) 이론을 근거로, 소프트웨어 품질, 요구사항 불확실성, 통제표준화, 상호작용에 대한 5개의 가설이 설정되었고, 3개의 SI 회사로부터 117개 표본이 추출되어서 구조방정식모형에 의해서 실증적으로 검증되었다. 본 연구결과에 따르면, 요구사항 불확실성과 통제 표준화는 소프트웨어 품질에 직접적인 영향을 미치지 않고, 사용자-프로젝트 팀 간의 상호작용에 직접적인 영향을 미치며, 이를 통해서 소프트웨어 품질에 간접적인 영향을 미친다. 그럼으로, 요구사항 불확실성이 낮을수록, 그리고 통제 표준화가 높을수록, 사용자-프로젝트 팀 간의 상호작용은 높아져서 궁극적으로는 소프트웨어 품질이 높아진다고 해석할 수 있다.

1. 서론

전통적으로 소프트웨어 개발 프로젝트는 기능적 행위로 생각했으며, 프로젝트팀에서 숙련된 기술자들의 개발도구 및 방법을 통제하였다. 이로인하여 소프트웨어 개발조직이 미성숙할수록 소프트웨어 개발 프로젝트의 비용, 일정, 품질 등에서 소프트웨어 개발위험이 높아졌다. 본 연구의 초점은 주어진 예산 내에서 적시에 완성해야하는 소프트웨어 개발프로젝트의 진행과정에서 존재하는 위험들이며, 이런 위험에는 불확실성과 손실이라는 두 가지 특성이 있다. 불확실성은 조직 내에서 완전한 정보의 부재를 의미하며, 손실은 원치 않는 결과를 의미한다.

Carnegie Mellon University의 SEI(Software Engineering Institute)에서는 소프트웨어 개발위험을 제품공학(Product Engineering), 개발환경(Developme nt Environment), 프로그램 제약(Program Constrai nts)으로 분류하였다. 제품공학은 요구사항, 설계, 코드 및 단위시험, 통합 및 시험, 공학 전문가들로 구성되고, 개발환경은 개발과정, 개발시스템, 관리과정, 관리방법, 작업환경으로 구성되며, 프로그램 제약은 자원, 계약, 프로그램 인터페이스로 구성된다.

소프트웨어 개발 프로젝트에서 과업 측면에서 사용자 요구사항을 충족시키는 소프트웨어 품질이 중요하고, 기술 측면에서는 통제 표준화가 중요하고, 행위자 및 구조 측면에서는 사용자와 프로젝트 팀간의 상호작용이 중요하다. 본 연구의 목적은 소프트웨어 개발프로젝트의 위험관리 관점에서 요구사항 불확실성과 통제 표준화가 상호작용과 소프트웨어 품질에 미치는 영향을 분석하는 것이다. 또한, 사용자와 소프트웨어 개발프로젝트 팀간의 상호작용이 소프트웨어 품질에 미치는 영향을 분석하는 것이다.

2. 이론적 배경과 연구가설

소프트웨어 개발위험 혹은 소프트웨어 위험(software development risk or software risk)에 관한 연구는 의사결정 및 위험관리(혹은 위험기반 접근방법(risk- based approach)) 관점에서 연구되었다. 의사결정 관점에서의 연구(Boehm and Ross 1989, Charette 1989, Boehm 1991)는 가능한 손실을 줄이고 이익을 늘리는 게임으로 일어날 수 있는 결과에 대한 확률분포를 고려해야 하는 방법과 위험을 기술하기 위해서 퍼지 집합이론(fuzzy set theory)을 이용해서 위험을 측정하는 접근방법이 있다(Kangari & Boyer, 1989).

