공공소프트웨어 사업의 개발 성향과 품질에 대한 실증적 연구

김 용 경* · 김 병 기**

An Empirical Study on the Development Propensity and Quality of the Public Software Project

Yong Kyong Kim* · Pyung Kee Kim**

Abstract

This study was empirically performed to demonstrate the development propensity and quality of the public software projects in Korea. The sample employed in this study contains 168 auditing reports on 107 public software projects which were carried out in the period of 1998 to 2003.

The important findings of this study can be summarized as follows.

The quality issue in the development process is getting more important with the lapse of time. In addition, the importance of end users' conveniency increases from year to year. Although the Pareto Principle(20 : 80 principle) is not applied strictly, most problems are caused by a few items. Finally, we find evidence that the overall quality of public softwares is positively influenced by the information system auditing.

Keywords : Information Systems Audit, Information Systems Quality, Software Engineering, Public Software

논문접수일 : 2004년 10월 11일 논문게재확정일 : 2004년 12월 9일

* 건양대학교 경영정보학과 교수
** 건양대학교 경영학부 교수

1. 서 론

1.1 연구 의의 및 목적

정보시스템이 효율적으로 개발되고 관리되기 위해서는 정보시스템이 본래의 사업 목적에 맞게 추진되고 있는가, 사용자의 요구사항이 적절하게 반영되고 그리고 구현되고 있는가를 점검하는 활동이 필요하다.

크로스비는 품질을 "요구사항에 대한 적합성"으로 정의하고 품질의 목표는 '무결함(zero defect)'으로 하였으며, 수정보다는 예방 위주의 시스템으로 접근해야 함을 주장하였다[Crosby, 1980]. 기본적으로 품질관리의 원칙은 발생 가능한 결함을 예방함으로써 정의된 요구사항에 대하여 무결성을 보장할 수 있다는 것이다. 대부분의 조직에서 정보시스템 개발에는 품질의 결함, 오랜 개발기간, 사용자 불만족 그리고 높은 개발비용 등과 같은 문제들이 나타난다. 그리고 이와 같은 문제들은 서로 복합적으로 작용하여 새로운 정보시스템의 개발을 강요하게 된다[Cusumano, 1991].

미국 소프트웨어 개발회사의 경우 결함율이 15%인 제품이 소비자에게 그대로 제공되는가 하면, 전체 개발 프로젝트의 25%가 실패, 이미 제작된 소프트웨어에 대한 재작업을 위해 전체 작업시간과 비용의 30%~44%를 투자, 프로젝트 일정의 50%만이 계획된 시간에 맞추는 등, 개발 프로세스 상에 많은 문제를 가지고 있다. 또한 소프트웨어 개발 생산성 향상이 대부분의 소프트웨어 개발 조직의 주요 과제였음에도 불구하고, 컴퓨터 하드웨어의 성능은 3년마다 거의 2배로 향상되는 반면 소프트웨어의 생산성은 매년 4% 정도의 향상에 그치고 있는 실정이다[Curtis, 1995].

선진국에 비해 소프트웨어 시장이 영세한 한국은 개발 프로젝트의 수행능력이나 문화가 상대적으로 성숙되어 있지 않아 소프트웨어 품질관리에 대한 인식이 아직은 미흡한 상태라고 할 수 있다. 우리나라는 1987년부터 공공부문의 정보시스템 개발 사업에 한하여 부분적으로나마 한국전산원과 그 위탁업체들을 통해 감리활동이 진행되어 오고 있다. 1990년대 이후 공공소프트웨어 개발 수요의 증가와 함께 감리 수요도 급격히 증가되었다. 그러나 그동안 국내에서 수행된 공공소프트웨어 개발 사업을 대상으로 하여, '우리나라 공공소프트웨어의 개발 성향은 어떠하며', '감리가 공공소프트웨어 품질에는 어떤 영향을 미치고 있는 가'에 대하여는 본격적인 연구가 되어있지 않은 것이 사실이다.

공공소프트웨어 개발 사업에 대하여, 연도별로 소프트웨어 품질에는 얼마나 많은 관심을 보이고 있는가? 개발 단계별로 어느 부분에서 가장 많은 지적사항이 발생하고 있는가? 등 즉, 우리나라 공공소프트웨어의 개발 성향을 알아보는 일은 공공소프트웨어의 발전과 효과적인 품질관리를 위하여 큰 의의가 있다고 생각한다.

본 연구에서는 국내에서 수행된 공공소프트웨어 개발 사업을 대상으로 실시된 감리보고서를 이용하여, 공공소프트웨어는 각 개발 단계별로 어떤 지적사항(문제점)들이 발생하고 있는가를 세부적으로 조사하였다. 이렇게 조사된 지적사항들을 통해, 우리나라 공공소프트웨어는 어떤 개발 성향을 나타내고 있는가를 알아보고, 또한 그것을 통해 정보시스템 감리 활동이 결국 공공소프트웨어 품질에 어떤 영향을 미치고 있는가를 밝혀보는데 연구의 목적이 있다.

1.2 연구 방법 및 구성

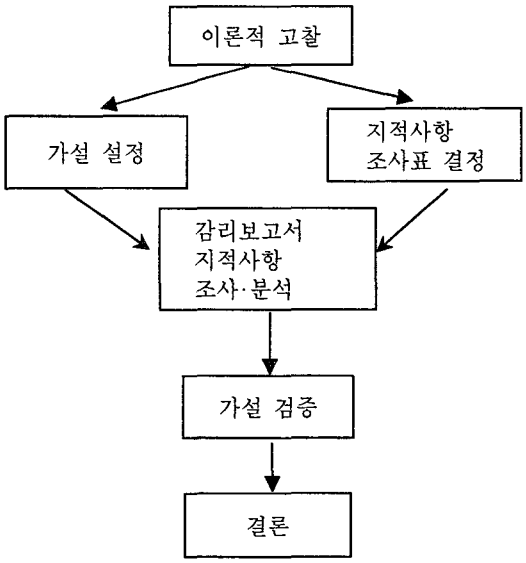
본 연구에서는 한국전산원의 위탁을 받아 H감리원과 Y감리원이 1998년부터 2003년까지 6년

동안에 실시한 정보시스템 감리 중 107개 프로젝트에 대한 168건의 감리보고서를 대상으로 조사를 실시했다. 연구 목적상 조사 대상이 된 감리보고서는 소프트웨어 개발 사업에 한 하였으며, 가능한 자료의 일관성을 위해 사업규모가 특별히 작거나 크지 않은 중·소규모 사업만을 (사업금액 30억원 미만이 대부분 임) 주 대상으로 하였다. 따라서 사업별 감리 횟수도 1회(단일감리)로 끝났거나 아니면 1, 2차(중간감리, 최종감리)로 나누어 실시된 감리를 대상으로 조사하였다.

연구에 사용된 자료는 정보시스템 개발사업별로 감리를 실시한 후에 작성된 '감리보고서'이며, 보고서 내의 '개선 및 권고사항'에 명시된 개발 단계별 세부지적사항을 면밀히 조사했다. 감리보고서의 세부지적사항들은 감리인마다 그 표현 방법이 다르고 내용이 지나치게 세부적이거나 추상적이기도 하다. 따라서 세부지적사항들을 표준화된 일정한 틀(항목)에 맞추어 다시 정리하고 지적된 회수를 집계하기 위해 <표 4>과 같이 '지적사항 조사표'를 작성하였다. '지적사항 조사표'에 옮겨진 세부지적사항들은 사업수행연도별, 감리형태별 또는 사업규모별 등으로 집계하고 분석하여, 공공소프트웨어 개발사업의 개발 성향과 정보시스템 감리가 소프트웨어 품질에 미치는 영향을 확인하였다. 통계처리를 위해 사용된 패키지는 SAS Version 8.2이다.

본 논문은 5장으로 구성되어 있다. 제1장은 서론 부분으로 연구 의의 및 목적 그리고 연구방법이 기술되어 있으며, 제2장은 이론적 고찰 부분으로 소프트웨어의 개발 단계, 소프트웨어 품질, 정보시스템 감리, 선행 연구사례 등이 기술되어 있다. 제3장은 연구의 설계 부분으로 가설 및 변수 설정, 조사 범위 및 방법이 기술되어 있으며, 제4장에는 조사결과의 분석 및 가설검증이 포함되어 있다. 제5장은 결론 부분으로 연구 결과의 요약과 연구의 제한점이 기술되어 있다.

본 연구의 틀은 <그림 1>과 같다.



<그림 1> 연구의 틀

2. 이론적 고찰

2.1 소프트웨어 개발 및 품질

2.1.1 소프트웨어 개발

기업의 실제 문제를 해결하기 위해 소프트웨어 개발팀은 소프트웨어 프로세스에, 개발 방법론, 개발 도구 그리고 일반적인 개발 과정을 포함하는 개발 전략을 수립한다. 이러한 전략을 소프트웨어 프로세스 모형(software process model) 또는 소프트웨어 공학 패러다임(software engineering paradigm)이라고 부른다. 소프트웨어 프로세스란 고품질의 소프트웨어를 구축하는 데 요구되는 태스크(task)에 대한 프레임워크(frame work)로 정의된다[Pressman, 1997].

