韓國컴퓨터情報學會 論文誌
第16卷 第12號, 2011. 12.

2011-16-12-2-3

소프트웨어 개발 프로젝트를 위한 요구관리도구의 기능요건 연구

박 구 락*

A study of the functional requirements to management tool for software development projects

Koo-Rack Park*

요 약

정보시스템 구축 프로젝트의 복잡성과 규모가 점차 커지면서, 프로젝트가 실패하는 경우가 발생한다. 프로젝트의 실패의 원인을 분석해보면 사용자의 프로젝트 이해 부족, 불명확한 요구사항 및 요구사항변경 등으로 소프트웨어 개발 생명주기 중 요구사항 분석 단계에서 대부분의 원인을 찾을 수 있다. 본 논문은 소프트웨어 개발 프로젝트에서 요구분석을 통해 도출된 기능과 변경요구 기능간의 추적 등 개발 생명주기 전체 기간 동안 지속적인 요구사항 변경 관리가 가능하게 하는 도구를 설계하였다. 그리고 프로젝트 관련자들 사이에서 충분한 합의와 협의를 통해 공통의 이해를 구축하는 효과적인 의사소통을 지원하는데 필요한 기능에 대해 논의한다.

▸ Keyword : 의사결정시스템, 소프트웨어 공학, 프로젝트 관리, 요구사항 분석

Abstract

Information system gradually increases the complexity and scale of the project, while if the project fails to occur. To analyze the causes of failure of the project a lack of understanding of your project, unclear requirements and requirements change, etc. of the software development life cycle from requirements analysis phase is to find the source of most. In this paper, a software development project needs analysis derived from the traceability between features and functionality, and development needs throughout the life cycle requirements during the ongoing change management tool was designed to allow. And among those related to the project through consultation with a sufficient consensus to build a common understanding of effective communication will discuss the features required to support.

▸ Keyword : Decision Support, Software Engineering, Project Management, Requirements analysis

• 제1저자 : 박 구 락
• 투고일 : 2011. 11. 02, 심사일 : 2011. 11. 30, 게재확정일 : 2011. 12. 14.
* 공주대학교 컴퓨터공학과(Dept. of Computer Science & Engineering, Kongju national University)

I. 서 론

정보시스템 구축 프로젝트의 복잡성과 규모가 점차 커져가고 협업을 통한 사업 수행이 많아지면서 위험요인 또한 증가하고 이로 인해 프로젝트의 실패율 또한 높아지고 있다. 소프트웨어 개발 프로젝트에서 대부분의 실패의 원인을 분석해보면 사용자의 프로젝트 이해 부족, 불완전한 요구사항 및 요구사항변경 등으로 소프트웨어 개발 생명주기 중 요구사항 분석 단계에서 대부분의 원인을 찾을 수 있다. 소프트웨어 개발 프로젝트의 성공적 수행을 위해서는 요구사항에 대한 이해관계자들 간의 공통의 이해와 체계적 관리가 중요해 졌다. 현업의 경우 기존 업무를 수행하면도 동시에 프로젝트에 참여하기 때문에 완전한 요구사항을 기대하기 어려운 것이 현실이다. 개발회사의 입장에서도 비용의 초과 지출을 염려하여 고객의 요구사항을 적극 수용하지 못하는 문제점을 갔게 된다. 발주자가 소프트웨어의 요구사항을 처음부터 모두 기능별로 정의하는 것은 사실상 불가능하다. 사업비용 산정을 위한 요구사항 기능 정의는 실제 소프트웨어 개발 생명주기 중 초기단계인 요구사항 분석 및 정의 단계에서 도출되는 것이며 정확한 사업비용을 산정하기 위해서는 요구사항 정의를 위한 별도의 프로젝트를 수행하기도 한다. 실제 소프트웨어 개발 프로젝트 현장에서는 제한된 자원과 일정으로 인해 초기 공정에서 요구사항의 본질을 올바로 파악하지 못하는 것과 프로젝트 참여자들 간의 원활한 의사소통의 부재로 인해 불완전한 요구사항이 도출되고 있다. 이는 개발단계에서 지속적인 변경을 일으킴으로써 프로젝트의 지연과 추가 비용을 발생시키고, 고객이 원하는 완전한 기능을 제공하지 못함으로써 소프트웨어의 전체적인 품질완성도를 저하시키는 문제점을 갔고 있다. 이것은 결국 프로젝트의 성패에 막대한 영향을 미치게 되고 개발회사와 발주자에게 추가 비용 부담을 가중시킨다. 이렇게 복잡하고 어려운 요구사항 분석 및 정의를 요구관리 도구를 활용하여 시스템적으로 효율적인 관리할 수 있도록 하는데 필요한 기능을 도출하고자 한다. 요구관리 도구는 소프트웨어 개발 생명주기 동안 요구사항 변경을 식별하고 통제하며 산출물 사이의 추적과 영향분석을 가능하게 함으로써 요구사항의 일관성을 유지하고 개발 비용의 변화에 대한 프로젝트 이해관계자들 사이에서 충분한 합의와 협의를 통해 공통의 이해를 구축하는 효과적인 의사소통 도구로 활용 가능하게 하여야 한다. 기존의 변경 관리 방법은 요구사항 변경 발생 시 변경을 추적하고 처리 하는데 연구에 초점을 두고 있어 개발비용의 변화에 대한 이해관계자들 사이에 합의와 협의를 위한 효과적인 의사소통 방안의 연구는 부족하다. 요구관리 도구를 통해 변경을 요구하는 고객에게 전체 개발 일정 및 비용의 변화를 인식하도록 함으로서 요구사항 변경 결정이 다른 요구사항에 미치는 영향을 파악하여 적절한 시기에 결정을 내리도록 도와주어야 한다. 이때 필요한 요구관리 도구의 기능이 무엇인지 조사하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 소프트웨어 개발 프로젝트 위험요인에 대한 소개와 국내 SW개발 요구관리 현황 및 필요성을 소개하고, 프로세스 개선을 위한 지침서 등 요구관리 도구가 갖춰야할 기능을 도출하기 위한 관련 연구를 수행하였다. 3장에서는 요구관리도구가 갖춰야할 기능에 대해 제시한다. 4장에서는 결론 및 향후 연구 과제를 제시한다.

