정보시스템 감리의 계량적 접근방법

신동익*

A Quantitative Approach to Information System Audit

요 약

최근의 정보 시스템 감리수요는 공공부문을 중심으로 매우 빠르게 증가하여 점차적으로 민간부문으로 확대되고 있는 추세이다. 이는 정보 시스템 감리를 통해 정보 시스템의 품질 향상을 기대할 수 있기 때문이며, 향후에도 조직의 정보시스템에 대한 의존도가 증가할 것이므로, 정보시스템의 효과성, 효율성 및 보안성은 더욱 중요한 문제로 대두 될 것이며, 따라서 시스템 감리의 중요성은 높아질 수 밖에 없을 것이다. 그러나 아직 정보시스템감리는 이러한 기대에 부응하기 위한 체계적 기술분야로 정립되지 못하고 있으며, 그 원인으로서는 분석 결과의 객관적 증거 확보를 통한 감리 결과의 신뢰성 제고가 미흡한 것이 제기되고 있다. 이는 정보시스템 감리가 다분히 주관적인 요소에 의해 수행되며 이로 인하여 감리인과 피감리인 간의 의견 상충이 다수 발생되며, 이러한 갈등은 감리의 효과성을 크게 저하시키고 있다. 본 연구 이러한 문제를 극복하기 위해 다양한 문헌 고찰을 통해 실질적인 계량적 감리 접근방법을 제시하고자 한다. 본 연구에서 제시된 방법론 실무에서 유용하게 쓰일 수 있으며, 이러한 노력은 우리나라의 감리 품질 제고에 큰 도움이 될 것이라 생각한다. 또한 이러한 계량적 데이터 추후 감리 연구의 기초 자료로 활용될 수 있어 감리 연구에도 많은 도움이 될 것이다.

Key words : 정보시스템 감리, 정보시스템 개발방법, 소프트웨어 품질

I. 서론

정보시스템 감리는 한국전산원을 중심으로 행정전산망 사업에 대한 회계중심 즉 선투자 후정산에 대한 감리로부터 시작되었으며, 정보화사업에 대한 개념과 추진방식의 변화와 정보화사업 주관기관의 요구로 인하여 기술적 부분에 대한 주관기관의 판

* 홍익대학교 교수 (dishin@hongik.ac.kr)

단을 도와주는 객관적 의견을 제시하는 형태로 바뀌게 되었다. 정보화사업의 지속적 증가와 이에 따른 감리에 대한 폭발적인 수요가 발생되었고, 따라서 감리서비스의 민영화가 이루어져서 현재는 10 여 개의 민영 감리법인과 한국전산원이 감리서비스를 제공하고 있다(한국전산원, 1999).

우리나라의 경우 감리는 정보화사업 특히 정보시스템 개발에 치중되어 수행되어 왔으며 이는 우리나라의 정보화 수준이 아직 시스템을 개발하여 보급하는 단계에 있기 때문으로 볼 수 있다. 따라서 개발 감리는 다른 형태의 감리 즉 운영이나 유지보수보다 현재 우리나라의 경우 그 중요도가 높다 하겠다. 따라서 개발 감리의 효율적이고 체계적인 접근방법에 대한 요구가 높아졌다.

우리나라의 경우 정보시스템 감리의 기본적인 체계는 전세계적으로 널리 알려져 있는 정보시스템 감사 단체인 ISACA(Information System Audit and Control Association)의 감사표준이나 실무보다는 ISO/IEC 12207 "소프트웨어 생명주기" 표준에 준하고 있다. 이렇게 다르게 된 이유는 ISACA 는 주로 운영 시스템에 대한 감사이며 그 목적이 정보의 무결성에 집중되어 있다. 물론 최근에는 COBIT(Control Objective for Information Technology)를 발표하면서 정보시스템의 품질, 보안, 회계적 요건 등을 포함하는 광범위한 체계를 수립하였으나, 그 기술적 수준과 구체성이 실제로 개발 사업에 대한 감리를 수행하기에는 부족한 것으로 보인다.

반면에 ISO/IEC 12207 은 양자간 소프트웨어 개발 계약에 있어서 수행되어야 할 활동을 기술함으로써 계약범위를 명확히 하고 또한 이러한 활동이 적절히 수행되었는지를 확인함으로써 계약에 대한 충실도를 알 수 있게 된다. 현실적으로 우리나라의 대부분의 정보화사업에서 요구되는 점은 이러한 기술적 측면에서의 계약에 따른 적절한 활동과 결과물에 대한 확인이며, 제 3 자의 이러한 확인을 통하여 정보화사업 주관기관은 사업을 원활하게 추진할 수 있게 된다. 이런 측면에서 볼 때 감리체계를 ISO/IEC 12207 에 따라 수립하는 것이 타당한 것으로 보이며, 또한 현재 정보통신부에서 고시한 감리기준도 위 표준에 준거하여 작성되었다.

우리나라의 정보시스템감리는 위에서 설명한 것처럼 개념적으로는 ISO/IEC 12207 을 준하고 있으나 명확히 이 표준을 준수한다고는 천명하지 못하고 있다. 특정 국제표준에 따른 국내 체계를 수립하는 것이 국제적인 신뢰도를 갖기 위해서는 매우 중요할 수 있으나, 국내 환경을 반영하고 또한 기타 표준 및 실무 모범사례를 고려하여 독자적인 체계를 구축하는 것도 바람직하다. 그러나 우리나라의 경우 소프트웨어 공학 수준이 외국 선진국에 비해 크게 뒤떨어져 있고, 따라서 선진 수준을 정확히 이해하여 사용해 보고 추후에 발전을 시키는 방법이 더욱 효과적일 수 있다.