소프트웨어 프로세스 모형은 시대적 흐름과

정보기술의 발전 그리고 대상 업무의 규모와 성격 등에 따라 매우 다양하게 제시되어 왔다. 대표적인 모형으로는 ① 선형순차적 모형(linear sequential model), ② 프로토타이핑 모형(prototyping model), ③ 신속응용개발 모형(rapid application development model), ④ 점증적 모형(incremental model). ⑤ 나선형 모형(spiral model) ⑥ 4세대기술 모형(fourth generation techniques model) 등이 있다. 앞서 제시된 모형들은 각각의 특성과 장점 그리고 적절하게 적용되는 응용시스템이 있기 마련이지만, 소프트웨어 프로세스 모형은 응용시스템의 영역, 프로젝트의 크기, 또는 복잡도와 관계없이 일반적으로 3단계로 구분된다.

첫째는 정의단계(definition phase)로 무엇(what)에 초점을 맞춘다. 이 단계에서는 개발하고자 하는 소프트웨어의 기능적 범위는 어떻게 되며, 처리되는 정보는 무엇이며, 요구되는 성능은 무엇이고, 설계 시 기술적 제한점은 무엇인가 등을 정의하게 된다. 이 단계에서 수행하게 되는 주요 업무는 계획(planning)과 대상업무의 분석(analysis)이라고 할 수 있다.

둘째는 개발단계(development phase)로 방법(how)에 초점을 맞춘다. 이 단계에서 개발자는 데이터는 어떻게 구조화하는지, 요구된 기능은 소프트웨어로 어떻게 구현되는지, 인터페이스는 어떻게 하는지, 세부 처리절차는 어떻게 구현하는지, 테스트는 어떤 방법으로 어느 수준까지 하는지 등을 정의하게 된다. 개발 단계에서 수행되는 업무는 아주 다양하지만 크게 소프트웨어의 설계(design), 코딩(coding), 시험(test) 3단계로 나누어진다.

셋째는 유지보수단계(maintenance phase)로 변경(change)에 초점이 맞추어진다. 개발된 소프트웨어는 운영단계에 들어가며, 환경의 변화에 따라 새로운 적용, 발생된 문제의 수정, 그리고 기능 및 성능의 향상 등이 주 대상업무가 된다.

일반적으로 소프트웨어 개발 프로젝트가 수행되는 경우 사업의 대상은 정의단계와 개발단계가 되며 감리 대상 또한 이 부분이 된다.

2.1.2 소프트웨어 품질

소프트웨어 품질은 크게 소프트웨어 개발 및 관리 프로세스의 품질과 소프트웨어 제품의 품질로 나눌 수 있다. 소프트웨어 개발 및 관리 프로세스의 품질을 측정하는 기준은 여러 가지가 사용되고 있지만 그 중 국제표준 ISO 9000 시리즈가 가장 체계적으로 잘 발전하고 있으며, 소프트웨어 제품의 품질을 평가하는 국제 표준으로는 ISO 9126이 있다.

ISO 8402에서는 품질을, "제품이나 서비스가 가지고 있는 명시적 또는 묵시적 요구를 만족시키는 능력에 관한 특성 및 특징의 전체"라고 정의하고 있다. 여기서 제품이란 ISO에서 정의한 제품의 분류 즉, 하드웨어, 소프트웨어, 가공재료, 서비스가 될 수 있다. 이 품질에 대한 정의에서 "묵시적 요구"라는 말에 주목할 필요가 있다. 문서 형태로 비교적 명확하게 정의된 명시적 요구사항 뿐만 아니라, 공급자 측에서 당연히 해주리라 믿고 명시하지 않은 묵시적 요구사항까지 파악하여 만족시켜 주어야 한다는 것이다.

ISO 8402에서는 품질관리를 "품질에 대한 요구사항을 만족하는데 사용되는 운용기법과 활동"이라고 정의하고 있다. 일반적으로 개발회사는 제품을 고객에게 납품하기 전에 제품검사를 통해 품질목표를 달성하려고 했었다. 그러나 품질관리의 철학은 부적합 제품, 공정, 장비, 서비스가 고객에게 영향을 미치기 전에 찾아내자는 것이다. 품질문제는 종종 검사한 공정이나 활동과는 거리가 먼 요인에서 발생한다. 예를 들어

부적합사항의 근본 원인이 부적절한 교육, 문서 관리의 부실, 업무분장의 불명확 등에서 기인할 수도 있다. 품질관리가 품질확보에 중요한 역할을 담당하고는 있지만 그것만으로 품질이 보증된다고 볼 수는 없다.

ISO 8402에서는 품질보증을 "제품이나 서비스가 주어진 요구사항을 만족함에 대한 적절한 신뢰감을 주는데 필요한 모든 계획적이고 체계적인 활동"이라고 정의하고 있다. 이 정의에서 중요한 용어는 "활동이 계획적이고 체계적 이어야한다"는 것과 이들 활동이 "품질을 확보한다는 데에 믿음을 주어야한다'는 것이다.

IBM의 Humphrey는 소프트웨어 성숙도 프레임웍을 Carnegie Mellon 대학의 소프트웨어공학연구소(SEI : Software Engineering Institute)로 가져와, 조직이 소프트웨어 프로세스 성숙의 어느 단계에 도달했는지를 제시해 주는 성숙도 모델(CMM : Capability Maturity Model for software)을 개발하였다[Humphrey, 1989]. CMM에서 제시하고 있는 프로세스 성숙도는 다섯 단계로 초기(initial)단계, 반복(repeatable)단계, 정의(defined)단계, 관리(managed)단계, 최적화(optimizing)단계로 나눈다.

CMM은 소프트웨어 제품 자체의 품질을 평가하기 보다는 제품을 생산하는 프로세스의 효과적인 관리에 더욱 초점을 맞추고 있다. 이는 CMM의 적용을 통해 지속적인 프로세스 개선이 가능하며 개선된 공정은 제품 품질에 반영된다는 CMM의 기본 가정에 근거한다. 즉, 성숙된 소프트웨어 관리체계를 가진 조직은 ① 소프트웨어의 크기, 비용, 품질 그리고 개발일정의 예측력 향상, ② 프로세스 관리를 통한 제품의 결과 향상, ③ 소프트웨어 개발과정의 가시성 증대와 같은 조직적 이득을 가져온다는 것이다[Gaffney, 1995]. 이와 같이 품질에 관련된 활동은 소프트웨어 개발 프로세스 중 계획은 물론 분석 및 설계 그리고 구현 등 모든 단계에서 반영되어야 한다는 점이 중요하다.

2.2 정보시스템 감리

2.2.1 감리의 정의

국내의 정보시스템 감리제도는 1987년 행정전산망에 대한 한국전산원의 감리가 시행되어 온 이후 국내의 실정에 맞게 발전되어 오고 있다.

우리나라의 '정보화촉진기본법 제15조의 2항'의 '정보시스템감리기준'에 따르면, 정보시스템 감리는 "정보시스템의 효율성, 효과성, 안전성(신뢰성) 달성 여부를 독립적으로 평가하여 문제점의 개선을 권고하는 활동"이라고 정의하고 있다.

미국의 ISACA는 정보시스템 감리를 "자동화된 정보처리시스템의 모든 측면 또는 특정부문을 검토하고 평가하는 각종 활동"이라고 정의하면서 시스템적인 성격 즉, 신뢰성, 유용성 및 합법성 측면을 강조하고 있다.

일본의 '감사기준'에서는 정보시스템 감리를 "감리대상으로부터 독립된 객관적인 입장에서 컴퓨터를 중심으로 하는 정보시스템을 종합적으로 점검 및 평가하여 관계자에게 조언하고 권고하는 것으로, 정보시스템의 유효한 이용의 촉진과 폐해 제거를 동시에 추구하며 건전한 정보화를 도모하는 것"이라고 정의하고 있다.

감리와 유사하게 사용되는 용어로 감사가 있다. 사전적 의미로만 보자면 감사는 '감독하고 조사하는 것'이며 감리는 '감독하고 관리하는 것'이다. 따라서 감사의 특징은 오류의 지적과 시정에 역점을 두고 있으며, 정당성과 합법성을 강조하고, 계약의 준수나 처리의 정확성 등을 강조한다. 그러나 감리의 특징은 관리 및 엔지니어링에 역점을 두고 있으며 따라서 기획, 계획, 설계, 엔지니어링을 강조하고 정당성, 합리성, 적

정성을 강조하는데 있다. 감사가 사후평가방식으로 법에 의한 처벌과 의사결정에 유용한 정보로 활용될 수 있다면, 감리는 사전, 진행, 사후 평가방식으로 엔지니어링 의사결정에 유용한 정보로 사용할 수 있는 점이 다르다.

본 연구에서는 사전적 의미에만 연연하여 감사와 감리를 별도의 개념과 행위로 보지 않고, 감리는 감사를 포함하는 광의의 개념으로 인식하고자 한다.