II. 관련 연구

1. 소프트웨어 개발 프로젝트의 위험요인

소프트웨어 개발 프로젝트를 성공적으로 완수하기 위해서는 무엇보다 위험요인에 대한 적절한 대처 방안을 마련하는데 있을 것이다. 소프트웨어 개발 프로젝트의 위험이란 프로젝트의 성공/실패와 관련된 불확실성과 프로젝트 실패로 인한 잠재적인 손실이라는 두 가지 차원으로 정의할 수 있으며, 결국 소프트웨어 개발 프로젝트에서 통제 되어야 하는 핵심 요소가 바로 프로젝트의 위험 요인이 된다는 것으로 이해할 수 있을 것이다[1,2,3]. 소프트웨어 개발 프로젝트 초기단계에서 요구사항을 완벽하게 정의하기란 사실상 불가능하다. 불확실한 요구사항을 바탕으로 진행되는 프로젝트에서 요구사항변경은 불가피한 현실이다. 이러한 위험요인을 최소화 하고 효과적으로 관리하기란 쉽지 않다[4]. 발주사의 경우 세부적인 기능을 제시하지 않고 포괄적인 기능요구사항만을 제시하는 경우가 많다[5]. 개발회사의 경우는 이러한 요구사항을 개발자의 경험에 의해 일정 및 투입인력공수 등을 막연하게 정하게 됨으로 프로젝트 진행 중에 잦은 요구사항분석 및 개발일정 변경 등이 발생하게 된다[1]. 소프트웨어 개발 프로젝트 성공적 완수란 발주자의 경우 요구사항을 완벽하게 반영하고 납기일 내에 인도되는 것을 의미할 것이며, 개발회사의 경우 사업초기 예상한 이익을 실현하고 납기일 내에 개발하여 고객에게 인도하는 것을 성공의 중요한 판단 기준으로 여기고 있다. 개발회사의 경우 요구사항 변경은 안정화 또는 유지보수 기간을 통해 개발할 사항으로 보는가 하면, 발주자의 사업기간 내에 요구사항이 반영되어 인도 받기를 원한다. 발주자의 경우 우수

한 성능의 시스템을 결과물로 얻고자 하여 개발회사에게 요구사항 수용을 강요할 것이며 개발회사의 경우 발주사와의 좋은 관계를 유지고하 싶어 하기 때문에 쉽게 거절할 수 없는 것이 현실이다. 개발회사의 경우 사업기간 및 개발비용의 초과로 인한 손해가 발생하게 되면 사실상 개발회사 입장에서는 실패한 프로젝트가 되지만 발주자에게는 성공적인 프로젝트가 될 것이다.