본 연구는 정보시스템감리를 ISO/IEC 12207 에서 정의한 범위 내에서 해석하고, 이를 기초로 감리의 체계를 제시한다. 우선 본 절에서는 정보시스템감리를 정의하고 이에 비추어볼 때 현행 감리 실무의 문제점을 도출한다.

II. 정보시스템 감리의 정의

감리를 ISO/IEC 12207 에서는 "수행되

는 소프트웨어 프로세스와 프로덕트의 요구사항에의 준수성을 인정된 사람이 독립적으로 심사하는 것" 으로 정의하고 있다. 여기서 요구사항이란 양자간 계약 요구사항으로부터 모든 종류의 관리적 및 기술적 요구사항을 모두 포함한다.

여기서 프로세스란 입력물을 출력물 또는 산출물로 변환하는 일련의 상호 관련된 활동을 말한다. 출력물은 궁극적인 소프트웨어가 되는 원시코드 및 실행파일과 같은 프로덕트와 이러한 프로덕트를 개발하는데 필요한 데이터로 구분된다. 이중 데이터는 흔히 문서의 형태로 작성된다. 여기서 문서는 단순히 전통적 의미의 종이에 작성된 문서 뿐만이 아니라 전자적 형태로 데이터를 보관하는 전자 문서도 포함한다. ISO/IEC 12207 에서 감리의 정의는 프로덕트만 포함함으로써 문서에 대한 검증을 배제하고 있다. 이러한 정의는 감리를 주로 시험단계에서 수행되는 것으로 인식하기 때문인 것으로 보인다.

그러나 우리나라의 경우 감리의 사회적 기능이 단순히 개발이 거의 완료된 시점에서 확인을 해주는 기능에 국한되는 것이 아니고, 사전에 즉 계획 및 분석/설계 단계에서 충분한 검증을 통하여 부실 위험을 줄이고 품질을 향상하자는 것에 더욱 중요성을 두고 있으므로, 정보시스템감리는 품질보증/검증/확인/공동검토 등의 프로세스를 포함하는 것으로 보는 것이 타당하다. 따라서 정보시스템감리의 정의를 개발 프로세스 및 문서와 프로덕트의 요구사항에의 준수성을 심사하는 것으로 보는 것이 타당할 것으로 생각된다.

III. 정보시스템 감리의 실무 현황 및 문제점

현재 정보시스템감리는 한국전산원을 비롯한 10 여 개의 민간 감리법인들이 주로 수행하고 있으며, 각 기관 내부에서 내부감사 차원에서 감리를 수행하는 경우도 있다. 정보시스템감리의 문제점을 총체적으로 파악하여 근원적인 대책을 수립하는 것이 필요할 것이나, 본 연구는 감리 수행실무에만 국한되었으므로 이 범위 내에서 현황을 분석하고자 한다.

현재 감리는 일반적으로 감리 RFP, 감리계획서 작성, 감리업체 선정, 감리 계약, 감리수행, 사후검토 등과 같은 순서로 진행된다. 여기서 감리의 가장 중요한 활동은 감리수행으로 볼 수 있으며, 적절한 실무방법에 따른 감리수행만이 감리의 효과를 보여줄 수 있을 것이다. 현재 감리 수행은 감리대상업무 특성에 따라 다를 수 있으나, 일반적으로 다음과 같은 정보기술 요소별로 감리인에게 업무를 할당하고 추후 공동 회의를 거쳐 보고서를 통합하는 방법을 택하고 있다(한국전산원, 1997, 1998, 1997; 문대원과 장시영, 1998).

- 응용 소프트웨어
- 데이터베이스
- 네트워크
- 시스템 아키텍처
- 보안 등

위와 같은 실무방법이 자리를 잡게 된 것은 기본적으로 각 정보기술 요소별로 전문성이 다르고, 감리에 필요한 최소한의 전문성 없이는 감리 수행이 어렵다는 점이다. 또 다른 요인으로는 위와 같이 정보기술 요소별로 구분은 정보기술 아키텍처를 파악하

기 쉽고 따라서 전반적인 시스템의 문제점을 쉽게 찾아낼 수 있다는 점이다. 물론 현재로서는 정보기술 아키텍처를 이용한 감리가 수행되고 있지는 못하지만 현재의 실무 방법으로 충분히 정보기술 아키텍처를 이용한 감리로 발전될 가능성이 있다는 점이다 (한국전산원, 1999).

현재의 정보시스템 감리는 나름대로 많은 공헌을 한 것이 사실이나 아직도 많은 문제점을 갖고 있다. 특히 다음과 같은 문제점들이 주요한 문제로 제기되고 있다.

- 감리 방법론적 지식의 공유체계 미흡

공공부문 정보시스템 감리는 1987 년도 1 차 행정전산망 구축사업이후 10 여년간 시행되어 왔으나 감리인에 의한 시행경험이 체계적으로 축적되지 못하고 감리품질에 있어 크게 향상되지 못하고 있다. 감리결과에 대한 감리 이력이 관리되지 않아 동일한 감리대상에 대하여 일관성 있는 의견이 제시되지 못하는 사례가 있다.