2.2.2 감리의 목적

정보시스템에 관련된 문제점으로는 정보자원의 효율적인 이용 등 투자를 중심으로 하는 경영상의 문제, 그리고 컴퓨터 범죄와 오류에 관련된 안전 및 보안상의 문제, 사생활 보호 문제 등이 두드러지게 드러난다. 정보시스템 감리는 이러한 여러 문제들을 종합적이고 합리적으로 해결하는 수단으로서 낭비를 제거하고 정보시스템의 역기능에 의한 피해 발생을 방지할 수 있다. 또한 정보시스템 개발 시 품질이 확보된 시스템 개발을 실현하여 궁극적으로 정보시스템의 효과를 달성할 수 있도록 해 준다. 정보시스템 감리의 목적을 세분화하여 정리하면 다음과 같이 다섯 가지로 나눌 수 있다.

첫째, 정보시스템의 효과성 증진이다. 즉, 소프트웨어 개발 과정 중 기획업무의 타당성 검토, 소프트웨어 개발 공정의 적합성 검토, 개발된 소프트웨어의 품질 보증 그리고 감리를 통하여 정보시스템의 효과를 증진시킬 수 있다.

둘째, 정보시스템의 효율성 증진이다. 정보시스템 감리는 정보시스템을 개발하고 운영하는 조직 중에 어디에 문제점이 있는지, 그리고 무엇을 우선적으로 개선해야 하는지를 제안하고 권고하여 정보시스템의 효율성을 증진시키는데 목적이 있다.

셋째, 정보시스템의 안전성을 확보하는데 있다. 정보시스템 안전성에는 데이터의 무결성, 기밀성 및 가용성이 포함된다. 특히 데이터의 무결성은 정보시스템의 효과를 보장하는 가장 중요한 요인인 동시에 정보시스템 감리가 보장하는 특성이기도 하다.

넷째, 정보시스템의 개발과 운영에 관련된 각종 법, 규정, 지침 또는 표준 등에 대한 준수 여부를 확인함으로서 준거성을 확보에 도움을 줄 수 있다는 것이다.

다섯째, 감리를 통해 사용자와 개발 및 운영자들 간에 상호 이해를 증진시키고 정보시스템과 관련된 객관적인 정보를 제공함으로서, 관련자들 간의 이해를 증진시켜 정보시스템의 전 생명주기 단계에서 정확한 개발이 이루어질 수 있게 한다.

2.2.3 감리의 구분

정보시스템 감리는 감리 내용에 따라 감리 형태를 '사업감리'와 '운영감리'로 나누고, 감리 시점에 따라 감리 종류를 '단일감리', '중간감리', '최종감리'로 나눈다. '사업감리'는 주로 프로젝트 방식으로 추진되는 정보시스템 개발 사업이 대상이 되며, '운영감리'는 개발된 정보시스템에 대한 운영과 유지보수가 대상이 된다. '단일감리'는 정보시스템이 개발되어 인도되기 직전에 1회만 수행되는 감리를 말한다. 감리를 2회 이상 실시하는 사업 중 '중간감리'는 보통 정보시스템의 설계단계가 끝난 후 실시되고, '최종감리'는 정보시스템 개발이 완료되어 사용자에 인도되기 직전에 실시된다. 그러나 사업의 규모에 따라 감리는 여러 차례 실시될 수도 있다.

감리 형태에 따른 감리 내용과 감리 목적은 <표 1>과 같다[문대원, 1999].

〈표 1〉 감리 형태에 따른 감리 내용 및 목적

감리 형태	감리 내용	감리 목적
사업감리	◦ 정보시스템 계획 ◦ 응용시스템 분석 · 설계 감리 ◦ 응용시스템 구현 감리 ◦ 시스템 시험 및 통합 감리	개발사업의 성공적인 수행을 목적으로 하며, 개발사업의 진행단계에 따라 실시
운영감리	◦ 시스템 통제 준거성 감리 ◦ 시스템 안전성 감리 ◦ 시스템 효율성 감리 ◦ 시스템 효과성 감리	정보시스템의 설비, 응용소프트웨어 운영, 오류 및 보안 등을 점검하는 감리로서 주기적 또는 특별한 사안이 발생하면 실시

2.3 선행 연구 사례

소프트웨어는 일반적으로 제조업과는 달리 기술이 인간중심으로 이루어져 있기 때문에 의학이나 물리학처럼 단지 실험만을 통해서 품질에 관한 모형을 세우고 검증하기는 대단히 어렵다. 다른 학문분야와 마찬가지로 소프트웨어에도 여러 가지 원인과 학습 그리고 대상 업무에 대한 이해 등 결과에 대한 수많은 변수들이 존재한다. 이에 따라 아직도 소프트웨어공학 분야에는 그 결과에 대한 이유를 명료하게 설명할 수 있는 모형이 거의 없으며, 특정 환경에서는 기술의 한계에 대한 인식이 부족하고, 분석이나 실험도 충분하지 못한 형편이다[Basili, 1998].

R. S. Pressman은 소프트웨어 개발 시 정의(definition)와 개발(development) 단계에서 노력의 분배는 일반적으로 40-20-40의 법칙을 적용할 것을 제시하였다. 노력의 40% 이상은 분석과 설계에 배분되어야 하고, 20%는 구현(coding)에 분배되어야 하며, 나머지 40% 정도는 시험(testing)에 분배할 것을 권유하고 있다. 이를 좀더 세분화 시키면 프로젝트 계획에 2%~3%, 요구사항 분석에 10%~25%, 그리고 설계에 20%~25%를 분배하고 있다. 구현은 분석과 설계의 결과에 따라 실시되는 것이므로 약 15%~20% 정도로 설계에 비해 상대적으로 노력이 적게 투입된다. 소프트웨어의 시험과 그에 뒤따르는 디버깅(debugging)은 30%~40%의 노력이 필요하며, 소프트웨어가 사람의 생명과 관련이 깊은 것일수록 그 노력의 비중은 더해져야 한다고 주장하였다. 또한 Pressman은 소프트웨어에서 발생하는 오류(error)도 Pareto 법칙이 적용됨(소프트웨어 오류의 80%는 20% 범위 내에서 발생함)을 주장하였으며, 소프트웨어 신뢰도를 높이기 위해서는 오류가 많이 발생하는 부분을 집중적으로 검토하고 시험할 필요가 있다는 점을 강조하였다[Pressman, 1997].

정보시스템에 대한 종래의 연구에서는 소프트웨어 개발과정에서 사용자들의 심리적이고 행동과학적인 상태 관리의 중요성이 제기되어 왔다. 이와 같은 연구에서 중요한 발견은 사용자의 관여와 참여가 정보시스템의 제품과 서비스 측면에서 긍정적인 영향을 미친다는 사실을 알게 되었다는 점이다[Hartwick, 1994].

T. Ravichandran과 Arun Rai는 정보시스템 개발에서 품질 향상에 대한 중요한 발견을 하였는데, 그것은 품질에 대한 산발적인 노력은 정보시스템의 품질 향상에 큰 영향을 미치지 못하며, 산발적인 노력들의 상호작용을 결합하여 조직적인 시스템을 형성해야만 품질 향상을 기할 수 있다는 점이다[Ravichandran, 1996].

한국전산원은 "감리결과 분석을 통한 주요 문제점 및 개선사례 연구"에서 자체적으로 수행했던 26건의 감리보고서를 조사했다. 조사대

상 소프트웨어 프로젝트를 '프로젝트 관리', '응용시스템', '데이터베이스 및 자료 변환' 등 3분야로 나누고 각 분야별로 사업수행단계를 4단계 즉, '계획', '분석', '설계', '구현'으로 나누어 각 단계에서 지적된 주요 문제점과 개선권고사항을 분석하였다[한국전산원, 2001].

'프로젝트 관리'에 있어 사업수행단계별로 지적된 문제점은 총 158건으로, 계획단계가 10건, 분석단계가 8건, 설계단계가 32건, 구현단계가 108건으로 나타났다. '응용시스템'에 있어 사업수행단계별로 지적된 문제점은 총 96건으로, 계획단계에서는 지적사항이 없으며, 분석단계 2건, 설계단계 9건, 구현단계 85건으로 구현단계에서 많은 지적사항이 발생되었다. '데이터베이스 및 자료 변환'에 있어 사업수행단계별로 지적된 문제점은 총 157건으로 계획단계 4건, 분석단계 7건, 설계단계 43건, 구현단계 103건으로 대부분 구현단계에서 지적사항이 발생되었다.

문대원은 1996년부터 2000년까지 시행된 124건의 감리보고서를 대상으로 감리분야를 대구분, 중구분으로 나눈 뒤 세부항목별로 분석하여 정보시스템 개발 프로젝트의 성공요인을 실증적으로 식별하였다. 프로젝트의 성공여부는 감리결과를 정량화 하여 결정하였으며, 그 결과 15항목의 성공변수를 중요한 순서대로 판별하였다[문대원, 2001].

