DOI: 10.3745/KIPSTD.2008.15-D.4.523

위험요소의 상태분석에 의한 프로세스 개선에 관한 연구

이 은 서†

요 약

소프트웨어 개발 시, 생명주기의 프로세스 개선에 저해 요인이 되는 결함이 다수 존재한다. 생명주기의 저해 요인을 제거하고 동시에 체계적으로 이를 관리하기 위하여 본 논문에서는 위험요소의 관리방안을 제안한다. 유사한 프로젝트를 수행 시 영역 전문가의 지식을 활용한 결함요소의 상태전이를 관리하여 발생되는 문제점을 예측, 대비할 수 있게 하여, 소프트웨어 프로세스를 개선할 수 있다. 본 연구에서는 소프트웨어 개발 시 발생하는 위험요소 관리에 대한 결함의 전이를 찾아내고, 예방 및 원인을 식별하고자 한다. 또한 이를 정량화 하여 전이단계를 제시한다.

키워드 : 위험관리, 소프트웨어 프로세스 개선, 결함관리, OT 프레임워크

A Study for Process Improvement by State Analysis of Risk Items

Lee Eun-Ser†

ABSTRACT

There are many defects that cause the process improvement of lifecycle problems during software development. This paper propose the management method of risk items that removes and manages the lifecycle problems as well. For the similar projects, we can estimate defects and prepare to solve them by using domain expert knowledge and the state analysis of defect items, which can greatly improve the software process. This research provides solution of management of risk items problem and detection of defect transition and its prevention and causes that happen on software development. Also, In this paper propose to making quantity of degree and transition phase.

Key Words : Risk Management, Software Process Improvement, Defect Management, Opportunity Tree Framework

1. 연구배경

소프트웨어 공학은 소프트웨어 개발과 운용, 유지보수를 합리적으로 행하기 위한 사고방법, 절차, 조직 및 기법 등을 총칭한 것으로, 기술적인 측면과 관리적인 측면의 두 가지 기능을 가지고 있다. 기술적인 측면은 소프트웨어를 구성하는 구성 요소의 적합성과 이의 효과적인 조합에 의한 효율의 극대화를 추구하고 있고, 관리적인 기능은 소프트웨어 개발에 관련된 업무가 제대로 이루어지도록 하기 위해 인원, 설비, 자재 등에 대해 계획을 세우고, 통제하는 관리기술을 의미한다[1].

관리적인 기능을 어떻게 적용하느냐에 따라서 같은 환경에서도 다른 산출물이 생산되게 된다. 이와 같은 형태는 관리 방안 및 정책으로 나타나게 된다.

오늘날 소프트웨어 과학자들과 기술자들은 점차 소프트웨어 시스템이 대형화함에 따라 소프트웨어 개발 방법, 유지비용의 절감과 품질보증에 큰 관심을 갖고 연구를 계속하고 있다. 이를 위해 소프트웨어 생명주기와 중간 제품을 분석하여 소프트웨어 비용과 품질에 영향을 주는 요인을 측정하려는 시도가 이어지고 있다[2].

품질에 영향을 미치지 않게 하기 위하여 각 사항들이 위험요소로 전이되는 것을 예방하고 예측하여 품질관리를 하려는 노력이 대두되고 있다.

따라서 본 논문에서는 품질관리와 연관된 위험요소 발생의 진척상황을 전이 단계를 제시하여 예방 및 원인 제거를 수행하였다. 이를 수행하기 위하여 정량적인 접근법으로 단계를 제시하고 각 단계의 분석을 통한 관리를 수행하고자 한다. 또한 이를 활용하여 유사한 영역의 프로젝트에서 활용하여 결함을 관리할 수 있게 된다.

2. 기본 개념

2.1 결함

결함 정보는 다른 유형의 원시데이터이며, 소프트웨어 프로젝트에서 매우 중요한 것이다. 결함은 소프트웨어의 품질과 직접 관련이 있으므로 여러 의미에서 결함 데이터는 공수 데이터보다 더 중요하다[3, 4, 5].