- 많은 개발 산출물 분석 노력으로 감리의 효율성 저해

정보시스템 감리의 대상인 분석, 설계, 구현단계 산출물과 시스템에 대한 검토 및 확인 작업에 있어 대부분 수작업으로 대조, 추적함으로써 대형 프로젝트 감리시 감리 생산성을 저하시키고 감리 지적사항이 단편적인 경향이 있다.

- 분석 결과의 객관적 증거 확보를 통한 감리 결과의 신뢰성 제고 미흡

감리 지적사항에 대한 근거(fact finding)에 있어 감리인의 직관적 판단과 제한적인 대상 검토로 인하여 객관성과 충분성(Audit coverage)이 부족하여 감리의 신뢰성 확보에 한계가 있다.

위의 문제점 중에서도 가장 근본적인 문제는 객관적 감리 증거의 수집 및 평가 문제이다. 실제로 다른 두 문제는 이 문제의 해결에 많이 의존하고 있다. 감리 지식의 공유와 의견의 일관성 문제는 감리 체계를 서로 다르게 갖고 있기 때문이며 이는 객관적 감리 증거를 수집하게 하는 기본틀이 있으며 상당부분 해소 가능하다. 의견의 일관성 역시 객관적 증거와 평가 방법을 적용하면 의견의 일관성 더욱 향상될 것이다.

또한 개발 산출물을 분석하는 노력이 많이 필요해 흔히 감리를 노동 집약적인 업무로 이해되기 쉽다. 실제 많은 실무 감리인은 감리를 매우 하급의 직종으로 폄하하는 경우를 볼 수 있다. 이는 분석 방법이 매우 직관적이고 경험적이라는 것이다. 물론 감리 업무의 특성상 이러한 능력이 매우 필요하기는 하는 좀더 체계화된 증거의 수집, 평가 방법은 분석 노력을 좀더 완화할 수 있을 것이다.

이러한 측면에서 볼 때 감리 증거를 수집하고 평가하는 프로세스야 말로 감리에서 핵심적인 활동으로 볼 수 있다. 그럼에도 불구하고 정보시스템 감리 분야에서는 이러한 문제에 대한 연구가 매우 미흡하였다. 본 연구는 감리 증거와 평가에 관련된 다양한 기법을 분석하고 이를 기초로 객관적인 감리 접근방법을 제안한다.

IV. 감리 증거 수집 및 평가 기법

현재의 실무는 매우 비정형화된 형태로 수행되고 있으므로 접근방법의 수립에 앞서

감리수행 프로세스의 정형화가 필요하다. 따라서 본 연구에서는 공신력 있는 국제적 문서와 실무관행에 기초하여 감리 프로세스를 정의하고 모델링 우선 수행한다.

일반적으로 감리는 검토 또는 검증/확인 방법을 활용하여 수행한다. 즉 ISO/IEC 12207 에서 정하고 있는 기타 프로세스 즉 품질보증이나 감리 프로세스는 위 두 프로세스를 기본 단위로 사용하는 상위 프로세스로 볼 수 있다는 점이다. 따라서 감리 프로세스를 이해하기 위해서는 검토, 검증/확인 프로세스에 대한 이해가 필요하다. 이들 프로세스에 대한 이해를 구하기 위해 국제적으로 인정된 다음에 제시하는 IEEE 표준을 기초로 분석을 한다(IEEE, 1998).

- 소프트웨어 검토(IEEE 1028-1997)
- 소프트웨어 검증 및 확인(IEEE 1012-1998)

IV.1 소프트웨어 검토(Software review)

소프트웨어 검토는 일반적으로 "소프트웨어 제품이 의견 또는 승인을 위하여 프로젝트 관리자나 관련 인력, 고객, 고객의 대리인 또는 관련 이해관계자에게 제출하는 프로세스 또는 회의"로 정의된다(IEEE 1028, 1997) . 여기서 소프트웨어 제품은 컴퓨터 프로그램, 절차 및 관련된 문서와 자료의 총체를 말한다.

흔히 소프트웨어 검토의 필요성으로는 검토는 일정 목표를 만족하는데 도움을 준다든지, 검토는 작업량을 줄인다든지 하는 것이 강조된다. 일반적으로 총 개발노력의 44%가 재작업이며, 개발단계별로 구분하면 요구사항 분석 단계에서 1%, 설계 단계에서 12%, 코딩 단계에서 12%, 시험단계에서 19%의 재작업이 이루어지는 것으로 보고되고 있다.

소프트웨어 검토는 사전에 수행되는 시험으로 간주될 수 있으며 시험으로 발견하기 어려운 오류를 발견하여 교정하는 것이다. 흔히 오류는 발생시점에서부터 시간이 지날수록 교정에 더욱 많은 비용이 필요로 한다. 따라서 오류는 가능한 빠른 시간 내에 식별하여 교정하는 것이 필요하다. 이런 측면에서 검토는 매우 필요한 작업이다.

소프트웨어 검토는 또한 개발자에게 오류 정보를 제공, 동료에게는 작업 제품과 개발과정에 대한 정보제공, 시험자에게는 오류 가능성 자료를 제공, 관리자에게는 개발현황에 대한 정보를 제공하며, 소프트웨어 프로세스 향상 그룹에게 프로세스 현황을 제공한다. 따라서 소프트웨어 검토는 소프트웨어 개발에서 품질 향상에 상당한 공헌을 하게 된다.

소프트웨어 검토는 잘 정의된 프로세스를 따라 진행된다. 참가자 역시 잘 정의된 역할을 맡아 수행하게 된다. 역할로서는 중재자, 검토자, 기록자, 개발자 등이 있다. 검토는 오류 제거, 요구사항 도출 등과 같이 잘 정의된 목표에 만족하기 위해 일관성 있는 자료 수집을 통해 측정치를 제공한다.