이상엽은 "소프트웨어 프로세스 성숙도가 프로젝트 성과에 미치는 영향"에서, 프로세스 성숙도가 개선되면 프로젝트의 납기 준수율과 프로젝트 규모의 예측율에 정(+)의 영향을 미침을 발견하였다. 그리고 프로세스 성숙도는 고객 의사소통을 통하여 프로젝트 성과에 간접적인 영향을 미침을 알 수 있었다[이상엽, 2000].

김용경은 "정보시스템 감리가 소프트웨어 품질에 미치는 영향"에서 소프트웨어 개발 사업의 감리보고서 74건에 나타난 1301회의 지적사항들을 조사·분석한 결과, 감리가 소프트웨어 품질 향상에 긍정적인 영향을 미치고 있음을 확인하였다[김용경, 2002].

3. 연구의 설계

3.1 가설 및 변수의 설정

3.1.1 가설의 설정

본 연구에서는 우리나라 공공소프트웨어는 개발 단계에서 품질 문제가 매년 얼마나 중요하게 고려되고 있는가? 사용자의 편리성은 매년 얼마나 중요하게 고려되고 있는가? 개발단계의 어느 부분에서 가장 많은 지적사항이 발생하고 있으며, 지적사항의 발생은 파레토 법칙을 따르는가? 등 즉, 공공소프트웨어의 개발 성향을 알아보고 그리고, 공공소프트웨어 개발 사업에 대한 감리가 공공소프트웨어의 품질에 결국 어떤 영향을 미치는가를 실증적으로 연구하기 위해 다음과 같이 4개 항목의 가설을 설정하였다.

[가설 1] 공공소프트웨어 사업연도와 품질 관련 세부항목의 지적회수는 음(-)의 상관관계가 있을 것이다. 즉, 공공소프트웨어 사업연도가 높아질수록 품질 문제가 중요하게 고려될 것이다.

[가설 2] 감리 결과의 지적사항 항목에 파레토 법칙(Pareto principle)이 적용될 것이다. 즉, 소프트웨어 개발단계 중 특정 부분(20%의 세부항목)에 80%의 지적회수가 집중될 것이다.

[가설 3] 공공소프트웨어 사업연도와 사용자 편리성 관련 세부항목의 지적회수는 음(-)의 상관관계가 있을 것이다. 즉, 공공소프트웨어 사업연도가 높아질수록 소프트웨어 개발에 사용자(end-user)의 편리성이 강하게 반영될 것이다.

[가설 4] 정보시스템 사업감리가 공공소프트웨어의 품질에 긍정적인 영향을 미칠 것이다.

① 단일감리나 중간감리의 지적회수보다 최종감리의 지적회수가 상대적으로 적게 나타날 것이다.

② 감리가 실시될수록(최근년도일수록) 공공소프트웨어 사업의 전체 지적회수는 상대적으로 감소될 것이다.

3.1.2 변수의 설정

본 연구에 사용된 독립변수는 사업 수행연도(1998, 1999, 2000, 2001, 2002, 2003)와 감리 종류(단일감리, 중간감리, 최종감리), 사업기간 및 사업금액 그리고 각 개발단계의 세부지적사항 등이다. 또한 종속변수는 사업 수행연도, 감리 종류, 사업기간, 사업금액에 따른 각 개발 단계(프로젝트 계획 및 관리, 분석, 설계, 구현, 시험)의 세부지적사항별 지적회수가 사용되었다.

3.2 조사 범위 및 방법

3.2.1 조사 범위

본 연구에서는 국내의 전문 감리업체인 H감리원과 Y감리원에서 1998년부터 2003년까지 6년 동안에 실시된 공공기관의 정보시스템 개발사업 107건에 대한 감리실적(보고서) 총 168 건을 대상으로 조사를 실시했다.

조사는 정보시스템 개발사업에 대한 사업감리만을 대상으로 하였으며, 단일감리와 중간감리, 최종감리가 모두 포함되어 있다. 자료의 일관성을 위해 사업규모가 특별히 크지 않은 사업을 대상으로 하였으며, 따라서 사업별 감리회수도 1회(단일감리)로 끝났거나 아니면 1, 2차(중간감리, 최종감리)로 나누어 실시된 감리를 대상으로 조사하였다. 조사 대상의 정보시스템 개발 사업 107건 중 감리를 1회만 실시한 사업은 42개이며, 2회(중간감리와 최종감리)를 같이 실시한 사업은 61개이고, 중간감리만 실시한 사업은 4개이다. 따라서 전체 감리회수는 168회(42 + 61 * 2 + 4)가 된다. 중간감리 회수와 최종감리 회수에 차이가 있는 것은, 감리는 2회 실시하였으나 최종감리의 자료수집 과정에서 누락이 발생한 것으로 보인다. 년도 별 감리 회수는 <표 2>과 같다.

〈표 2〉 년도별 감리 회수

구분 / 년도	전 체	단일감리	중간감리	최종감리
1998	6	6	0	0
1999	14	10	2	2
2000	25	5	10	10
2001	46	21	14	11
2002	46	0	23	23
2003	31	0	16	15
계	168	42	65	61

사업규모별 감리회수는 <표 3>와 같다. 총 감리회수가 161건으로 연도별 총 감리회수 168건과 7건의 차이가 나는 것은, 감리보고서에 사업금액이 명시되어 있지 않았거나 단돈 1원으로 표기되어 있는 사업도 있기 때문이다.

〈표 3〉 사업규모별 감리 회수

구분 / 금액(원)	전 체	단일감리	중간감리	최종감리
1억 미만	22	12	5	5
1억~5억	51	13	19	19
5억~10억	38	7	16	15
10억~15억	20	3	10	7
15억~20억	5	3	1	1
20억~30억	12	0	6	6
30억 이상	13	3	5	5
계	161	41	62	58

3.2.2 조사 및 분석 방법

먼저 H감리원, Y감리원에서 정보시스템 개발 사업의 감리보고서 168건을 입수하여 보고서 내용 중 '개선권고사항'을 중점적으로 조사했다. '개

〈표 4〉 지적사항 조사표

Project명			감리구분				
수행기간	-	사업금액		감리회사			
개발 단계	지적 회수	세 부 항 목		지적정도			원인
				◎	○	△	
프로젝트 계획 및 관리 (Planing & Control) "P"		01. 범위계획					
		02. 일정계획					
		03. 비용계획					
		04. 자원계획					
		05. 품질계획					
		06. 위험계획					
		07. 시험계획					
		08. 유지보수계획					
		09. 사용자 교육계획					
		10. 개발방법론 적용					
		11. 기타					
분석 "A"		01. 시스템(요구사항)의 타당성 검토					
		02. 시스템(요구사항) 분석 내용의 정확성					
		03. 시스템(요구사항)의 내부적 일관성					
		04. 시스템(요구사항)의 추적 가능성					
		05. 시스템(요구사항)의 설계, 구현, 유지보수 가능성					
		06. 위험분석(risk analysis) 여부					
		07. 분석 결과의 문서화 및 관리					
		08. 기타					
설계 "D"		01. 요구사항 분석 내용의 설계 반영					
		02. 데이터 설계					
		03. 데이터베이스 설계					
		04. 시스템 구조 설계(개략설계)					
		05. 내부 인터페이스 설계(모듈-모듈)					
		06. 외부 인터페이스 설계(소프트웨어-사람)					
		07. 처리절차 설계(상세설계)					
		08. 시험 설계					
		09. 보안 및 품질 설계					
		10. 타 시스템과 연결 설계					
		11. 설계자료(명세서)의 문서화 및 관리					
		12. 기타					
구현 "C"		01. 소프트웨어 오류					
		02. 기능 누락					
		03. 사용자 편리성					
		04. 사용 프로그램 언어					
		05. 프로그램 내 문서화					
		06. Source Program 관리					
		07. 운영지침서 작성					
		08. 기타					
시험 "T"		01. 소프트웨어 단위시험					
		02. 소프트웨어 통합시험					
		03. 요구사항 검증시험					
		04. 시스템시험					
		05. 수락시험					
		06. 신뢰도 검증					
		07. 타 시스템과의 연결시험					
		08. 기타					
계		지적정도 : ◎ 긴급개선 ○ 통상개선 △ 권고사항					

선권고사항'은 크게 4분야(프로젝트관리 및 품질, 응용시스템, 데이터 및 데이터베이스, 아키텍쳐 및 보안)로 나뉘어 있으며 각 분야마다 다수의 세부항목으로 지적사항을 기술하고 있다. 감리보고서에 명시된 '개선권고사항'의 세부항목들은 감리 전문기관의 공인된 전문가들에 의해 지적된 사항이므로 그 내용의 타당성이나 신뢰성에 대해서는 문제가 없을 것으로 판단된다.