2. 국내 SW개발 요구관리 현황 및 문제점

소프트웨어 개발 프로젝트에서 고객의 기대를 만족시키기 위한 중요한 공정 중의 하나는 요구사항을 명확하게 정의하는 것이다. 이것은 신규 개발하거나 개선하려고 하는 어플리케이션의 방향과 목표를 제시하는 것과 같기 때문이다. 국내 소프트웨어 개발 프로젝트 실패의 원인 중 대부부이 요구사항 정의를 소홀이 한데서 기인한다. 최근 국내 기업의 소프트웨어 개발 프로젝트는 다양한 분야와의 협업으로 분산되고 복잡해지면서 점점 대형화되고 있다[6]. 반면 개발업체들 간의 경쟁과 개발기술의 어려움, 개발 주기의 단축 등으로 인해 통합개발되지 못하고 분산 아웃소싱 되면서 요구사항 관리는 점차 어려워 졌다. 글로벌 시장조사기관인 Standish Group의 자료를 보면 해외 IT프로젝트 실패 항목들 중 50%가 요구관리와 관련되어 있고, 프로젝트의 전체 비용 중 7,80%가 요구관리 실패로 인한 재작업 비용이라고 한다[7]. 그림 1은 요구사항 에러의 발견이 늦어질수록 비용이 증가함을 보여준다. 이러한 조사 결과는 해외뿐 아니라 국내 프로젝트에도 적용된다[6,8].

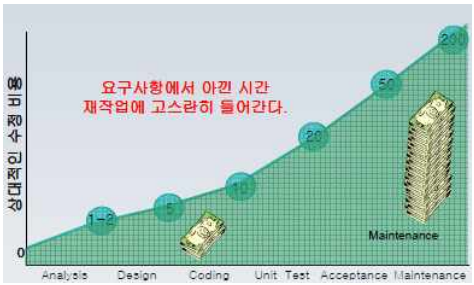

그림 1. 요구사항 에러 발견된 단계별 비용
Fig. 1.Step-by-step cost of requirements errors found

이러한 결과에 대한 가장 큰 이유는 다음 두 가지로 요약될 수 있다. 하나는, 불충분한 요구사항 정의와 이해관계자간의 의사소통의 문제 이다. 대부분의 짧은 개발 일정 때문이기도 하지만 요구사항의 본질을 정확하게 전달하지 못하고 파악하지 못해서 이다. 또 하나는 요구사항에 대한 변경관리가 제대로 이루어지지 못해서 고객의 변경요구가 개발 산출물에 대한 영향이나 연관성을 분석하지 못해 계획한대로 프로젝트가 진행되지 않는다. 국내 기업들은 개발 역량을 향상시켜 프로젝트를 성공적으로 수행하기 위해 많은 노력을 하고 있으나 불명확한 요구사항 및 잦은 요구사항 변경과 같은 외적인 위협 요소에 의해 프로젝트가 실패하는 경우도 있다[9]. 이러한 외적 위협을 차단하거나 최소화하기 위해서는 이해관계자간의 프로젝트에 대한 공감대 형성을 위한 의사소통 방안을 마련해야 하며 기존의 요구사항관리 방법에서 벗어나 공학적이고 체계적인 문제해결 방법에 의한 요구사항 관리가 요구된다[6].

국내 개발회사 대부분은 요구사항의 정의 및 변경 요구사항에 대한 관리를 엑셀이나 이와 유사한 프로그램을 이용해 요구사항을 관리하고 있다. 그림 2는 정보화사업 수행부분별 요구사항 관리도구 활용현황을 보여주고 있다[10]. 일부 회사에서는 외국의 유명 요구사항관리 도구를 도입하거나 자체 개발하여 요구사항관리 도구로 활용하고 있다. 엑셀을 이용하든 전문적인 도구를 이용하든 국내 개발회사 일부를 제외한 대부분은 요구사항관리 전담 인력을 배치하지 않고 개발자에게 개발과 요구사항관리를 병행하고 있는 것이 현실이다.