이러한 소프트웨어 검토는 상당한 품질 향상 효과가 있는 것으로 알려져 있다 (Dobbins and Donnelly, 1998; Fitzgerald and O'Kane, 1999). 검토는 모든 오류의 60~100%를 식별할 수 있으며, 이러한 오류를 사전에 제거함으로써 전체 프로젝트 비용을 감소한다. 비용 감소는 사후적 오류제거 비용이 사전적 오류제공 비용보다 약 10~100 배

<표 1 검토의 종류>

방법	목적	속성
Walkthroughs	최소한의 노력 개발자 훈련 필요 적은 검토시간 소요	준비가 거의 필요없음 비정형적인 프로세스 계량적 측정 없음 정향적 검토방법이 아님
Technical reviews	요구사항 도출 모호성 해결 훈련필요	정형적 프로세스 개발자 발표 다양한 범위의 토의 가능
Inspections	모든 오류의 효율적이고 효과적인 식별과 제가	정형적 프로세스 점검표 활용 측정치 활용 검증 단계 필요

비싼데 기인한다. 그러나 검토 활동역시 약 ~5%까지의 비용을 필요로 한다.

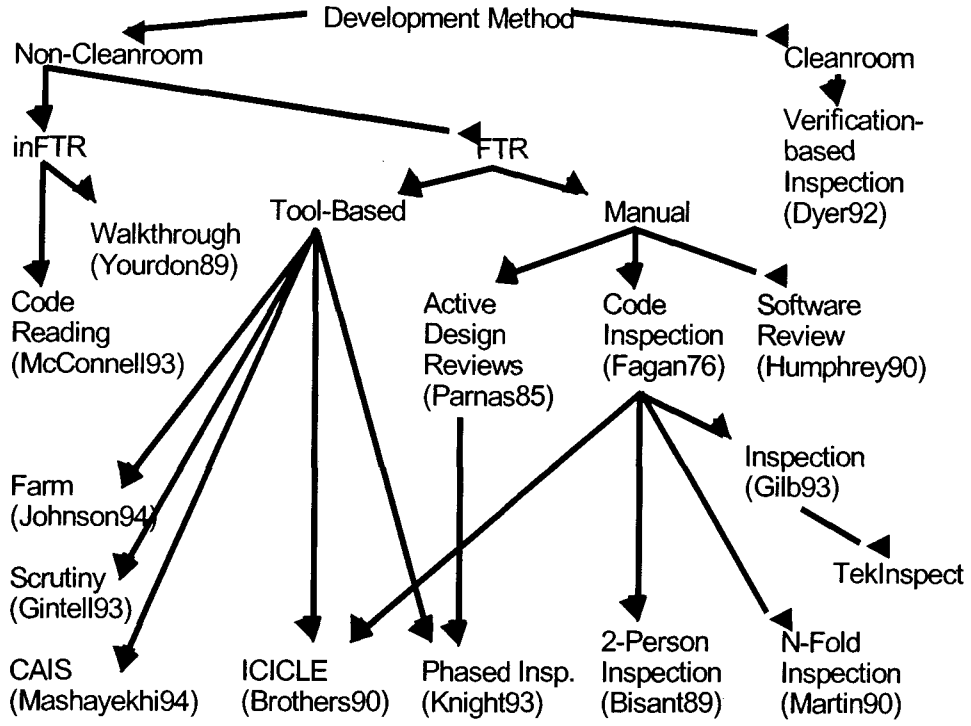
소프트웨어 검토에 대한 실제 사례는 많이 보고되고 있다. Aetna 보험회사의 경우 검토를 통해 82%의 오류를 발견했고, 25%의 비용절감 효과를 얻었다. Bell-Northern Research 는 하나의 오류에 대해서 검토비용은 1 시간, 시험비용: 2~4 시간, 배포후 비용은 33 시간인 것으로 보고하고 있다. Hewlett-Packard 는 검토로 인한 비용절감이 1993 년에 $21,454,000 에 달한다고 한다.

소프트웨어 검토에는 다양한 종류가 있다(<표 1>). 가장 많이 거론되는 것으로는 기술검토, 검사, 워크스루를 들 수 있다. 기술검토(technical review)는 자격 있는 인력이 수행하는 소프트웨어 제품의 의도된 사용에의 적합성 검사와 표준이나 규격에의 차이를 식별하는 체계적 평가 (대안 제시 및 평가를 포함할 수 있음)를 말한다.

검사(inspections)는 소프트웨어 제품의 표준 및 규격에의 차이 및 오류를 포함하는 예외사항을 식별하는 검사로서, 독립적 진행자가 주도하고 내부동료가 검사하여 예외사항을 찾으나 해결책의 제시는 하지 않는 것이 보통이다. 마지막으로 워크스루(walkthrough)는 설계자나 프로그래머가 개발팀 및 이해관계자와 함께 소프트웨어 제품을 비공식적 방법으로 검토하면서 가능한 오류나 문제점을 토의하는 것으로 가장 비정형적인 방법이다.

감리는 소프트웨어 제품 및 프로세스의 규격, 표준, 계약사항 및 기타 기준에의 준수성을 독립적을 심사하는 것으로 검토와 유사하게 볼 수도 있다. 그러나 검토는 실무자가 주도하여 기술적 문제에 대해서 검사하는 것으므로 감리는 검토 산출물을 활용하여 감리를 수행하는 것이 기본이다. 그러나 수행되지 않았을 경우 현실적으로는 대행하는 역할도 수행할 수 있을 것이다.