감리보고서의 '개선권고사항'에서 지적된 세부항목들은 감리인마다 표현 방법이 다소 다를 수 있음은 물론 그 내용이 지나치게 세부적이거나 아니면 추상적일 수도 있다. 따라서 본 연구에서는 감리보고서에 명시된 개선권고사항의 세부항목들을 일정한 틀 즉, 표준화된 양식(항목)에 맞추어 다시 정리하고 지적된 회수를 집계하기 위해서 <표 4>와 같이 '지적사항 조사표'를 작성하였다.

'지적사항 조사표'의 개발 단계는 소프트웨어 선형순차형모형(Linear Sequential Model)에 나타나는 5단계를 기초로 하였다. 각각의 개발 단계는 다시 여러 개의 세부사항으로 나뉘는데, 이것은 문대원, 장시영이 제시한 '정보시스템 감리지침'의 검토 항목과[문대원, 1999], 김용경이 표준화하여 제시한 선행연구자료[김용경, 2002], 그리고 감리 실무자 또는 대학의 소프트웨어공학 분야 교수들의 의견을 모아 선정하고 정리한 것이다. '지적사항 조사표'에는 공통사항으로 사업명, 사업기간, 사업금액, 감리회사, 감리종류가 기입된다. 개발 단계의 세부항목에는 지적내용의 중요성과 긴급성을 반영하여 지적정도를 '긴급개선', '통상개선', '권고사항'으로 나누어 지적회수를 기입하도록 하였다. 여기서 '긴급개선'은 지적된 사항이 중대하여 프로젝트 다음 단계를 위해 시급하게 수정 또는 보완을 해야 할 사항을 의미한다. '통상개선'은 긴급개선에 비해서는 덜 중대하고 시급하나 시간을 갖고 수정 또는 보완을 해야 할 사항이다. '권고사항'은 프로젝트 다음 단계를 위해 관련자 간 협의에 의해 추진할 사항이다.

지적사항 조사표에 기입된 세부항목별 지적회수는 조사 결과를 분석하고 가설을 검증하기 위해 통계처리 하였다. 먼저 감리 결과 지적사항의 총괄현황을 알아보기 위해 감리종류별, 사업연도별, 사업규모별, 개발 단계별, 세부 지적항목 등으로 다양하게 빈도분석을 실시했다. 또한 사업의 성격, 감리회사, 사업규모에 따라 지적회수에 유의한 차이가 있는가를 알아보기 위해서는 분산분석, 차이t검정 방법을 사용했다. 특히 본 연구에서 설정한 [가설 1], [가설 3], [가설 4]를 검증하기 위해서 즉, 사업연도에 따라 품질 관련 항목과 사용자 편리성 항목 그리고 전체 지적회수에 유의한 변화가 있는가를 알아보기 위해서는 회귀분석을 실시했다. 본 연구에 사용된 통계패키지는 SAS Version 8.2 이다.

4. 조사결과의 분석 및 가설 검증

4.1 지적사항 조사 및 분석 결과

4.1.1 총괄 현황

지적사항 조사표에 기입된 세부항목별 지적회수를 집계하여 공공소프트웨어 개발 사업에 대한 지적현황을 총괄적으로 알아보기 위해 빈도분석을 실시했다.

<표 5>에서 보는 것과 같이 단일감리와 중간감리 그리고 최종감리 모두를 합한 전체 168건의 감리 결과, 지적된 세부 지적항목 수 즉 지적회수는 총 3027회이며 이 중 긴급개선사항은 966회, 통상개선사항은 1766회, 권고사항은 295회이다. 전체 3027회의 지적건수 중 계획단계는 683회(23%), 분석단계는 388회(13%), 설계단계

는 1385회(46%), 구현단계는 373회(12%), 시험단계는 198회(7%)이다.

공공소프트웨어 개발 사업은 사업 당 설계단계에서 평균 8.24회의 지적이 발생했으며, 다음으로 계획단계의 4.06회, 분석단계의 2.3회 순으로 발생하고 있음을 알 수 있다. 감리 1회당 전체 평균 지적회수는 18.01회이며, 단일감리에서는 20.02회, 중간감리에서는 18.04회, 최종감리에서는 16.48회의 지적이 발생했다. 사업 당 평균 지적회수에 있어 단일감리보다 중간감리, 중간감리보다 최종감리의 지적회수가 적게 발생했다는 것을 알 수 있다.

〈표 5〉 감리 지적회수 총괄 현황

구분 / 단계	전 체				단일감리				중간감리				최종감리			
	감리회수 : 168				감리회수 : 42				감리회수 : 65				감리회수 : 61			
	계 (%)	긴급	통상	권고	계 (%)	긴급	통상	권고	계 (%)	긴급	통상	권고	계 (%)	긴급	통상	권고
합계	3,027 (100)	966	1766	295	849 (100)	359	402	88	1,173 (100)	345	727	101	1,005 (100)	262	637	106
계획	683 (23)	227	374	82	188 (22)	79	82	27	291 (25)	91	173	27	204 (20)	57	119	28
분석	388 (13)	103	251	34	114 (13)	37	63	14	196 (17)	55	126	15	78 (8)	11	62	5
설계	1385 (46)	414	838	133	364 (43)	150	181	33	604 (51)	175	380	49	417 (41)	89	277	51
구현	373 (12)	145	194	34	115 (14)	60	47	8	53 (5)	15	30	8	205 (20)	70	117	18
시험	198 (7)	77	109	12	68 (8)	33	29	6	29 (2)	9	18	2	101 (10)	35	62	4

4.2 개발 단계별 지적 현황

4.2.1 계획 및 관리 단계

계획 및 관리단계의 전체 지적회수(단일감리, 중간감리, 최종감리의 긴급, 통상, 권고를 모두 합한 것) 683회이다. 세부항목의 지적회수에서는 일정관리에 153회로 가장 많은 지적을 받았으며, 다음으로 품질계획에 92회, 범위계획에 78회의 순으로 지적을 받았다. 이를 통해 우리나라 공공소프트웨어 개발 사업은 일정이 잘 지켜지고 있지 않음을 알 수 있다.

〈표 6〉 계획단계의 세부항목별 지적회수

세부항목	합 계	전 체		
		긴급	통상	권고
합 계	683	227	374	82
범위계획	78	39	32	7
일정계획	153	84	59	10
비용계획	1	1	0	0
자원계획	59	10	35	14
품질계획	92	18	73	1
위험계획	52	17	30	5
시험계획	50	15	30	5
유지보수계획	45	11	24	10
개발자교육계획	52	4	40	8
개발방법론적용	51	17	23	11
기 타	50	11	28	11

4.2.2 분석단계

분석단계의 전체 지적회수는 388회로 긴급개선이 103회, 통상개선이 251회, 권고사항이 34회로 나타났다. 총 388회의 지적사항 중 세부항목별 지적회수에서는 '분석결과의 문서화 및 관리'에 133회, '시스템(요구사항) 분석내용의 정확성'에 60회, 위험분석(risk analysis) 여부에 52회의 순서로 지적되었다.

'분석 결과의 문서화 및 관리' 항목에서 가장 많은 지적이 발생하고 있는 것은, 이를 통해 공공소프트웨어 개발 과정에서 담당자들이 문서화를 중요시 않고 있거나, 공공소프트웨어 개발 과정에서 단계별로 정형화된 문서화의 모형을 사용하고 있지 않은 결과라고 분석된다.

〈표 7〉 분석단계의 세부항목별 지적회수

세 부 항 목	합계	전 체		
		긴급	통상	권고
합 계	388	103	251	34
시스템(요구사항)의 타당성 검토	33	12	18	3
시스템(요구사항) 분석 내용의 정확성	60	25	30	5
시스템(요구사항)의 내부적 일관성	18	6	12	0
시스템(요구사항)의 추적 가능성	37	8	25	4
시스템(요구사항)의 설계, 구현, 유지보수 가능성	40	9	25	6
위험분석(risk analysis) 여부	52	13	31	8
분석 결과의 문서화 및 관리	33	29	98	6
기 타	15	1	12	2

4.2.3 설계단계

설계단계의 전체 지적회수는 1385회로 공공소프트웨어의 개발 단계 중 가장 높은 비중을 차지하고 있다. 설계단계의 세부항목별 지적회수에서는 '보안 및 품질 설계' 항목에서 234회의 지적을 받았으며, '설계 자료의 문서화 및 관리' 항목에서 230회의 지적을 받았다. 다음으로 지적을 많이 받은 세부항목은 '데이터베이스 설계'로 228회의 지적을 받았으며, '데이터 설계' 항목은 219회의 지적을 받았다.

'보안 및 품질 설계' 항목에서 가장 많은 지적을 받고 있는 것은, 계획단계에서 '품질계획'에 가장 많은 지적을 받은 것과 무관하지 않은 것으로 생각되며, 공공소프트웨어를 개발하는데 있어 품질과 보안에 좀 더 많은 관심을 기울여야 할 필요성을 보여주는 것으로 해석된다. '설계자료(명세서)의 문서화 및 관리' 항목이 두 번째로 많은 지적을 받고 있는 것은, 분석단계에서 '분석결과의 문서화 및 관리' 항목이 가장 많은 지적을 받은 것과 유사한 이유로 해석된다.