소프트웨어 위험에 대한 위험관리연구 중 고전적인 연구(Lyytinen, Mathiassen and Ropponen 1998)는 다차원적이고 정성적인 사회 기술적 모형(socio-techn ical model)으로서 다음과 같이 네 가지 접근방법으로 구분할 수 있다. 첫째, Alter와 Ginzberg(1978)의 실행 접근방법(implementation approach)은 위험을 "프로젝트 완료에 대한 불확실성"으로 정의했고, 둘째, Davis(1982)의 상황적합 접근방법(contingency approach)은 위험을 "정확하고 완전한 사용자 요구사항을 충족시키는 것에 대한 어려움"으로 정의했다. 셋째, McFarlan(1982)의 포트폴리오 접근방법(portfolio approach)은 위험을 "수익획득의 실패, 계획수준을 초과한 실행 비용 및 시간, 추정치 이하의 기술적 성과, 시스템과 하드웨어 및 소프트웨어의 불일치"로 정의했으며, 네째, Boehm(1991)의 소프트웨어 위험 접근방법(software risk approach)은 위험을 "불만족스런 결과의 발생확률과 손실을 곱한 것"으로 정의했다,

이러한 위험관리 접근방법 중에서 McFarlan(1981), Bohem(1989), Barki(1993) 등은 특히 프로젝트 성과와 관련된 위험인 성과위험에 연구의 초점을 두었다. 소프트웨어 위험에 대한 위험관리연구 중 최근연구인 Nidumolu(1996)의 위험기반 접근방법에 의하면, 구조적 상황적합(structural contingency) 관점에서 요구사항의 불확실성과 표준화가 프로젝트 성과에 영향을 준다고 했다. 이러한 소프트웨어 개발위험(Software Development Risk)을 관리하기 위해서는 지속적인 위험관리(CRM; Continuous Risk Management)가 필요하다. 지속적인 위험관리 패러다임은 프로젝트 생명주기 동안 연속적인 행위들로 파악되는 일련의 기능들로서 위험에 대한 식별 -> 분석 -> 계획 -> 추적 -> 통제 -> 의사소통에 대한 지속적인 관리이다.

소프트웨어 개발 프로젝트에서 과업 불확실성은 요구사항 불확실성으로 나타나고, 이러한 불확실성을 낮추기 위해서 사용자-프로젝트팀간의 상호작용이 실행된다. Jones와 Harrison(1996)에 의하면, 이러한 프로젝트팀의 성과에 영향을 주는 요인들은 사용자 요구사항에 대한 이해와 시스템 설계에 대한 팀 구성원들의 몰입이라고 했다.

Nidumolu(1996a)는 사용자 요구사항의 불확실성이 감소할수록 수직적 조정 및 수평적 조정이 증가해서 공정통제와 제품유연성이 증가한다고 했다. 이를 반대의 의미로 표현하면, 요구사항 불확실성이 높을수록 사용자-프로젝트팀간의 상호작용은 낮아져서, 그 결과로 프로젝트 성과는 낮아질 것이라는 추론이 가능하다. 우리나라의 소프트웨어 개발 프로젝트 경우에는 대부분 사용자 주도개발보다는 사용자-프로젝트팀간의 상호작용에 의해서 이루어지고 있다.

가설 1: 요구사항 불확실성이 높을수록 사용자-프로젝트팀간의 상호작용은 낮아진다.

Nidumolu(1996b)는 요구사항 불확실성이 잔여성과위험을 증가시켜서 프로젝트의 성과를 감소시킨다고 했다. 이와 같이 소프트웨어 위험 중에서 특히 요구사항 불확실성이 높을수록 프로젝트 성과인 소프트웨어 품질은 낮아질 것을 예상하여 다음과 같은 가설을 설정하였다.

가설 2: 요구사항 불확실성이 높을수록 소프트웨어 품질은 낮아진다.

공식적 통제에 대한 개념은 조직의 대리이론(agency theory)에서 성과평가전략을 위해서 제시되었고(Eisenhardt, 1985), 행위통제(behavioral control)와 산출통제(outcome control)로 구분된다(Eisenhardt, 1989; Klepper, 1990). Kirsch(1997)는 정보시스템 개발(ISD: Information System Development) 프로젝트에 대한 통제를 크게 공식적인 통제와 비공식적인 통제로 구분했다. McFarlan(1982)은 프로젝트 관리와 관련된 프로젝

트 3가지 위험(규모, 기술경험, 프로젝트 구조, 등)을 줄이기 위한 공식적인 통제들을 제시했다. 소프트웨어 개발 절차나 기술을 위한 표준화된 도구와 기술을 이용하는 정도를 나타내는 통제표준화의 행위 및 산출 통제를 높일수록, 프로젝트팀에 대한 사용자의 몰입을 증가시키는 상호작용이 높아질 것이다. Nidumolu(1996b)는 통제 표준화가 높을수록 잔여성과위험이 낮아져서 프로젝트의 성과는 높아진다고 했는데, 이와 같이 프로젝트 성과가 높아지도록 위험을 감소시키는 의사소통 과정이 바로 사용자-프로젝트팀간의 상호작용이다. 그러므로, 통제 표준화가 높을수록 사용자-프로젝트팀간의 상호작용은 높아질 것으로 예측된다.