현재 실제적으로 SI 업체가 검토 프로세스를 수행하는 경우가 거의 없으므로 Non-cleanroom 환경에서 시스템이 개발된다고 볼 수 있으며, 이러한 환경에서 감리는 정형적인 기술검토보다는 비정형적인 기술검토, 그 중에서도 walkthrough 의 형태를 취하는 것으로 판단된다(<그림 2> 참조). 그러나 이러한 방식으로는 감리의 객관화를 도모하기 어렵다. 실제로 검토는 실무자가 자체의 품질향상을 수행하는 것인데 감리가 이러한 측면만 강조하는 것은 문제가 된다.



왜냐하면 감리는 발주자와 수주자의 중간에서 중립적 입장에서 객관적 평가도 수행해야 하기 때문이다.

현재 수준에서의 감리는 소프트웨어 검토가 매우 중요한 역할을 차지하고 있다. 감리에서 소프트웨어 검토의 정형화를 위해서는 IEEE 1028 을 약간 수정하여 감리 환경에 맞게 검토절차를 적용하는 것이 필요할 것이다. 검토는 기본적으로 오류를 조기 발견하여 수정하고자 하는 노력이며, 이러한 노력에서 더욱 발전하여 소프트웨어 제품과 프로세스를 객관적으로 심사하기 위해서는 검증 및 확인 방법의 활용이 필요할 것으로 생각된다.

V. 소프트웨어 검증(verification) 및 확인(validation)

소프트웨어 검증 및 확인은 소프트웨어 생명주기 상에서 소프트웨어 제품 및 프로세스의 객관적 심사를 제공하며, 소프트웨어 및 시스템 요구사항이 완전하고, 정확하고, 일관적이며, 시험가능한지 여부를 심사한다.

확인 프로세스는 소프트웨어가 할당된 요구사항을 만족하는지 및 올바른 문제를 풀고있는지 여부에 대한 증거를 제공하며, 검증 프로세스는 다음 항목에 대한 증거를 제공한다.

- 요구사항에의 적합성(e.g., correctness, completeness, consistency, accuracy)
- 표준 및 실무관행의 만족
- 생명주기 활동의 완료 여부 및 다른 활동의 시작 여부 결정을 위한 기초 수립

<표 2 개발 활동별 V&V 작업>

활동	V&V 작업
시스템 요구사항	(1) 시스템요구사항 문서 평가 (2) 중요도 분석 (3) 하드웨어/소프트웨어/사용자인터페이스 요구사항 할당 분석 (4) 추적성 분석 (5) 안전성 분석 (6) 위험 분석
소프트웨어 요구사항	(1) 추적성 분석 (2) 소프트웨어 요구사항 평가 (3) 인터페이스 분석 (4) 중요도 분석 (5) 시스템 시험계획 검증 (6) 인수 시험계획 검증 (7) 형상관리 심사 (8) 안전성 분석 (9) 위험 분석
소프트웨어 설계	(1) 추적성 분석 (2) 소프트웨어 설계 평가 (3) 인터페이스 분석 (4) 중요도 분석 (5) 컴포넌트 시험 계획 검증 (6) 통합 시험계획 검증 (7) 시험 설계 검증 (8) 안전성 분석 (9) 위험 분석
구현	(1) 추적성 분석 (2) 원시코드 및 문서 평가 (3) 인터페이스 분석 (4) 중요도 분석 (5) 시험사례 검증 (6) 시험절차 검증 (7) 컴포넌트 시험결과 검증 (8) 안전성 분석 (9) 위험 분석
시험	(1) 추적성 분석 (2) 인수시험 절차 검증 (3) 통합시험 결과 검증 (4) 시스템시험 결과 검증 (5) 인수시험 결과 검증 (6) 안전성 분석 (7) 위험 분석

감리는 검토의 일종으로 볼 수 있으나, 감리의 역할이 사용기관의 입장을 대신하여 개발 소프트웨어의 성과를 평가하는 것으로 까지 확장될 경우, 감리는 검증/확인 프로세스를 사용하여 객관적으로 소프트웨어를 심사해야 한다. 검증/확인 프로세스는 개발,

운영, 유지보수 각각에 대해 다른 활동이 고려될 수 있으므로, 본 연구에서는 개발에 한하여 연구를 수행한다.

개발 프로세스는 ISO12207 에 따르면 시스템 요구사항, 소프트웨어 요구사항, 소프트웨어 설계, 구현, 시험, 설치로 크게 구분되며, 각 단계별로 다양한 V&V 활동이 필요하게 된다. 이러한 검증 및 확인 활동들은 검토보다는 훨씬 더 객관적인 입장에서 심사하는 것이다. 그러나 이러한 활동이 의미기 있기 위해서는 이러한 활동을 통해 수집된 증거의 객관적 평가가 이루어져야 한다.