〈표 8〉 설계단계의 세부항목별 지적회수

세 부 항 목	합계	전 체		
		긴급	통상	권고
합 계	1385	414	838	133
요구사항 분석내용의 설계반영	89	30	56	3
데이터 설계	219	98	96	25
데이터베이스 설계	228	80	126	22
시스템 구조 설계(개략설계)	62	17	39	6
내부 인터페이스 설계(모듈-모듈)	18	4	10	4
외부 인터페이스 설계 (소프트웨어-사람)	57	15	35	7
처리절차 설계(상세설계)	89	31	55	3
시험 설계	73	22	46	5
보안 및 품질 설계	234	51	153	30
타 시스템과의 연결 설계	48	14	26	8
설계자료(명세서)의 문서화 및 관리	230	45	172	13
기 타	38	7	24	7

4.2.4 구현단계

구현단계의 지적회수는 전체 373회로, 긴급개선 145회, 통상개선 194회, 권고사항 34회로 나타났다. <표 5>에 나타나 있는 것과 같이 구현단계의 전체 지적사항은 중감감리 보다 최종감리에서 많이 발생하고 있는데, 그것은 대부분 중간감리는 설계를 마친 후(구현 전)에 실시되고 최종감리는 대부분 구현 및 시험을 마친 후

실시되기 때문인 것으로 분석된다.

<표 9>에서 보는 것과 같이 구현단계의 전체 지적회수 373회 중 '기능 누락'이 93건으로 가장 많으며, '운영지침서 작성'이 73회, '사용자 편리성'이 71회의 순으로 나타났다. '기능 누락' 항목이 가장 많이 지적되고 있는 것은, 설계단계에서 많이 지적되고 있는 '처리절차 설계(상세설계)'상의 문제는 물론, '설계 자료의 문서화 및 관리'에서 누락되었거나 잘못 표기된 문제점들이 구현단계의 '기능 누락'으로 이어진 결과라고 분석된다.

〈표 9〉 구현단계의 세부항목별 지적회수

세 부 항 목	합계	전 체		
		긴급	통상	권고
합 계	373	145	194	34
소프트웨어오류	55	43	11	1
기능누락	93	59	30	4
사용자편리성	71	22	38	11
사용자프로그램언어	4	0	3	1
프로그램내문서화	8	0	8	0
SourceProgram관리	40	4	35	1
운영지침서작성	73	10	50	13
기 타	29	7	19	3

4.2.5 시험단계

시험단계의 지적회수는 전체 198회로 긴급개선 77회, 통상개선 109회, 권고사항 12회로 나타났다. <표 5>에서 보는 것과 같이 시험단계의 지적사항이 중간감리에서는 현저하게 적게 나타나고 있는 것은 구현단계와 마찬가지로 중간감리는 설계가 완료된 후에 실시되기 때문인 것으로 분석된다.

<표 10>에서 보는 것과 같이 전체 198회의 지적사항 중 '시스템시험'에 63회로 가장 많은 지적이 발생했으며, '신뢰도 검증'에 27회, '소프트웨어 통합시험'에 22회의 순서로 지적사항이 발생했다. '시스템시험'에 많은 지적사항이 발생하고 있는 것은 공공소프트웨어를 개발하는 과정에서 다른 시스템 개발 담당자들과의 업무적 대화는 물론 동일한 시스템을 개발하는 담당자들과도 업무적 대화가 잘 이루어지지 않고 있는 결과라고 해석된다. 또한 '신뢰도 검증'에서 많은 지적이 발생하고 있는 것은 '신뢰도 검증' 그 자체가 가지고 있는 소프트웨어 공학적인 기술의 어려움 때문인 것으로 해석된다.

〈표 10〉 시험단계의 세부항목별 지적회수

세 부 항 목	합계	전 체		
		긴급	통상	권고
합 계	198	77	109	12
소프트웨어단위시험	21	10	11	0
소프트웨어통합시험	22	7	14	1
요구사항검증시험	17	7	8	2
시스템시험	63	21	37	5
수락시험	17	7	10	0
신뢰도검증	27	7	17	3
타시스템과의연결시험	12	8	4	0
기 타	19	10	8	1

4.3 가설의 검증

4.3.1 [가설 1]의 검증

공공소프트웨어 개발사업이 수행된 연도별로 품질문제가 얼마나 중요하게 고려되고 있는가를 알아보기 위해, 각 개발 단계별로 소프트웨어 품질과 직접 관련이 있는 세부항목을 선정하여 연도별 지적회수를 통계처리 했다. 선정된 세부항목은 계획단계에서는 '품질계획(P05)', 분석단계에서는 '위험분석(risk analysis) 여부(A06)', 설계단계에서는 '보안 및 품질설계(D09)', 구현단계에서는 '소프트웨어 오류(C01)', '기능 누락(C02)', '사용자 편리성(C03)'이며 테스트 단계는

전체 세부항목(T01-08)이 대상이었다.

<표 11>에서 보는 것과 같이 선정된 세부항목 중 여러 항목에서 연도별로 사업 당 평균 지적회수가 뚜렷하게 감소되고 있음을 알 수 있다. 또한 사업연도별 지적회수를 통계처리(회귀분석) 한 결과 특히 A06과 D09 항목은, 사업연도와 품질 관련 세부항목의 지적회수와는 1% 이하의 유의수준으로 음(-)의 상관관계를 나타내고 있으며, C03 항목은 5% 이하의 유의수준으로 음(-)의 상관관계가 있음을 알 수 있다. 특히 선정된 세부항목의 합계에서도 사업연도별로 사업 당 평균 지적회수가 1%이하의 수준으로 매우 유의하게 영향(t=-2.98)을 미치고 있어 본 가설을 지지하고 있다. 그리고 <표 11>에는 직접 명시되지 않았지만, 긴급개선에 3점, 통상개선에 2점 권고사항에 1점을 가중치로 부여한 후 통계처리를 해도 같은 결과를 보여주었다. 이에 따라 '공공소프트웨어 사업연도와 품질 관련 세부항목의 지적회수는 음(-)의 상관관계가 있을 것이다. 즉, 공공소프트웨어 사업연도가 높아질수록 품질 문제가 강하게 반영될 것이다' 라는 [가설 1]은 채택되었다.

이러한 결과는 공공소프트웨어 개발사업이 최근 연도로 올수록 개발 과정의 각 단계마다 품질문제가 중요하게 고려되고 있는 성향을 나타내고 있으며, 그 결과 감리과정에서 지적회수가 점차 적게 발생하고 있는 것이라고 해석된다.

〈표 11〉 연도별 품질 관련 항목의 지적회수

( )는 사업 당 평균 지적회수

세 부 사 항	합 계	사 업 연 도						
		1998	1999	2000	2001	2002	2003	비 고
		6	14	25	46	46	31	사 업 수
합 계	78 (4.7)	38 (6.33)	78 (5.57)	126 (5.04)	248 (5.39)	169 (3.67)	124 (4.0)	t = -2.98***
품질계획(P05)	92 (0.5)	3 (0.5)	8 (0.57)	17 (0.68)	22 (0.48)	22 (0.48)	20 (0.64)	
위험분석여부(A06)	52 (0.3)	4 (0.67)	10 (0.71)	6 (0.24)	18 (0.39)	8 (0.17)	6 (0.19)	t = -2.98***
보안및품질설계(D09)	232 (1.3)	8 (1.33)	24 (1.71)	43 (1.72)	75 (1.63)	49 (1.07)	33 (1.06)	t = -2.59***
소프트웨어오류(C01)	54 (0.3)	5 (0.83)	5 (1.25)	8 (0.32)	13 (0.28)	12 (0.26)	11 (0.35)	
기능누락(C02)	91 (0.5)	4 (0.67)	7 (0.5)	10 (0.4)	25 (0.54)	24 (0.52)	21 (0.68)	
사용자편리성(C03)	71 (0.4)	5 (0.83)	7 (0.5)	12 (0.48)	22 (0.48)	15 (0.33)	10 (0.32)	t = -1.94*
시험단계전체항목(T)	191 (1.14)	9 (1.5)	17 (1.21)	30 (1.2)	73 (1.59)	39 (0.85)	23 (0.74)	

주) ***, **, * 각각 1%, 5%, 10% 이하 수준에서 통계적으로 유의

4.3.2 [가설 2]의 검증

사회현상을 통계적으로 설명하는데 많이 사용되는 파레토 법칙(20 : 80 법칙)은 전체 원인 가운데 20%가 전체 결과의 80%를 만들어 낸다

는 법칙이다.

우리나라 공공소프트웨어 개발사업에서는 어느 세부항목을 가장 많이 지적받고 있는가를 알아보고, 또한 공공소프트웨어 개발사업의 감리결과 지적사항에서도 파레토 법칙이 적용되는가를 확인하기 위해 지적회수가 많은 세부항목 순으로 정렬을 시켜보았다. 지적을 많이 받고 있다는 것은 그 부분이 공공소프트웨어를 개발하는데 있어 취약한 부분이라고 할 수 있으며, 그 부분을 집중적으로 관리한다면 공공소프트웨어의 품질은 개발단계에서부터 효과적으로 향상시킬 수 있게 된다.