그림 2. 요구 관리 도구
Fig. 2.Requirements Management Tools

소프트웨어 개발 프로젝트에서 소프트웨어의 품질을 결정짓는 중요한 요소인 요구사항관리를 프로세스, 인력, 기술, 문화 측면에서 분석해 보면, 프로세스 측면에서는 기업들의 요구관리에 대한 필요성 인식하지 못하거나 수용의지가 부족하고 요구관리 프로세스가 없거나 관리되지 않고 있다. 그림 3은 요구사항 관리의 표준 프로세스를 보여주고 있다. 하지만, 그림과 같이 요구사항 정의 및 변경관리 프로세스가 지켜지지 않고 있는 상황에서 요구관리의 반복적인 수행에 따른 일관성 및 추적성을 확보하기 어렵다. 또한 지속적이고 종합적인 요구관리 개선 노력도 찾아보기 어렵다. 이러한 현실에서는 요구관리상의 문제점들을 지속적이면서 종합적인 개선을 통해 해결하기보다는 임시방편적으로 해결하기 때문에 문제의 근본 원인은 해결되지 않고 되풀이된다[6].

요구사항관리가 잘 이루어지지 않는 원인을 인력 측면에서 보면 수요인력이 절대적으로 부족한데 있다. 한정된 요구관리 인원 및 전문가의 부족, 요구관리 전문성 향상을 위한 교육의 부족, 요구관리로 인한 추가비용의 발생 등 인력운영과 관련

된 부분이 상당히 존재한다. 요구사항의 추적성 및 일관성을 확보하지 못한 상태에서 요구사항 변경이 발생하게 되면 시스템 간의 영향분석을 위한 추가 시간과 노력을 들여야만 한다. 이렇듯 요구관리를 위한 인적 자원의 확보는 매우 중요한 요소이다. 기술 측면에서도 요구관리 도구를 도입하지 않거나 효과적으로 활용하지 않는데 있다. 요구관리 자동화 도구는 요구관리를 효율적으로 수행할 수 있는 방법이다. 자동화를 위한 템플릿이 없다면 요구사항 추출/분해/정의 방법 또는 기법의 적용이 쉽지 않고 문서화도 어렵게 되어 요구관리 작업은 비효율적으로 수행될 것이다. 요구관리를 위한 중장기적인 기술 개선 전략 로드맵이 없이 단기적인 문제 해결을 위해 기술을 도입하는 것 또한 문제이다. 기업의 문화적인 측면에서 보면 경영진의 무관심과 현실을 무시한 과도한 수행 의지, 프로세스 개선 효과의 불확실성, 제품/시스템 정의 미흡으로 빈번한 요구 변경 등이 있다. 해외 유명 기업인 SAP, ORACLE, IBM과 같은 기업들의 요구관리 체계를 보면 대부분 전문 요구사항관리 도구를 활용하여 시스템적으로 관리하고 개발 업무의 일부로 정의하고 전사적인 관점에서 요구관리를 통제하고 개선하는 것을 볼 수 있다[6].

요구관리의 공통적인 문제점 및 국내 기업의 문제점, 해외 선진 기업의 모범 사례 등을 비교 분석하면 국내 기업이 나아갈 방향을 정리할 수 있다. 바로 시스템적이고 종합적인 요구관리다. 국내 IT 기업에서 요구관리에 대한 지속적인 관심 및 노력, 올바른 투자를 한다면 다른 프로세스 영역보다 효과가 보다 빨리 현실화 될 수 있다. 기업 문화에 요구관리가 체질화까지 된다면 고품질의 시스템 및 소프트웨어를 신속하게 생성할 수 있는 역량을 갖게 된다[6].

3. 프로세스 개선을 위한 지침서

요구관리 도구는 국제적 규약 및 지침을 준수 할 수 있도록 개발되어야 한다. 뿐만 아니라 품질보증/평가/관리를 위한 다양한 기능을 요구한다.

3.1 CMMI 모델 구성요소를 반영한 UML 기반의 프로세스 모델링 v1.0

프로세스는 고객의 요구와 기대를 충족하기 위해 자원 및 기술을 이용하여 제품과 서비스로 변환시키는데 필요한 일련의 활동과 이와 관계되는 사람들, 활동에 필요한 도구들의 집합을 의미 한다. 이러한 프로세스를 이해하고 프로젝트관리에 사용하며 해당 프로세스를 개선하거나 수행하기 위해 프로세스 모델의 필요성이 대두되었다. 소프트웨어는 프로세스의 수행을 통해 생산되는 것이므로 소프트웨어 개발 프로세스가 정확하고 우수하며 잘 지켜졌을 때 좋은 품질의 소프트웨어를 생산할 가능성이 높다. 따라서 각 조직들은 우수한 품질의 제품을 얻기 위해 프로세스를 지속적으로 개선하기 위한 노력을 해왔고 CMMI 및 PMBOK 등과 같은 소프트웨어 프로세스 개선을 위한 모델들을 이용하여 왔다[11].