결함 데이터는 우선 프로젝트 관리를 위해 필요하다. 대규모 프로젝트는 수천개의 결함을 포함할 수 있는데, 이 결

† 종신회원 : 안동대학교 컴퓨터공학과 조교수
논문접수 : 2008년 3월 14일
수 정 일 : 1차 2008년 4월 24일, 2차 2008년 5월 30일
심사완료 : 2008년 5월 31일

함은 다른 사람들이 프로젝트의 각각 다른 단계에서 발견하게 된다. 흔히 프로세스에서 결함을 고치는 사람과 결함을 발견하거나 보고하는 사람은 서로 다르다[6]. 보통 프로젝트는 소프트웨어가 최종 인도전에 발견한 모든 또는 대부분의 결함을 제거하려고 할 것이다. 이런 시나리오에서 결함 보고와 해결은 비공식적으로 수행할 수 없다. 비공식적인 메커니즘의 사용은 발견한 결함에 대해 잊어버리는 결과를 가져올 수 있으므로 결함을 제거하지 않거나 다시 찾는데 추가 공수가 들어갈 수도 있다. 그러므로 적어도 결함을 기록해야 하고 해결할 때까지 추적해야 한다. 이런 절차를 위해서 결함의 징후, 의심되는 결함의 위치, 발견자, 해결자 등과 같은 정보가 필요할 것이다. 따라서 결함이란 프로젝트의 작업 산출물에서 발견되는 것으로 이 때문에 프로젝트의 목표를 달성하는데 부정적인 영향을 끼칠 수도 있다[7, 8].

결함에 대한 정보는 여러 가지 다양한 결함 검출 활동을 통해 발견한 결함의 개수도 포함 한다. 그래서 요구 사항 검토, 설계 검토, 코드 검토, 단위 테스트, 그리고 다른 단계에서 발견된 모든 결함 등을 기록한다. 프로젝트의 서로 다른 단계에서 발견된 결함 개수의 분포 데이터 역시 프로세스 역량 기준선(Process Capability Baseline) 생성에 사용한다[9, 10, 11]. 아울러 PDB(Process Data Base) 입력항목에 몇 가지 설명이 기록되는데, 예측에 대한 설명과 위험 관리에 대한 설명이 해당된다.

2.2 위험분석

위험이 소프트웨어 프로젝트에만 한정된 것은 아니다. 소프트웨어가 성공적으로 개발되고 고객에게 인도된 후에도 위험은 일어날 수 있다. 이러한 위험은 보통 현장에서의 소프트웨어 실패와 관련이 있다[12, 13].

비록 잘 공학화된 시스템이 실패할 확률이 낮다고 해도, 컴퓨터-기저 제어 또는 감시 시스템에서 검출되지 않은 결점은 막대한 경제적 손실을 초래하거나 사람이 부상하거나 죽음을 초래할 만큼 더욱 나쁘고 심각할 수 있다. 그러나 컴퓨터-기저 제어와 감시 시스템의 비용상의 이점과 기능상의 이점은 보통 이러한 위험보다 중요하다. 오늘날 컴퓨터 소프트웨어와 하드웨어가 안전을 증시하는 시스템을 제어하기 위해 사용되고 있다.

소프트웨어 안정성과 위험분석은 소프트웨어에 부정적으로 영향을 미치는, 그리고 전체 시스템을 작동하지 않게 만드는 잠재적인 위험의 식별과 평가에 초점을 맞춘 소프트웨어 품질보증활동이다. 만약 소프트웨어 공학 프로세스의 초기에 치명적 위험을 식별할 수 있다면, 소프트웨어 설계 특징들을 명시할 수 있고, 잠재적인 치명적 위험을 제거하거나 조정할 수 있다.

2.3 OT(Opportunity Tree)의 필요성

실제 프로젝트 수행 시에, 관리자들은 결함을 관리한다. 그리고 개발일정에 맞추기 위하여 많은 수의 결함들은 완전히 조치를 취하지 못하고 다음 개발 단계로 진행되기도 한다. 이와 같은 현상은 많은 프로젝트에서 발생하고 있다. 그 이유는 관리자의 경험부족과 일정상의 여유가 없기 때문인 경우가 대다수이다[14, 15].

결함을 체계적으로 관리한다는 것은 결함을 분류하고 원인을 찾을 수 있으며, 이에 대한 대처 및 예방을 할 수 있는 것을 의미한다. 따라서 결함을 체계적으로 관리하는 것이 필요하게 되는데, 이러한 체계적인 관리를 위하여 OT(Opportunity Tree)를 사용하게 된다.