가설 3: 통제 표준화가 높을수록 사용자-프로젝트 팀간의 상호작용은 높아진다.

소프트웨어 개발 절차나 기술을 위한 표준화된 도구와 기술을 이용하는 정도를 나타내는 통제표준화에는 행위통제와 산출통제가 있다. Phan, Vogel, 그리고 Nunamaker(1995)의 사례연구에 의하면, 소프트웨어 개발 자동화 도구인 IDSS(Integrated Development Support System)에 의해서 개발생명주기 단계를 통제하는 산출통제는 물론, 설계, 코딩, 품질보증, 등에 관한 행위통제를 함으로써, 개발프로젝트의 생산성을 22% 이상 증가시킨다고 했다. Nidumolu(1996a, 1996b)는 통제 표준화(control standardization)가 높을수록 잔여성과위험이 낮아져서 프로젝트의 과정 및 제품 성과는 높아진다고 했다. 그러므로, 공식적인 통제인 행위통제와 산출통제에 대한 표준화가 높을수록 프로젝트 성과인 소프트웨어 품질은 높아질 것이다.

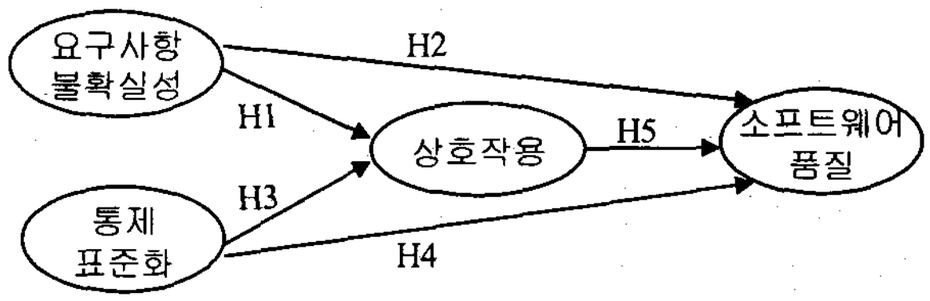
가설 4: 통제표준화가 높을수록 소프트웨어 품질이 높아진다.

사용자가 정보시스템 개발과정에 참여해서 개발 프로젝트 팀과 상호작용 함으로써, 개발되는 정보시스템에 대한 이해도가 향상되고 사용자 요구사항도 정확하게 반영되므로, 궁극적으로 사용자 정보 만족도를 극대화한다고 주장했다(McKeen, Guimaraes, 1994; Doll and Torkzadeh, 1989). 또한, 사용자 참여로 인한 상호작용으로 정보 시스템의 기능성이 향상되고, 정보 시스템 사용도 혹은 활용도가 높아진다고 주장했다(Anderson, 1985; Baroudi, Olson, and Ives, 1983). 이와 같이 사용자와 프로젝트 팀 간의 상호작용은 소프트웨어 개발 프로젝트의 성과인 사용자 만족도와 시스템 활용도는 물론 소프트웨어 품질에도 영향을 줄 것이라고 예상하여 다음과 같은 가설을 설정하였다.

가설 5: 사용자-프로젝트 팀 간의 상호작용이 높을수록 소프트웨어 품질이 높아진다.

3. 연구설계

구조방정식모형(Structural Equation Modeling)으로서 LISREL(LInear Structural RELations)은 잠재변인들의 복잡한 인과적 관계성을 검증하고, 각 잠재변인을 측정하는 관찰변인이 얼마나 잠재변인을 적절히 측정하는지를 파악할 수 있는 복합적인 통계기법이다. 본 연구에서는 이론적으로 안정된 개념적 틀 속에서 간명한 구조방정식 모형을 설정하고, 이를 타당하게 분석하기 위해서, 우선 소프트웨어 개발 프로젝트의 성과인 소프트웨어 품질에 영향을 주는 개발위험으로서, 요구사항 불확실성, 통제표준화, 그리고 사용자-프로젝트 팀 간의 상호작용과 같은 변수들을 도출하였다. 그리고, 기존 연구에서 입증되어진 요구사항 불확실성과 통제 표준화가 사용자-프로젝트 팀 간의 상호작용에는 직접적으로, 소프트웨어 품질에는 직접 및 간접적으로 영향을 미칠 것이라는 것을 제안하면서 다음과 같은 연구모형을 설정하였다.