3.2 프로덕트 라인 아키텍처 모델링

소프트웨어 재사용을 위한 새로운 패러다임으로 프로덕트 라인 공학 기술이 대두되고 있다. 프로덕트 라인 아키텍처는 요구사항의 변화에 따른 컴포넌트의 신속한 조립을 지원하기 위한 핵심 자산이다. 재사용 가능한 프로덕트 라인 아키텍처 개발을 지원하기 위해 필요한 아키텍처의 가변성을 정의하고 이를 명시적으로 표현하기 위한 기능을 필요로 한다[11].

3.3 품질관리활동 체계서

소프트웨어 개발 프로젝트에서 고객의 요구 기대를 충족하기 위한 품질관리 활동에는 품질보증활동 및 품질평가활동이 있다. 이러한 활동을 체계적으로 지원하고 관리하기 위한 많은 연구문헌을 볼 수 있다.

품질보증활동은 품질기능전개를 활용한 최신의 품질보증 체계에 대한 이해와 요구기능을 정의하고 품질보증 체계를 구축하려는 국내 관련 기업 및 기관의 여건에 적합한 품질보증 체계를 구축할 수 있도록 지원하며 자체적으로 품질보증 체계를 구축하고자 하는 관련 기관에서 활용할 수 있다.

품질평가활동은 품질평가 체계를 구축하기 위한 품질특성을 체계적으로 정리한 것으로 이를 위해 국제표준인 ISO/IEC 9126을 도입하여 국내 여건에 적합한 품질평가 체계를 구축하여 자체적으로 품질평가를 수행하고자 하는 소프트웨어 개발회사, 제3자 품질평가 기관에 활용할 수 있다. 이러한 품질특성은 품질평가 기술과 수집될 데이터를 결정하는데 쓰이게 되는데, 품질특성에 대한 관점에 따라 평가기술 및 수집 데이터가 달라질 수 있다[11].

품질평가활동 절차를 정의하여 품질평가 수행 프로세스에 대한 상세한 지침을 제공한다. 품질평가활동의 복잡성을 해결하기 위해 평가절차와 기법의 재사용을 용이하게 할 목적으로 품질평가를 위한 제반 사항을 문서화하여 평가에 활용하게 하여 소프트웨어 제품에 대한 품질평가를 통해 문제점을 발견하게 되면 개발자에게 피드백을 제공하고 이를 개선하게 함으로써 품질을 보증할 수 있게 한다. 소프트웨어 품질평가를 수행하고자 하는 평가 기관이나 소프트웨어 품질 향상을 추구하는 개발회사에서 활용함으로써 고품질의 소프트웨어 개발을 지원할 수 있다[11].

3.4 소프트웨어 시험/인증 매뉴얼 및 절차서

소프트웨어 시험/인증을 위한 제반 활동에 관한 가이드와

구체적인 조직의 업무 분장 및 상세한 절차를 제시 한다. 이는 독립적인 소프트웨어 시험/인증기관이나 기업 내부의 시험/인증부서에서 시험/인증의 효과적인 수행을 위해 활용될 수 있다[11].

III. 요구관리도구의 기능요건

1. 목적

소프트웨어 품질의 근간인 요구사항관리 방안 및 솔루션을 제시하고 효과적인 의사소통 도구로 활용하기 위한 필수 기능을 정의하고자 한다. 국내 기업들은 요구관리의 중요성을 점차 인식하고 시스템적으로 관리할 수 있도록 하기위해 도구를 도입하려 하고 있다. 개발회사의 입장에서는 개선해야할 당면과제이면서 새로운 비즈니스 도전 과제이기도 하다. 본 연구를 통해 요구관리의 필요성 및 효과에 대해 조사하면서 요구관리의 중요성에 비해 이를 관리할 수 있는 시스템적 지원은 너무나도 미흡하였다. 본 연구에서 제시하는 요구관리를 위한 의사결정지원시스템은 개발회사와 발주사 도두에게 좋은 결과를 가져올 것으로 기대한다. 그림 3은 일반적인 요구사항 관리 프로세스로 IBM사의 요구사항관리 시스템인 DOORS의 요구사항 관리 프로세스를 도식화 하여 보여주고 있다[8]. 고객으로부터 도출된 요구사항을 수집하여 문서화하며 요구사항간의 관계를 추적할 수 있도록 분석 정리하고 검증하여 고객과의 협의 및 합의 후 베이스라인을 확정하게 된다. 이후 발생한 변경이나 추가요구는 모두 요구사항변경관리 프로세스를 따르게 된다.