감리 실무에서의 증거평가 방법

현재 실무에서 사용되는 증거평가 방법은 공식적인 방법이 없으며 감리인 주관에 의해서 결정된다. 그러나 감리인간의 주관적 편협성이 치우침을 막고자 흔히 다음과 같은 규칙을 적용한다

- 감리가 시행되는 단계에서 나타난 문제점이 다음 단계의 시작이나 혹은 다음 단계에서 필요로 하는 시점까지 완료되기 어렵고, 또한 그 문제가 시스템에 막대한 영향을 미칠 경우 흔히 부적정으로 평가하게 됨
- 감리가 시행되는 단계에서 나타난 문제점이 다음 단계의 시작이나 혹은 다음 단계에서 필요로 하는 시점까지 완료되기 어려우나 그 문제가 시스템에 최소한의 영향을 미칠 경우 흔히 미흡으로 평가하게 됨
- 감리가 시행되는 단계에서 나타난 문제점이 다음 단계의 시작이나 혹은 다음 단계에서 필요로 하는 시점까지 완료될 수 있는 경우 적정으로 평가하게 됨

위와 같은 규칙은 시점과 중요도를 고려한 의사결정 방법으로 매우 효과적이라 할 수 있다. 그러나 이러한 평가 방법은 증거의 전체적 측면을 보고 평가하지 못하고 소수의 눈앞에 쉽게 보이는 특징 또는 문제만으로 평가를 한다는 점에 증거를 체계적으로 평가 의견에 반영을 하지 못한다는 점이 결점이다.

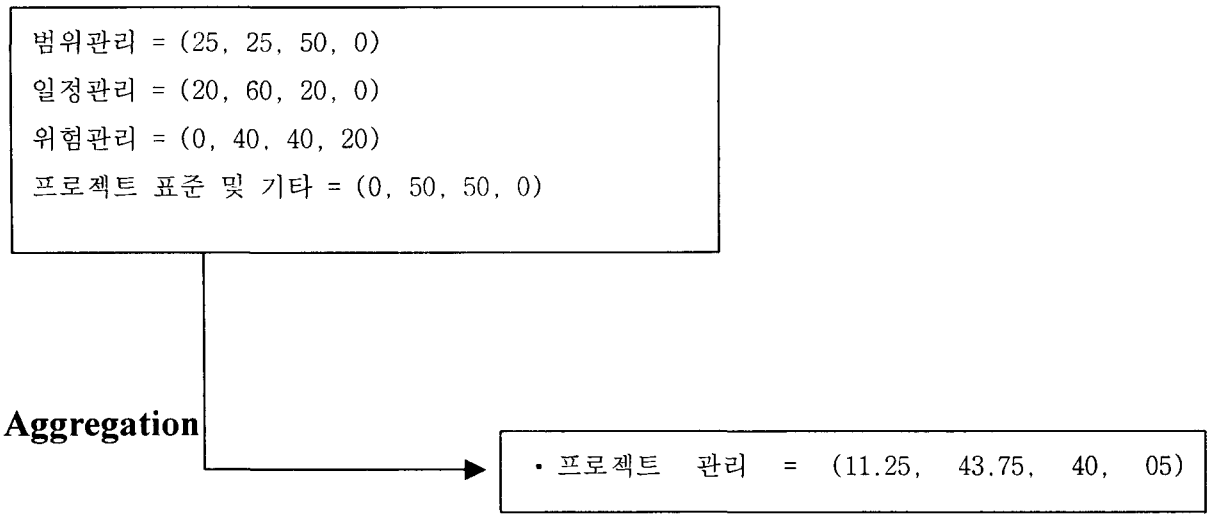
벡터 형식의 평가 기법

실무 감리 평가 체계가 가지는 단점인 주관적이며 정성적인 문제를 보완할 수 있는 방법으로 벡터 형식의 평가 기법이 제안되었다. 이러한 평가기법은 평가의 근거를 수치 즉 비율로 표시할 수 있는 장점이 있다.

- 벡터 형식의 평가절차
  (1) 기준항목들에 대해 평가등급을 책정한다..
  (2) 각 서브분야 (예로 프로젝트 관리 중 범위관리)에 대해 평가한 내용을 백분율로 표시한다..
  (3) 이러한 과정을 각 서브분야별로 반복하여 나온 값을 최종적으로 합산한다.

- 예시

본 예시는 현재 일반적으로 사용하고 있는 것과 비슷한 적절, 부분적절, 미흡에다



<그림 1 벡터 평가 예시>

미수행을 추가하여 4 단계의 평가등급을 사용해 보았다. 본 보고서의 예시에 따르면 범위관리는 총 4 항목 중 하나가 적절, 한 항목은 부분 적절, 2 항목은 미흡으로 평가되어 (25, 25, 50, 0)으로 표기되고 이는 적절이 25%, 부분적절이 25%, 미흡이 50%로 해석된다. 예를 들어 프로젝트 관리는 범위관리, 일정관리, 위험관리 및 프로젝트 표준 및 기타로 이루어 졌다고 가정하면 각각의 평가가 (25, 25, 50, 0), (20, 60, 20, 0), (0, 40, 40, 20), (0, 50, 50, 0)으로 표기된다. 이를 합하면 (11.25, 43.75, 40, 5)가 되어 전체 프로젝트 관리 분야는 약 11.25%가 적절하고 43.75%가 부분적절, 40%는 미흡이며 5%는 전혀 시행되지 않았음을 알 수 있다.