<그림 1> 연구모형

표본추출대상기업은 우리나라 소프트웨어 개발업체를 대표하는 LG-CNS, 삼성 SDS, DACOM-ST 3개회사이다. 소프트웨어 개발 프로젝트 및 응답자들에 관한 표본의 각 항목들이 비교적 고르게 분포되어 있어서, 본 연구에서 사용된 표본은 모집단을 비교적 잘 대표한다고 볼 수 있다. 연구자료 수집을 위한 프로젝트 관리자(PM)를 포함한 프로젝트내의 세부분야별 리더를 대상으로 했다. 설문지 배포 수는 3개회사별로 40부씩 120부

이며, 소프트웨어 품질 담당부서장들로부터 회수된 설문지는 120부 모두 회수되었으나, 응답이 부실한 3개 설문지를 제외한 총 117개의 설문지를 분석대상으로 했다.

요구사항 불확실성(requirement uncertainty)의 개념은 요구사항 불안전성으로 정의했다. 요구사항 불확실성은 3개 항(요구사항들의 후기단계에 변경, 초기단계 요구사항들이 후기단계와 동일, 요구사항들에 대한 미래 변경 가능성)으로 구성되었고, 5점 척도를 사용하여 측정하였으며, 점수가 높을수록 소프트웨어 개발위험이 높은 것을 의미한다.

통제 표준화(control standardization)의 개념은 산출통제 표준화로 정의했다. 통제 표준화는 4개 항목(프로젝트의 개발단계 구분, 프로젝트 개발 단계별 이정표 작성, 프로젝트 개발 단계별 완료 관련문서 작성)으로, 5점 척도를 사용하였으며, 점수가 높을수록 통제표준화가 구체적인 절차까지 준수된 것을 의미한다.

본 연구에서 사용자-프로젝트팀간의 상호작용이란 사용자들과 프로젝트팀원들 간의 의사소통을 포함한 전반적인 상호작용의 정도를 의미한다. 사용자-프로젝트팀간의 상호작용은 4개 항목(사용자 훈련의 완성도, 자료 처리부서와 사용자사이의 의사소통, 프로젝트 참여에 대한 사용자들의 반응, 사용자들의 전반적인 상호작용)으로 구성되었고, 5점 척도를 사용하였으며, 점수가 높을수록 상호작용의 정도가 높은 것을 의미한다.

소프트웨어 품질은 소프트웨어 개발 프로젝트의 제품성과로서 소프트웨어 유연성으로 정의했다. 구체적으로 소프트웨어 개발 프로젝트의 성과는 업무변화를 수용하기 위한 수정비용 및 유지보수비용, 업무변화를 수용하는데 소요되는 시간, 그리고 소프트웨어의 전반적인 장기 유연성으로 정의하는 것이 합리적이다. 소프트웨어 품질은 4개 항목(업무변화를 수용하기 위한 소프트웨어 수정비용, 업무변화를 수용하는데 소요되는 시간, 소프트웨어 유지보수 비용, 소프트웨어의 전반적인 장기 유연성)으로 구성되었고, 5점 척도를 사용하였으며, 점수가 높을수록 제품성과가 높은 것을 의미한다.

4. 실증분석

구조모형의 분석을 통해서 가설검증을 하기에 앞서 먼저 각 이론변수들의 단일차원성을 확보하기 위해 측정변수들에 대한 탐색적 요인분석을 실시하여, 부적합한 측정항목을 제거한 했다. 최종요인분석결과 요구사항 불확실성, 통제 표준화, 상호작용, 소프트웨어 품질를 측정하는 항목들은 모두 단일차원성을 지니고 있는 것으로 나타나고 있으며, 동일한 요인에 0.6 이상의 높은 적재값을 보이고 있다. 신뢰성 분석 결과, 신뢰성 계수인 Cronbach alpha 값은 0.64 - 0.87 로서, 측정변수들은 전반적으로 높은 내적 일관성을 가지고 있는 것으로 평가될 수 있다.