<표 12>에서 보는 것과 같이 지적회수가 가장 많은 세부 항목은 설계단계의 '보안 및 품질 설계(D09)'로 총 234회의 지적을 받아 전체 지적회수의 7.8%를 차지하고 있다. 다음으로 지적을 많이 받은 세부항목은 설계단계의 '설계자료(명세서)의 문서화 및 관리(D11)'로 총 230회의 지적을 받았다. 세 번째로 지적을 많이 받은 세부항목은 '데이터베이스 설계(D03)'로 총 228회의 지적을 받았다. 지적회수가 많은 상위 10개 항목 중에는 설계단계에서 6개 항목을 차지하고 있으며, 계획단계에서 2개 항목 그리고 분석단계와 구현단계에서 각각 1개 항목을 차지하고 있다. 전체 47개 세부항목 중 약 21%에 해당하는 상위 10개 항목이 차지하고 있는 지적회수는 총 1561회로 전체 지적회수 3027회의 약 52%를 차지하고 있다.

따라서, '공공소프트웨어 개발사업의 감리 결과에서도 지적사항 항목에 파레토 법칙(Pareto principle)이 적용될 것이다. 즉, 소프트웨어 개발단계 중 특정 부분(20%의 세부항목)에 80%의 지적회수가 집중될 것이다'라는 [가설 2]는 채택되지 않았다.

〈표 12〉 지적순위별 세부항목 현황

순 위	세 부 항 목	지 적 회 수			합 계	비 고
		단 일	중 간	최 종		
1	보안 및 품질 설계(D09)	71	81	82	234	
2	설계자료(명세서)의 문서화 및 관리(D11)	59	101	70	230	
3	데이터베이스 설계(D03)	61	84	83	228	
4	데이터 설계(D02)	55	108	56	219	
5	일정계획(P02)	36	64	53	153	
6	분석 결과의 문서화 및 관리(A7)	36	62	35	133	
7	기능 누락(C2)	24	13	56	93	
8	품질계획(P5)	26	41	25	92	
9	요구사항 분석내용의 설계반영(D1)	23	51	16	90	
10	처리절차 설계(상세설계)(D07)	25	46	18	89	
계		416 (49%)	651 (55%)	494 (49%)	1561 (52%)	
순위 11~47		433	522	511	1466	
합 계		849	1173	1005	3027	

주) ( )는 합계에 대한 %임.

이와 같은 결과는, R. S. Pressman이 소프트웨어 오류 발생에 적용한 파레토 법칙과는 일치하지 않았으며, 이런 현상은 개발된 소프트웨어에서 발생하는 오류 횟수만을 대상으로 한 Pressman과 공공소프트웨어 개발과정의 모든 단계에서 지적되는 지적회수를 대상으로 한 본 연구와의 자료의 차이에서 오는 결과라고 생각한다.

그러나 전체 세부항목의 21%에 해당하는 상위 10개 항목들이 전체 지적회수의 52%를 차지하고 있다는 것은, 이것만으로도 일부 항목에 지적사항이 심하게 편중되어 나타나고 있음을 보여주고 있다. 이와 같은 현상은 우리나라 공공소프트웨어 개발 성향을 나타내는 것이며, 개발과정에서 지적을 많이 받고 있는 일부 항목을 집중적으로 관리하여 오류를 줄여나간다면 공공소프트웨어의 품질을 효과적으로 관리할 수 있을 것으로 판단된다.

4.3.3 [가설 3]의 검증

소프트웨어 개발연도가 최근으로 올수록 사용자(end-user)의 편리성이 강조되는 정도를 알아보기 위해 관련 세부항목의 지적회수를 조사했다.

최근연도로 올수록 지적회수가 감소한다는 것은, 소프트웨어를 개발할 당시부터 사용자의 편리성을 철저하게 반영하고 있기 때문에 감리에서는 지적사항이 점차 적게 발생하는 것이라고 해석할 수 있다.

〈표 13〉 사용자 편리성 관련항목의 연도별 지적현황

세 부 사 항	합 계	사 업 연 도						
		1998	1999	2000	2001	2002	2003	비고
	168	6	14	25	46	46	31	감리회수
외부인터페이스설계(소프트웨어-사람)(D06)	57 (0.34)	1 (0.17)	5 (0.36)	10 (0.4)	20 (0.43)	13 (0.28)	8 (0.26)	
사용자편리성(C03)	71 (0.42)	5 (0.83)	7 (0.5)	12 (0.48)	22 (0.48)	15 (0.33)	10 (0.32)	t = -1.83*
요구사항검증시험(T03)	17 (0.1)	0	2 (0.14)	6 0.24)	5 (0.11)	1 (0.02)	3 (0.09)	
합 계	145 (0.86)	6 (1.0)	14 (1.0)	28 (1.2)	47 (1.0)	29 (0.6)	21 (0.7)	t = -1.79*

주) ***, **, *는 각각 1%, 5%, 10% 이하 수준에서 유의함.

<표 13>에서 보는 것과 같이 사용자 편리성과 직접 관련이 있는 세부항목으로 설계단계의 '외부 인터페이스 설계(소프트웨어-사람)', 구현단계의 '사용자 편리성', 시험단계의 '요구사항 검증시험'을 선정하여 조사했다. 외부인터페이스 설계는 전체 57회의 지적을 받았으며, 사용자 편리성은 71회, 요구사항 검증시험은 17회의 지적을 받았다. 세부항목의 연도별 지적회수를 당해 년의 사업수로 나누어 사업 당 평균 지적회수를 회귀분석 한 결과 '사용자 편리성(C03)'에서는 통계적으로 10%이하 수준에서 유의한 것으로 나타났으나, '외부인터페이스 설계(D06)'와 '요구사항 검증시험(T03)'에서는 통계적인 유의성을 발견할 수 없었다. 그러나 연도별 지적회수의 합계는 회귀분석 결과 통계적으로 10%이하의 수준에서 유의한 것으로 나타났다. 따라서 비록 통계적인 유의성이 강하지는 않으나 '공공소프트웨어 사업연도와 사용자 편리성 관련 세부

항목의 지적회수는 음(-)의 상관관계가 있을 것이다. 즉, 개발연도가 높아질수록 소프트웨어 개발에 사용자(end-user)의 편리성이 강하게 반영될 것이다'라는 [가설 3]은 채택되었다.

4.3.4 [가설 4]의 검증

정보시스템 감리가 공공소프트웨어 품질에 긍정적인 영향을 미치는가를 확인해 보기 위해 <표 14>과 같이 감리종류별로 지적사항 회수를 빈도분석한 후 비교해 보았다. 감리를 1회만(단일감리) 실시한 사업은 42개이며, 중간감리를 실시한 사업은 65개, 최종감리를 실시한 사업은 61개다.

감리를 1회만 실시한 42개 사업을 대상으로 조사한 결과 전체 849회의 지적을 받아 감리 당 평균 지적회수는 20.21회다. 중간감리를 실시한 65개 사업을 조사한 결과 전체 지적회수는 1173회로 감리 당 평균 18.05회의 지적을 받았으며, 최종감리의 전체 지적회수는 1005회로 감리 당 평균 16.48회의 지적을 받았다. 이를 통해 최종감리의 감리 당 평균 지적회수가 단일감리나 중간감리의 지적회수에 비해 현저히 감소하였음을 알 수 있다. 이에 따라 '단일감리나 중간감리의 지적회수보다 최종감리의 지적회수가 상대적으로 적게 나타날 것이다'라는 세부가설 ①은 채택되었다.

이와 같은 현상은 중간감리를 실시함으로서 공공소프트웨어 개발의 각 단계마다 사전에 문제점이 지적되어 개선되고 그 효과가 최종감리의 지적회수에 영향을 미친 것으로 분석된다. 즉, 간접적으로나마 정보시스템 감리가 공공소프트웨어 품질에 긍정적인 영향을 미치고 있음을 알 수 있다.

감리가 공공소프트웨어 품질에 어떤 영향을 미치는가를 알아보기 위해, 연도별로 전체 지적회수는 어떻게 변화되고 있는가를 조사했다. 감리가 매년 실시됨에 따라 공공소프트웨어 개발사업을 수행하는 업체에서 개발단계의 전 과정마다 사전에 철저한 준비를 한다면 감리 시 지적회수는 감소하게 될 것이다.