위에서 설명한 벡터 평가 방법은 감리항목별로 적절성의 수준을 계량화 하여 이를 벡터 방식으로 표현하고, 종합함으로써 전체 의견을 계량적으로 표명할 수 있도록 한 방식으로 볼 수 있다. 이 방식이 기존의 방식보다는 객관적인 평가가 가능하다. 그러나 이 방식도 감리항목이 사전에 표준화된 것이 있어 동일한 항목이 여러 감리에 적용가능 하다는 것을 가정하고 있고, 또한 감리항목의 개수가 동일하지 않은 경우 종합하기 어렵다는 점이 있다. 실제 감리의 경우 감리항목이 감리 목적이나 대상에 따라 달라 3>)지게 되므로 표준화된 감리항목을 수립하는 것은 사실상 불가능하다. 따라서 이 방법은 계량적 평가결과를 제공하고자 하는 좋은 의도로 시작되었으나 현실적으로 사용하기에 매우 어려울 것으로 보인다

GQM 을 이용한 평가방법

GQM(Goal-Question-Metric) 방법은 목표지향적 프로세스를 사용하는 평가방법으로, 측정하기 어려운 비정형적인 목표로부터 시작하여 목표를 만족하기 위해 필요한 활동을 문제로 구성하고, 각 문제들의 충족수준을 척도로 측정하는 방법이다. 이러한 GQM 개념을 좀더 확장하여 목표를 하위목표로 계속적으로 분할하여 최종적으로는 측정 가능한 수준으로까지 전개하는 것을 목표중심적 프로세스 방법이라고 한다. 이러한 전개

에서 가장 중요한 것은 하위목표와 상위목표 간의 추적성을 유지하는 것이다.

이러한 GQM 방법은 감리에서 수집된 증거를 통합하는 유용한 수단으로 활용될 수 있다. 시스템 요구사항 단계를 예를 들어 GQM의 적용가능성을 분석하기로 한다.

- 시스템요구사항 단계의 평가(그림 3)

  - 예를 들어 A, B, C의 세가지 하위시스템을 가진 시스템에 대하여 감리를 수행할 경우, 시스템 A에 대한 증거평가는 <표 4>를 이용하여 수행가능함
  - <표 4>에서 가상적으로 각각의 척도에 대한 한계수준을 80%로 동일하게 가정한다면 정확성과 응용 요구사항의 완전성은 부적합한 것으로 평가될 것임
  - 또한 시스템 A의 "요구사항 할당 분석" 목표는 부적합한 것으로 판정을 내릴 수 있음. 왜냐하면 부적합한 것이 하나라도 있으면 목표를 도달하지 못한 것으로 볼 수 있기 때문임
  - 동일한 방법으로 다른 목표들에 대한 적부합을 평가할 수 있으며, 이들 목표 중에서 하나라도 부적합이 있을 경우 시스템 A는 부적합한 것으로 결론을 내릴 수 있음

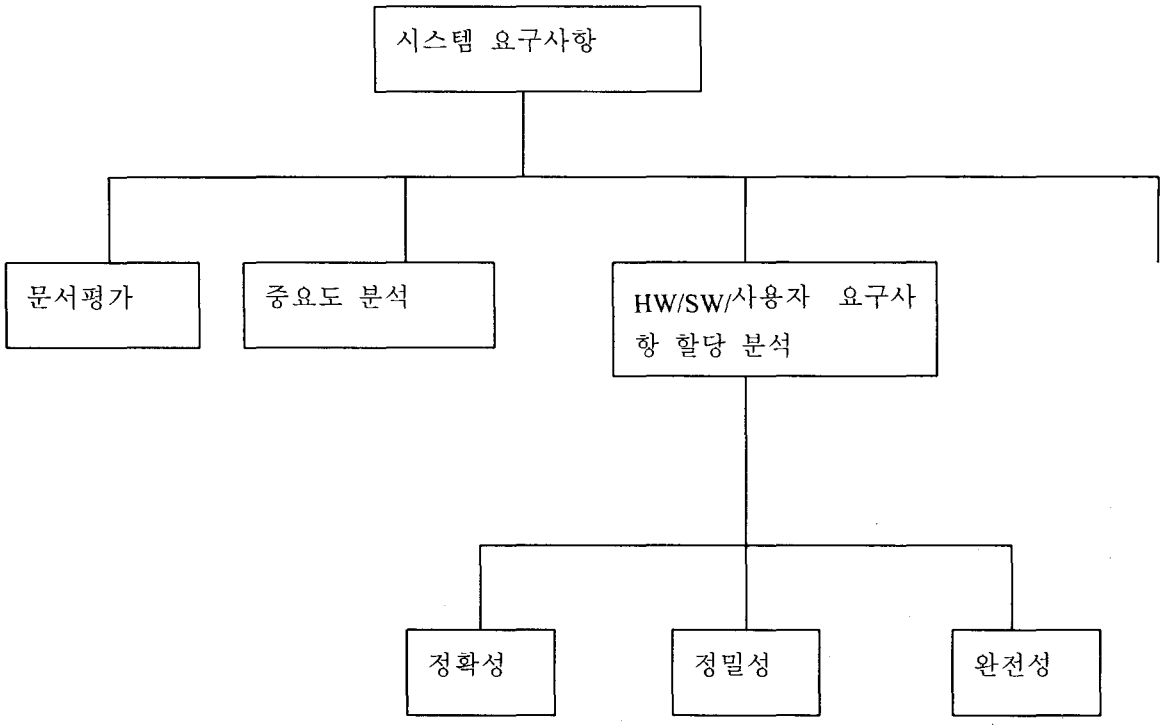
GQM 평가방법은 기본적으로 V&V 활동을 기본으로 평가를 시도하며, V&V 활동 목표들은 다시 측정 가능한 문제들로 세분화되고 측정된 값을 기초로 평가를 한다. 각각의 척도에 대해서 한계수준 또는 간격을 정하여 적합, 부적합 또는 적합, 미흡, 부적합 등으로 임의 평가수준을 정하여 평가를 수행하며, 하위 목표에서 최하위 평가등급을 상위 목표의 평가등급으로 정함으로써 보수적인 평가 결정을 유도한다.