탐색적 요인분석에 의해서 찾아낸 연구변수들에 대한 집중타당성을 검증하기 위해서, LISREL 측정모형을 이용해서 확인적 요인분석을 실시했다. 네가지 연구변수들의 측정항목들에 대한 표준 요인적재값에 대한 t 값은 모두 $\pm$ 3.29 이상으로 나타나고 있어($p < 0.001$), 각 개념에 대한 측정항목의 요인적재값이 통계적으로 유의함을 나타내 주고 있다. 따라서 각 개념의 척도들은 집중타당성을 가지고 있다고 평가 할 수 있다. 또한, 구성개념에 대한 대표성을 갖는지를 평가하기 위하여 구성개념 신뢰성(construct reliability)과 분산추출값(variance extracted)을 계산한 결과를 제시했다. 일반적인 추천기준값이 구성개념 신뢰성은 0.7 이고, 분산추출값은 0.5 인데, 요인적재값이 적은 측정항목이 포함된 연구변수인 요구사항불확실성 경우에는 이 기준에 다소 못 미친다.

연구변수들 간의 상관관계분석 결과, 상호작용과 요구사항 불확실성, 통제표준화, 소프트웨어 품질과 각각 $p<0.01$ 수준에서 통계적으로 유의하고, 소프트웨어 품질과 요구사항 불확실성이 $p<0.05$ 수준에서 통계적으로 유의한 것으로 나타났다(Pearson 상관계수의 절대값이 0.183이상일 경우 $p<0.05$ 수준에서 유의하고, 0.26이상일 경우 $p<0.01$ 수준에서 유의함). 요구사항 불확실성과 통제 표준화만이 약한 상관관계를 나타내고 있고, 특히 요구사항 불확실성과 나머지 변수들과는 부(-)의 상관관계를 나타내고 있다.

<표 1> 상관계수분석 결과

연구변수	1	2	3	4
1.요구사항 불확실성	1.000			
2. 통제표준화	-0.036	1.000		
3. 상호작용	-0.318**	0.238**	1.000	
4. 소프트웨어 품질	-0.213*	0.151	0.359**	1.000

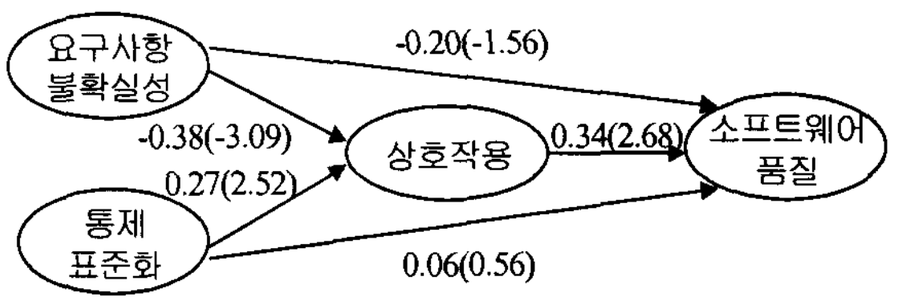
** $p<0.01$, * $p<0.05$

본 연구에서는 연구모형에서 제시된 구성개념들 간의 관계를 검증하는데 중점을 두고 있으므로, 공변량구조분석을 통해 구성개념들 간의 관계를 검증하고 평가하는데 유용한 LISREL을 이용하였다. LISREL을 사용하기 위한 기초통계치로는 상관행렬과 공분산행렬이 있는데, 본 연구에서는 상관행렬을 이용했다. 본 연구의 최종 연구모형에 대한 적합도 검증결과를 살펴보면, <그림 2>와 같은 구조모형이 도출되었다. Chi-square(χ^2)=128.39, df=85, p-value=0.0017, RMR=0.076, GFI=0.87, AGFI=0.82, NFI=0.82, NNFI=0.91, CFI=0.93 이다. 이러한 적합도 지수들은 다른 다변량기법과 마찬가지로 다변량 정상성 가정을 만족해야 한다. Q-plot에 의하면, 잔차에 대한 분포가 좌표의 45도를 따라 분포하고 있으므로, 다변량 정상성을 가정하고 있다고 할 수 있다.