〈표 14〉 감리종류별 지적회수

( ) : 사업 당 평균 지적회수

구분 / 단계	전체	단일감리 (42건)	중간감리 (65건)	최종감리 (61건)
계획단계	683	188	291	204
분석단계	388	114	196	78
설계단계	1385	364	604	417
구현단계	373	115	53	205
시험단계	198	68	29	101
계	3027	849(20.21)	1173(18.05)	1005(16.48)

〈표 15〉 년도별 지적회수

구분 / 년도	감리회수	전체 지적회수	감리당 평균 지적회수	감리종류		
				단일감리 (42건)	중간감리 (65건)	최종감리 (61건)
1998	6	149	24.8	149	0	0
1999	14	293	20.9	216	39	38
2000	25	482	19.3	105	184	193
2001	46	887	19.3	379	291	217
2002	46	769	16.7	0	415	354
2003	31	447	14.4	0	244	203
계	168	3027	18.0	849	1173	1005

조사 결과 <표 15>에서 보는 것과 같이 168회 전체 감리회수의 감리 당 평균 지적회수는 18.0회다. 표를 통해 눈으로도 직접 확인할 수 있지만 연도별 감리 당 지적회수의 변화를 회귀분석 한 결과 연도가 높아질수록 지적회수의 감소가 5%이하 수준에서 유의하기 나타났다. 이에 따라 '감리가 실시될수록(최근연도일수록) 공공소프트웨어 개발사업의 전체 지적회수는 상대적으로 감소될 것이다'라는 세부가설 ②도 채택되었다.

결과적으로 세부가설 ①과 ②가 채택됨으로써 '정보시스템 감리가 공공소프트웨어 품질에 긍정적인 영향을 미칠 것이다'라는 [가설 4]는 채택되었다.

5. 결 론

5.1 연구결과 요약

본 연구에서는 1998년부터 2003년까지 6년 동안 H감리원과 Y감리원이 실시한 107개 공공소프트웨어 개발사업에 대한 168건의 감리보고서를 조사했다. 공공소프트웨어 개발 단계별로 발생한 지적사항을 살펴보면, 전체 지적회수 3027회 중 설계단계에서 1385(46%)회가 발생해 가장 큰 비중을 차지하였고, 다음으로 계획단계 683(23%)회, 분석단계 388(13%)회, 구현단계 373(12%)회, 시험단계 198(7%)회 순으로 발생했다.

공공소프트웨어 품질에 직접 관련된 세부항목들의 연도별 지적회수를 조사해 본 결과, 최근연도로 올수록 지적회수가 감소되고 있어 개발과정에서 품질문제가 점차 중요하게 고려되고 있음을 알 수 있었다.

지적사항이 많이 발생한 10개 항목을 대상으로 파레토 법칙(20 : 80 법칙)의 적용여부를 확인하였으나 파레토 법칙이 적용되지는 않았다. 그러나 전체 세부항목 47개 중 약 21%에 해당하는 10개 항목의 지적회수가 전체 지적회수의 51.6%를 차지하고 있어 오류가 일부항목에 편중되어 나타나고 있음을 알 수 있었다. 이를 통해, 개발과정에서 지적을 많이 받고 있는 일부 항목을 집중적으로 관리한다면 소프트웨어 품질을 효과적으로 관리 할 수 있을 것으로 판단되었다.

공공소프트웨어 개발사업에서 사용자(end-user)의 편리성이 연도별로 강조되는 정도를 알아보기 위해 관련 세부항목의 지적회수를 조사하였다. 그 결과 최근연도로 올수록 관련 세부항목의 지적회수가 상대적으로 감소함을 알 수 있어, 공공 소프트웨어 개발단계에서부터 사용자의 편리성이 점차 중요하게 고려되고 있음을 알 수 있었다.

감리가 공공소프트웨어의 품질에 총체적으로 어떤 영향을 미치는가를 알아보기 위해 단일감리, 중간감리, 최종감리에서 발생한 사업 당 지적회수를 분석해 보았다. 그 결과 최종감리의 지적회수가 단일감리나 중간감리보다 적게 나타나고 있어, 간접적으로나마 감리가 공공소프트웨어의 품질에 긍정적인 영향을 미치고 있음을 확인 할 수 있었다. 또한 연도별로 사업 당 평균 지적회수를 집계한 결과 최근으로 올수록 점차 지적회수가 감소하고 있음을 알 수 있었는데, 이러한 현상 역시 결국은 감리가 직·간접적으로 공공소프트웨어의 품질에 긍정적인 영향을 미치고 있는 결과라고 판단할 수 있었다.

5.1 연구의 제한점

국내에는 2000년 이후 20여 개의 전문 정보시

스템 감리업체가 매년 300여 건의 감리를 하고 있다. 그러나 업무 특성상 감리업체가 감리보고서의 공개를 꺼려하는 이유로 자료의 수집에 많은 어려움이 있었으며, 따라서 본 연구에서 사용된 168건의 감리보고서 만으로 국내 공공소프트웨어 감리 결과를 대표하고 또한, 공공소프트웨어의 개발 성향을 단정 짓기에는 표본의 수에 한계가 있다고 생각한다.

정보시스템 감리가 공공소프트웨어 품질에 미치는 영향을 판단하는데 있어서도 한계는 있다. 동일한 소프트웨어를 가지고 품질의 향상여부를 정량적으로 판정할 수 있는 지표가 아직은 마련돼 있지 않음은 물론, 동일 소프트웨어를 대상으로 감리를 받지 않은 경우와 감리를 받은 경우의 품질 상 차이를 계량적으로 비교할 수 있는 방법이 현실적으로 없다는 점이다. 따라서 본 연구에서는 특정한 공공소프트웨어를 대상으로 감리보고서에 나타나는 지적사항의 변화만으로 감리가 품질에 미치는 영향을 간접적으로 증명하고 있다는 것이 또 다른 제한점이라고 할 수 있다.

좀더 많은 감리보고서를 입수하여 자료의 다양성과 신뢰도를 높이고, 감리를 받은 소프트웨어와 감리를 받지 않은 소프트웨어의 품질을 정량적으로 비교할 수 있는 객관적인 방법과 그에 필요한 지표를 마련하는 일은 향후에도 계속 연구되어야 할 부분이라고 생각한다.

참 고 문 헌

[1] 김용경, "정보시스템 감리가 소프트웨어 품질에 미치는 영향", 「Journal of Information Technology Application and Management」, Vol. 9 No. 4, 2002년.

[2] 문대원, "공공부문 정보시스템 개발 프로젝트의 성공요인 도출을 위한 탐색 연구", 국민대학교 대학원, 박사학위논문, 2001년.

[3] 문대원, 장시영, 「정보시스템 감리」, 명경사, 1999년.

[4] 이상엽, "소프트웨어 프로세스 성숙도가 프로젝트 성과에 미치는 영향에 관한 연구", 한국외국어대학교 대학원 박사학위논문, 2000년.

[5] 정기원, 윤창섭 외 「소프트웨어 프로세스와 품질」, 홍능과학출판사, 1997년.

[6] 한국전산원, "감리결과 분석을 통한 주요 문제점 및 개선사례 연구", 2001년.

[7] Basili, V., "Empirical Software Engineering", *Software Process Newsletter*, No. 12, Spring 1998.

[8] Crosby, Philips B., *Quality is free, The art of making quality certain*, NALPENGUIN, 1980.

[9] Cusumano, M.A., *Japan's Software Factories*, Oxford University Press, Oxford, UK, 1991.

[10] Curtis, W., "Building a Cost-Benefit Case for Software Process Improvement", *Notes from Tutorial given at the Seventh Software Engineering Process Group Conference*, Boston, MA, May, 1995.

[11] Hartwick, J., and Barki, H., "Explaining the Role of User Participation in Information Systems Use", *Management Science*, 1994.

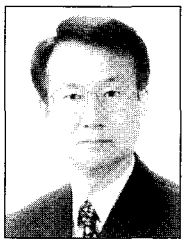
[12] Herbsleb J., Zubrow, D., and Siegal, J., "Software Process Improvement : State of the payoff", *American Programmer*, Vol. 7, No. 9, September, 1994.

[13] Humphery, Watts S., *Managing the Software Process*, Addison-Wesley, 1989.

[14] Pressman, R.S., *Software Engineering*, Mcgraw-Hill, 1997

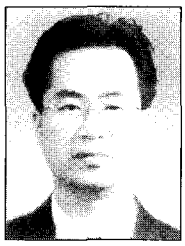
[15] Ravichandran, T., and Arun Rai, *Quality Management in System Development*, 1996.

저자소개



김 용 경

고려대학교를 졸업하고, 숭실대학교에서 이학석사 그리고 명지 대학교에서 경영학박사 학위를 취득했다. 현재 건양대학교 경영정보학과 교수이며, 한국정보기술응용학회 부회장으로 활동 중이다. 주요 관심분야는 소프트웨어공학, 소프트웨어 품질관리, 정보시스템 감리 및 감사 등이다.



김 병 기

성균관대학교를 졸업하고, 미오레곤주립대학교에서 경영학석사 그리고 콜로라도대학교에서 경영학박사 학위를 취득했다. 현재 건양대학교 경영학부 교수로 재직 중이며, 한국재무관리학회 및 한국전문경 영인학회 편집위원으로 활동 중이다. 주요 관심분야는 재무구조, 시장효율성, 생산성측정 등이다.

◈ 이 논문은 2004년 10월 11일 접수하여 1차 수정을 거쳐 2004년 12월 9일 게재확정되었습니다.