그림 3. 요구사항 관리 프로세스
Fig. 3. Requirements management process

2. 요구관리의 가시성 확보

소프트웨어 개발 프로젝트에서 요구사항 분석 및 정의는 무엇보다 중요하지만 많은 프로젝트에서 프로그램화 하기위한 요건위주로 요구사항을 정의하는 경우가 많다. 발주자는 업무적인 언어로 의사를 전달하려하고 개발자는 IT적 언어로 받아 드리려 한다. 서로 다른 관점에서는 요구사항도출이 원활하게 이루어질 수 없다. 이렇게 서로 다른 생각을 정리하고 원활한 대화를 유도하기 위해서는 의사소통을 위한 문서화 및 공유화 이해관계자 간의 합의를 위한 관리기능을 제공해야 한다. 가시성이 확보된 요구관리는 테스트 엔지니어가 초기에 테스트 계획 또는 테스트케이스를 만들 수 있게 해 줄 것이다. 요구사항과 테스트 간의 연결로 비즈니스와 고객의 필요를 만족함을 확인할 수 있어 결과적으로 개발 산출물을 적시에 출시하도록 도움을 줄 것이다.

3. 요구사항의 일관성 확보

요구사항의 일관성이 확보된 요구관리는 개발자와 발주자 간의 합의된 동일한 요구사항과 함께 작업할 수 있게 함으로써 품질을 향상시킬 것이다. 모든 이해관계자들과 비즈니스 목표에 대한 테스트 커버리지를 확보함으로써 품질완성도를 담보하게 된다. 일관성이 확보된 요구관리는 프로세스 지향적이고 자동화된 협업 환경을 통한 재개발을 줄임으로써 생산성과 효율성 향상을 통해 비용 절감 효과를 가져 온다[8].

4. 요구사항과 품질관리

품질이란 고객의 요구사항에 잘 부합되는 정도를 의미 한다.또한 고객의 기대에 얼마나 충족 했는가를 말한다. 시스템적으로 관리되는 요구사항은 품질관리를 위한 여타 시스템과 연계가 가능하고 기존의 수작업에 의한 불필요한 노력을 감소시킨다. 이렇게 관리되는 요구사항은 가시성과 예측성을 확보하게 되어 이해관계자의 논리적 의사결정을 지원하게 된다. 요구사항의 가시성은 테스트 엔지니어가 좀 더 빠른 시기에 테스트 계획을 세울 수 있게 해주며 마지막 단계에서 재작업 및 납기 지연을 줄여줄 것이다. 요구사항 변경에 대한 테스트 계획에 미치는 영향을 분석할 수 있는 능력은 스케줄, 위험, 비용 등과 같은 이슈를 논리적으로 의사결정 할 수 있도록 지원한다. 요구사항에 문제가 있어 변경요구사항이 발생한다면 개발자는 재작업을 피할 수 없을 것이다. 그림 4는 품질관리 시스템을 통한 자산 확보 및 재사용 프로세스를 보여주고 있다[10]. 이러한 품질관리 시스템의 구축은 테스트 단계별 품질관리 프로세스 표준화 및 시스템화를 지원하고 양질의 테스트 케이스를 확보할 수 있게 한다. 또한 품질관리 프로세스 전반의 정보 자산에 대한 체계적인 관리 및 재사용이 가능하게 한다.

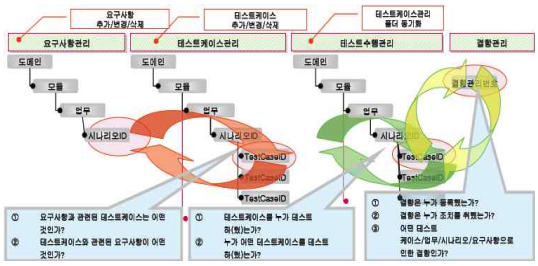

그림 4. 품질 관리 시스템을 통한 자산 확보 및 재사용 프로세스
Fig. 4. Secured through a quality management system and reuse process assets

5. 요구관리 기반 테스트

소프트웨어 개발 프로젝트에서 요구관리를 시스템적으로 지원하는 것은 요구사항이 도출 되었을 때 테스트 계획 수립이 가능하다는 것이며 개발 프로세스 중에도 가능한 한 빨리 테스트를 수행 할 수 있게 한다. 요구사항을 기반으로 테스트하는 것은 소프트웨어가 고객의 기대를 만족시키는지를 보장해 준다. 요구사항과 테스트를 연결하여 관리하는 것은 결함 발생 시 해당 결함과 관련된 요구사항을 분석할 수 있음을 의미한다. 발견된 요구사항은 다른 요구사항에 어떠한 영향을 미칠지 파악할 수 있다.