VI. 결론

본 연구는 감리의 객관화를 위한 시도이다. 감리가 실무적으로는 매우 중요한 역할을 담당하고 있으나 학술적으로는 연구가 매우 미흡한 상태이다. 본 연구에서는 소프트웨어 공학에서 사용되는 GQM의 방법을 활용하여 소프트웨어 개발 단계별로 평가하는 방법을 제안하였다.

본 연구에서 제안한 방법은 일반적으로 감리에서 수행하는 검토 형태가 아닌 검증 및 확인 방법에 기초한 것으로 좀더 객관적 입장에서 평가가 가능할 것이다. 또한 평가의 결과가 계량적이므로 감리인이 적정, 미흡, 부적정의 판단을 할 때 매우 주관적이며 임의적인 결정을 좀더 정형화된 형태로 의사결정할 수 있도록 도와줄 것이다.

향후 감리 연구는 좀더 소프트웨어 프로젝트 관리 차원으로 다가가야 할 것이다. 따라서 소프트웨어 측면과 프로젝트 관리 측면이 통합되어서 연구되어야 할 것으로 생각된다(Keil, 1995; Nidumolu, 1995; Nord, 1997). 또한 요구사항에의 만족여부만을 따지는 수준에서 좀더 상위 수준의 영향 측면도 고려되어야 할 것이다(Kaplan and Norton, 1992). 또한 현재의 감리는 개발에 많이 치우쳐져 있으나 운영과 통제 역시 하나의 체계 속에서 통합되는 것이 바람직 할 것이다(Weber, 1999).



<그림 3 시스템요구사항 단계에서의 V&V 활동>

<표3 증거평가 표>

목표 (요구사항 할당 분석)	문제	척도	값	판정
정확성	- 할당된 성능요구사항의 사용자 요구 만족	- 만족된 성능 요구사항의 비율	70%	부적합
정밀성	- 내외부 인터페이스의 사용자 요구 만족	- 만족된 내외부 인터페이스의 비율	95%	적합
완전성	- 응용 요구사항의 사용자 만족 - 유지보수 요구사항의 완전한 명시 - 전환 방법의 사용자 요구사항 만족	- 만족된 응용 요구사항의 비율 - 만족된 유지보수 요구사항 비율 - 만족된 전환 요구사항의 비율	80% 90% 95%	부적합 적합 적합

참고문헌

문대원 · 장시영. *정보시스템감리 사업관리, 시스템개발 및 감리실무*, 명경사, 1998.

한국전산원, *감리증적 확보방안 및 평가방법 연구*, 2000.

한국전산원, *국가 정보화 촉진을 위한 품질정책 연구*, 1997.

한국전산원, *소프트웨어 프로세스 평가지침(안)*, 1996.

한국전산원, *시스템 개발방법론 적용기준에 관한 연구*, 1997.

한국전산원, *시스템 시험 감리지침 연구*, 1997.

한국전산원, *정보기술 아키텍처 수립과 표준 적용을 위한 연구*, 1999.

한국전산원, *정보시스템 감리 효과성 측정에 관한 연구*, 1998.

한국전산원, *정보시스템 감리의 발전 방향*, 1999.

한국전산원, *정보시스템 객체지향 개발 감리 지침 연구*, 1998.

한국전산원, *정보시스템 보안/통제 감리 지침*, 1998.

한국전산원, *정보시스템 분석 감리지침 연구*, 1997.

한국전산원, *정보시스템 운영 감리 지침*, 1998.

한국전산원, *정보시스템 유지보수 감리 지침 연구*, 1998.

한국전산원, *정보시스템 품질관리 감리 지침 연구*, 1998.

한국전산원, *정보시스템 형상관리 감리지침 연구*, 1997.

한국전산원, *정보화 성과 관리*, 1997.

Dobbins, J. H. and Donnelly, R. G., "Summary Research Report on Critical Success Factors in Federal Government Program Management,"Acquisition Review Quarterly, Winter 1998, pp. 61-82.

Field, T., "When BAD Things Happen to GOOD Projects," CIO, 15, Oct. 1997, pp. 55-62.

Fitzgerald, B. and O'Kane, T., "A Longitudinal Study of Software Process Improvement," IEEE Software, May/June 1999, pp. 37-45

IEEE, *Collections of Software Engineering Standards*, IEEE press, 1998.

ISO/IEC 12207, "Software Life Cycle Processes", 1995.

ITMRA(Information Technology Management Reform Act), USA, 1996.

Jones, M. C. and Harrison, A. W., "IS project team performance: An empirical assessment," Information & Management 31, 1996, pp. 57~65.

Kaplan, R. S. and Norton, D. P., "The Balanced Scorecard-Measures that drive Performance," Harvard Business Review, January/February 1992, pp. 71~79

Keil, Mark, "Pulling the Plug: Software Project Management and the Problem of Project Escalation," *MIS Quarterly*, December 1995, 421-447

Nidumolu, Sarma, "The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable," *Information Systems Research*, 6:3, September 1995, 191-219.

Nord, G. Daryl and Jeretta Horn Nord, "Information Systems Project Development: Knowledge and Domain Requirements for the Systems Analyst," *Industrial Management & Data Systems*, 1997, 17-24

Weber, R., *Information Systems Control and Audit*, Prentice-Hall, 1999.