적합도 평가에서 Chi-square값이 작고(최적모형에서는 0) 확률값이 크면(p>0.05), 모형이 적합하다고 평가한다. 그러나, Chi-square값은 표본행렬과 예측행렬의 차이로 나타낼 때 표본크기의 함수로 표시되기 때문에, 표본크기가 크면 두 행렬간의 차이가 작아도 매우 유의한 차이를 나타낸다. 따라서 Chi-square값에만 의존할 때, 표본크기가 커지면 실제모형이 적합하여도 적합하지 않은 것으로 잘못된 결정을 내리게되므로, 일반적으로 Chi-square값이 자유도의 두배를 넘지 않으면, 확률값(p)이 작아도 적합한 모형으로 평가한다. 그럼으로, 본 연구모형은 공변량 구조분석의 일반적인 평가기준인 Chi-square값에 대한 확률값(p)의 기준을 충족시키지는 않으나, Chi-square값이 자유도의 두배 보다 적으므로, 확률값이 작아도 적합한 모형으로 평가할 수 있다.

RMR은 잔차평균이므로, 0에 가까울수록 좋은 모형이다. RMR의 크기는 표본자료의 측정단위에 의해서 영향을 받는데, 표본자료가 상관행렬로 되어 있으면 예측행렬 역시 상관행렬이 되기 때문에, RMR의 크기에 영향을 주지 않는다. GFI는 회귀분석의 R^2 와 같은 개념으로 일반적으로 0과 1사이의 값을 갖는다. AGFI는 자유도가 작아서 GFI를 일반적으로 해석하기 곤할 경우에, 자유도에 영향을 받는 GFI를 교정한 것이다. NFI는 모든 측정문항이 하나의 요인을 측정하고 있다고 가정한 null 모형에서 연구자가 설정한 모형에 의해 설명되는 전체 정보의 비율이다. NNFI는 표본크기가 작은 경우에 사용할 수 있는 대안적인 지수이다. CFI는 자유도에 따른 오차를 감소시켜서 그 영향을 배제하고 모형을 비교하는 적합지수이다. 일반적으로 RMR은 0.05 보다 작고, GFI, AGFI, NFI, NNFI, CFI는 0.9 보다 크면(최적모형에서는 1), 매우 양호한 모형으로 평가된다. 본 연구모형에서 RMR이 0.076 이고, GFI, AGFI, NFI, NNFI, CFI가 모두 0.8 이상으로 나타나고 있으므로, 전체모형의 적합도는 받아들여질 수 있는 것으로 판단된다

변수들 간의 인과관계를 중심으로 설정된 가설을 검증한 결과는 다음과 같다. 가설 1을 검증한 결과, 요구사항 불확실성과 상호작용간의 경로계수 γ_{21}의 값은 -0.38 (t=-3.09)로 나타나 95%의 신뢰



숫자는 표준경로계수, ( ) 안의 값들은 t 값을 나타낸다.

<그림 2> 연구모형의 가설검증 결과

수준에서 유의적 이다. 따라서 가설 1은 채택되고, 요구사항 불확실성은 사용자-프로젝트팀간의 상호작용에 유의한 영향을 미치고 있다고 할 수 있다. 즉 요구사항 불확실성이 높을수록 사용자-프로젝트팀간의 상호작용은 낮아진다. 가설 2는 요구사항 불확실성과 소프트웨어 품질간의 경로계수 γ_{11} 의 값은 -0.20 (t=-1.56)로 95%의 신뢰수준에서 유의적이지 못하다. 따라서 가설 2는 기각되고, 요구사항 불확실성은 소프트웨어 품질에 유의한 영향을 미치고 있다고 할 수 없다. 가설 1과 가설 2의 검증결과, 요구사항 불확실성은 소프트웨어 품질에 직접적으로 영향을 미치지 않고, 반드시 사용자-프로젝트 팀간의 상호작용에 영향을 미친 후에 소프트웨어 품질에 영향을 미치는 것으로 나타나고 있다. 가설 3은 통제 표준화와 상호작용간의 경로계수 γ_{22} 의 값은 0.27 (t=2.52)로 95%의 신뢰수준에서 유의적 이다. 따라서 가설 3는 채택되고, 통제 표준화는은 사용자-프로젝트 팀 간의 상호작용에 유의한 영향을 미치고 있다고 할 수 있다. 즉 통제 표준화가 높을수록 사용자-프로젝트 팀 간의 상호작용은 높아진다. 가설 4은 통제 표준화와 소프트웨어 품질간의 경로계수 γ_{12} 의 값은 0.06 (t=0.57)