6. 프로젝트 진행상황 파악

목표를 정하고 관련 요구사항을 만족했는지를 기준으로 테스팅 진척사항을 측정할 수 있게 함으로써 프로젝트가 계획에 따라 수행되고 있는지 지연되고 있는지 쉽게 파악할 수 있게 한다. 특히 대시보드 형태의 가시적인 진행상황에 대한 정보 제공은 이해관계자에게 진청상황에 대한 이해와 문제의 심각성을 확실하게 전달할 수 있는 방법이기도 하다. 또한 특정 요구사항이 변경되었을 때 이와 연관된 상세설계서의 요구사항들에 변경사실을 통보해 주는 기능이 필요하다. 아무리 시스템적으로 요구사항을 관리하더라도 개발자가 인지하지 못한다면 문제를 야기할 수 있다. 그림 5는 프로세스별 계획대비 실적을 도식화한 대시보드를 보여주고 있다[10]. 이러한 대시보드는 주요 성능 지수 및 실시간 프로젝트 관리 지수를 도식화하여 보여줌으로써 관리 가시성을 확보할 수 있게 하고 이해관계자별 개인화 페이지/뷰를 구성하여 특성화된 정보를 제공한다.

그림 5. 진행상황 모니터링 대시보드
Fig. 5. Progress monitoring dashboard

7. 요구관리 협업 환경 지원

소프트웨어 개발 프로젝트에서 서로 다른 분야의 개발자들이 자기가 익숙한 도구를 사용하여 다른 분야의 정보를 접근하고 관리하는 것은 흔한 일이다. 서로 다른 분야와 도구를 사용함으로써 요구사항 변경 등에 대한 통합관리, 변경영향 분석 및 통제 등을 어렵게 한다. 이것은 소프트웨어 품질에 나쁜 영향을 미칠 것이며 국제 표준 및 규칙에 대한 감시를 어렵게 한다. 요구관리 도구는 다양한 환경 및 사용자의 동시 작업을 지원하고 공통된 기법으로 도구를 활용할 수 있도록 하여야 한다.

8. 요구사항 추적성 관리

소프트웨어 개발 프로젝트에서 추적성이란 파일단위의 추적성이 아닌 프로젝트 산출물(문서) 내의 단위 요구사항간의 추적성을 의미한다. 발주자는 불안전한 요구사항을 제시하는 경우가 많다. 결과적으로 변경 요구사항이 많아질 수밖에 없다. 개발자는 시스템이 모든 요구사항을 만족하는 지를 확인해야 하고 각 단계마다 요구사항이 빠짐없이 올바르게 반영되었는지 확인하여야 하며 요구사항 변경에 따른 영향분석과 시험 또는 설계 변경에 따른 영향을 받는 요구사항 영향 평가를 실시해야 한다. 이러한 복잡하고 어려운 일을 수작업으로 처리해야 한다면 많은 시간을 소요해야 할 것이다. 요구관리 도구의 요구사항 추적관리 기능은 필수요건이다. 상위/하위 요구사항 간의 추적성 확보 및 요구사항과 테스트 정보간의 추적성 확보가 중요하다. 그림 6은 소프트웨어 개발 생명주기 동안의 요구사항관리와 품질관리간의 상관관계를 보여주고 있다[8]. 시스템을 통한 요구관리와 품질관리는 팀이 공유할 수 있는 명확한 공통요구사항 저장소를 제공함으로써 중요 요구사항의 누락을 방지하고 요구사항 변경영향 평가가 가능하

게 하며 시험해야 할 중요 요구사항을 식별하게 한다. 또한 전 주기에서 감시 및 통제가 가능하게 한다.