로 95%의 신뢰수준에서 유의적이지 못하다. 따라서 가설 4는 채택되고, 통제 표준화는 소프트웨어 품질에 유의한 영향을 미치고 있다고 할 수 없다. 가설 3과 가설 4의 검증결과, 통제 표준화는 소프트웨어 품질에 직접적으로 영향을 미치지 않고, 반드시 사용자-프로젝트 팀간의 상호작용에 영향을 미친 후에 소프트웨어 품질에 영향을 미치는 것으로 나타나고 있다. 가설 5은 상호작용과 소프트웨어 품질간의 경로계수 β_{12} 의 값은 0.34 (t=2.68)로 95%의 신뢰수준에서 유의적 이다. 따라서 가설 5는 채택되고, 상호작용이 소프트웨어 품질에 유의한 영향을 미치고 있다고 할 수 있다. 즉 사용자-프로젝트 팀 간의 상호작용이 높을수록 소프트웨어 품질이 높아진다.

<표 2> LISREL 분석 결과

가설	경로	경로 명칭	경로 계수	표준 오차	t값	검증 결과
1	요구사항 불확실성(ξ_1) -> 상호작용(η_2)	γ_{21}	-0.38	0.12	-3.09	채택
2	요구사항 불확실성(ξ_1) -> 소프트웨어 품질 (η_1)	γ_{11}	-0.20	0.13	-1.56	기각
3	통제표준화(ξ_2) -> 상호작용(η_2)	γ_{22}	0.27	0.11	2.52	채택
4	통제표준화(ξ_2) -> 소프트웨어 품질(η_1)	γ_{12}	0.06	0.11	0.57	기각
5	상호작용(η_2) ->소프트웨어 품질 (η_1)	β_{12}	0.34	0.13	2.68	채택

유의수준은 95% 수준에서 결정(t 값 > ± 1.96)

5. 결론

본 연구에서는 요구사항 불확실성과 소프트웨어 품질, 그리고 통제 표준화와 소프트웨어 품질 간의 관계에 있어서 사용자-프로젝트 팀간의 상호작용이 매개적 역할을 하는지에 대하여 살펴보고자 하였다. 이를 위해 관련된 기존연구이 검토를 통해 이론적 배경을 탐색하고, 연구문제 해결을 위한 가설을 설정하였으며, 이를 실증 분석하였다. 본 연구에서 제시된 연구가설에 대한 실증분석 결과와 시사점들은 다음과 같다.

첫째, 요구사항 불확실성이 높을수록 사용자-프로젝트팀간의 상호작용은 낮아지는 것으로 나타났다. 그러나, 요구사항 불확실성이 높을수록 소프트웨어 품질은 낮아진다는 것은 유의적으로 나타나지 않았다. 대부분 사용자 요구사항이 애매하거나 변경되어서 요구사항 불확실성이 높아질수록, 사용자와 프로젝트 팀 간의 상호작용은 오히려 낮아져서 궁극적으로는 소프트웨어의 개발위험을 증가시키게 된다. 그러나, 이와 같은 소프트웨어의 개발위험이 증가되어도 프로젝트의 성과인 소프트웨어 품질을 직접적으로 감소시키지 않는다는 것이다.

둘째, 통제 표준화가 높을수록 사용자-프로젝트 팀간의 상호작용은 높아지는 것으로 나타났다. 그러나 통제표준화가 높을수록 소프트웨어 품질이 높아진다는 것은 유의적으로 나타나지 않았다. 소프트웨어에 대한 통제 표준화는 소프트웨어 개발자들에게 체계적이고 일관성 있게 소프트웨어를 개발할 수 있도록, 사용자와 프로젝트 팀 간의 상호작용을 증가시키게 된다. 그러나, 이와 같은 통제 표준화가 증가되어도 프로젝트의 성과인 소프트웨어 품질을 직접적으로 증가시키지 않는다는 것이다.

셋째, 사용자-프로젝트팀간의 상호작용이 높을수록 소프트웨어 품질이 높아지는 것으로 나타났다. 사용자 요구사항이 명확하고 일관성이 있을수록, 표준화된 도구와 기술이 이용될수록 사용자와 프로젝트 팀 간의 상호작용에 직접적으로 영향을 주고, 이러한 상호작용이 높아질수록 프로젝트의 성과인 소프트웨어 품질를 높인다는 것이다.

결론적으로 요구사항 불확실성이 낮을수록, 그리고 통제 표준화가 높을수록, 사용자-프로젝트 팀 간의 상호작용은 높아져서 궁극적으로는 소프트웨어 품질이 높아지는데, 반드시 사용자-프로젝트 팀 간의 상호작용이 매개적 역할을 한다는 것이다. 그럼으로, 소프트웨어 품질향상을 위해서는 사용자-프로젝트 팀 간의 상호작용을 제고시키려는 노력이 필수적임을 시사하는 결과라고 할 수 있다.

[참고문헌]

[1] Anderson, E. E., "Managerial Considerations in Participative Design of MIS/DSS," *Information and Management,* Vol. 9, 1985, pp. 201-207.

[2] Barki, H., Rivard, S. and Talbot, J., "Toward an Assessment of Software Development Risk," *J. of Management Information System,* Vol. 10, No. 2, Fall 1993, pp. 203-225.

[3] Baroudi, J. J., Olson, M. H., and Ives, B.,

"An Empirical Study of the Impact of User Involvement on System Usage and User Satisfaction," *Communications of the ACM*, Vol. 29, No. 3, 1983, pp.232-238.

[4] Bentler, P. M., & Chou, C-P, "Practical Issues in Structural Modeling," *Sociological Methods and Research,* Vol. 16, 1987, pp.78-117.

[5] Boehm, B. W., *Software Risk Management,* Washington, DC: IEEE Company Society Press, 1989.

[6] Boehm, B. W., "Software Risk Management: Principles and Practices," *IEEE Software*, Jan. 1992, pp. 32-41.

[7] Boehm, B. W. and Ross, R., "Theory-W Software Project Management: Principles and Examples," *IEEE Trans. on Software Engineering*, Vol. 15, No. 7, 1989, pp. 902-916.

[8] Boomsma, A., The Robustness of Maximum Likelihood Estimation in Structural Equation Models, in P. Cuttance & R. Ecob (eds.), Structural Modeling by Example: *Applications in Educational, Sociological, and Behavioral Research,* New York: Cambridge University, 1989.

[9] Carr, Marvin J., Konda, Suresh L., Monarch, Ira, Ulrich, F. Carol, and Walker, Clay F., *Taxonomy-Based Risk Identification* (CMU/SEI-93-TR-6, ESC-TR-93-183), Pittsburg, Pa:Software Engineering Institute, Carnegie Mellon University, 1993.

[10] Cotterman, W. W. and Kumar, K., "User Cube: A Taxanomy of End Users," *Communication of the ACM*, Vol. 32, No. 11, 1989, pp. 1313-1320.

[11] Charette, R. N., *Software Engineering Risk Analysis and Management,* McGraw-Hill New York, 1989.

[12] Chittister, Clyde G. and Haimes, Yacov Y., "Sytem Integration via Software Risk Management," *IEEE Transactions on System, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 26, No. 5, Sep. 1996, pp. 521-532.

[13] Davis, G. B., "Strategies for Information Requirements Determination," *IBM Systems Journal*, Vol. 21, No. 1982, pp. 4-30.

[14] Doll, W., and Torkzadeh, J. G., "A Discrepancy Model of End-User Involvement," *Management Science*, Vol. 35, No. 10, 1989, pp.1151-1171.

[15] Dorofee, Audrey J., Walker, Julie A., Alberts, Christopher J., Higuera, Ronald P., Murphy, Richard L., and Williams, Ray C., *Continuous Risk Management,* Pittsburg, Pa:Software Engineering Institute, Carnegie Mellon University, 1996.