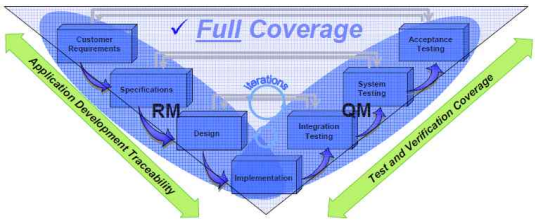

그림 6. 전체 생명주기동안 협업과 추적성
Fig. 6.Collaboration and traceability during the entire life cycle

그림 7은 비즈니스 요구사항이 시스템, 설계 테스트 요구사항까지 어떠한 연결 관계를 갔고 있는지 도식화하여 보여주고 있다. 이러한 문서위주의 요구사항관리를 그림 8과 같이 도식화하여 요구사항과 시스템요구사항의 관계를 좀 더 시각적으로 보여주고 있다[8].

그림 7. 단위 요구사항 추적성
Fig. 7. Unit Requirements Traceability

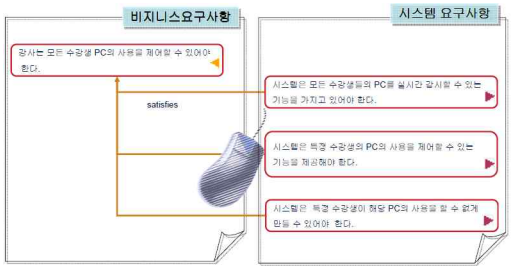

그림 8. 단위 요구사항 도식화
Fig. 8.Graphs the unit requirements

IV. 결론 및 향후 연구 과제

본 연구에서 제시하는 요구관리 도구의 기능 요건들은 요구사항변경 수용을 위한 논리적 의사결정을 지원하는 도구의 설계에 유용하게 활용될 것으로 기대한다. 본 연구에서는 고객의 불확실한 요구사항을 분석하고 분해하여 재정립함으로써 완벽한 요구사항을 정의할 수 있도록 시스템적으로 지원하는 도구 설계를 위한 기능 요건정의를 위한 것이다. 주요 기능으로 요구사항의 가시성, 일관성 및 추적성, 변경에 대한 영향분석, 협업지원, 프로젝트 진행상태 파악, 테스트 계획지원, 합의 및 협의 지원 등 프로젝트 관리에서 매우 중요한 요구사항에 대한 체계적 관리를 위한 필수 기능이다.

향후 연구 과제로는 요구사항변경과 그 결과에 대한 가치를 정량화하여 요구사항변경의 발생이 프로젝트에 미치는 영향을 객관적으로 판별 할 수 있게 하고 고객의 요구사항 도출 시 완성도 높은 요구사항분석 및 정의를 위한 프로젝트관리 도구로 의사결정지원시스템을 재설계 할 필요가 있다.

참고문헌

[1] Andriole, S., “Managing Systems: Requirements, Methods, Tools and Cases”, McGraw-Hill, 1996.
[2] Andriole, S., “Managing Systems: Requirements, Methods, Tools and Cases”, McGraw-Hill, 1996.
[3] Lock, S., and Kotonya, G., "Requirement Level Change Management and Impact Analysis," Cooperative Systems Engineering Group, Technical Report Ref: CSEG/21/1998, October 1998.
[4] Bosch, J., “Design and use of software architectures”, Addison Wesley, 2000.
[5] Lock, S., and Kotonya, G., "Abstract: An Integrated Framework for Requirement Change Impact Analysis," Requirenautics Quarterly: The Newsletter of the Requirements Engineering Specialist Group of the British Computer Society, Issue 18, January 2000.
[6] http://www.ciobiz.co.kr/news/articleView.html?idxno=4584
[7] Standish Group, 2001
[8] dae seung kim, “Rational DOORS + Rational Quality Manager”, IBM, 3, 2010.
[9] Mikyeong Moon, Keunhyuk Yeom, “An Approach to Managing Requirements as a Core Asset in Software Product - Line”, Journal of KIISE, vol.31, no.8, pp.1010-1026. 8, 2004.

[10] http://www.oracle.com/kr/products/091027-igo ver nment-session04-065496-ko.pdf, HP Software Solu tions, 04, 2008.
[11] http://spic.kaist.ac.kr, SPIC

저 자 소 개

박 구 락
1986 : 중앙대학교 전기공학과 공학사
1988 : 숭실대학교 전자계산학과 공학석사
2000 : 경기대학교 전자계산학과 이학박사
현 재 : 공주대학교 컴퓨터공학부 교수
관심분야 : 정보경영, 정보통신, 전자상거래
Email : ecgrpark@kongju.ac.kr