OT(Opportunity Tree)는 해결하려는 목표를 설정하게 되고, 전체 목표를 해결하기 위하여 요구되는 하위 목표를 결정하게 된다. 즉, 하위 목표를 달성하게 되면 상위 목표가 달성될 수 있게 된다. 그리고 목표 설정과 함께 목표를 해결하기 위하여 요구되는 문서와 전문가의 노하우를 OT(Opportunity Tree)화 하여 사용자에게 지침을 제공하고자 하는 것이 OT(Opportunity Tree)의 목적이 된다. 따라서 본 논문에서는 결함 관리를 전체 목표로 하여, 이를 달성하기 위한 하위 목표를 결정하고 해결책을 제시하여 사용자가 결함을 관리할 수 있도록 하는 것이 최종 목표가 된다. 또한 유사한 프로젝트를 시작하는 경우, OT(Opportunity Tree)를 활용하여 결함을 예방하고, 발생될 수 있는 결함을 예측하여, 이를 대처하기 위하여 일정 계획 수립 시, 참조가 될 수 있다.

2.4 소프트웨어 프로세스 개선의 필요성

프로세스를 개선하는 합리적인 방법은 프로세스의 특정한 속성들을 측정하고, 이들 속성에 근거해서 의미있는 메트릭스의 집합을 개발한 후에 특성에 관한 전략을 이끌어 내는 지표를 제공하기 위해서 이들 메트릭스를 사용하는 것이다. 그러나 우리가 소프트웨어 메트릭스와 소프트웨어 프로세스 개선에 있어서 영향을 말하기 전에, 프로세스는 소프트웨어 품질과 조직의 성능을 개선하는데 제어할 수 있는 많은 인자중의 하나라는 것이다[16, 17]. 따라서 소프트웨어 품질에 많은 영향을 주는 인자가 결함관리가 된다. 소프트웨어 품질과 조직의 성능에 영향을 주는 것으로는 프로세스를 기반으로 제품, 사람, 기술이 된다. 또한 제품, 사람, 기술이 프로세스를 일반화하는 과정에서 많은 결함들이 발생하여서 전체적인 소프트웨어 프로세스의 수준을 격하 시키게 된다. 그러므로 결함 관리의 필요성이 요구되게 된다.

프로세스 개선을 위하여 소프트웨어의 실패 분석이 필요하며, 이를 위하여 다음과 같은 방법으로 수행된다.

- 모든 오류와 결함은 원인으로 분류한다.
- 각 오류와 결함을 수정한다.
- 각 범주에 속하는 오류와 결함을 계산하여 내림 차순으로 정리한다.
- 결과 자료는 조직에 가장 많은 비용이 드는 범주를 발견하기 위해서 분석한다.
- 계획은 가장 비용이 드는 오류와 결함의 종류를 제거할 의도로 프로세스를 수정하기 위해 개발된다.

기존의 위험요소를 측정하기 위해서 많은 노력을 하고 있다. 많은 기업에서 위험요소의 분석 시, 개발되는 생산품에 주요 기능에 해당하는 위험요소인지 부가기능에 의하여 주요 요소에 영향을 주지 않는 위험요소인지를 식별하고 있다. 이와 같은 연구방향은 불량률을 줄이기 위한 6-시그마와 같은 도입을 가져왔으며, 정량적인 위험요소 분석을 위한 필요성이 대두되게 되었다.

그러나 위험요소를 분석하여 연관성을 정량화 하여 이를

분석하기 위한 연구는 많이 진행 되어 있지 않고 있다. 따라서 본 논문에서는 위와 같은 위험요소의 연관성을 정량적으로 분석하기 위하여 연구를 수행하고자 한다.

3. 본론 및 사례연구

본 장에서는 소프트웨어 프로세스 개선을 위하여 위험요소를 식별하고, 이에 대한 추이분석으로 더 큰 문제로 전이되는 것을 예방하기 위한 추이분석 방법을 제시한다. 또한 위험요소로 전이되는 각 단계별 기준을 제시하여 관리자로 하여금 이를 예측 및 조치를 취할 수 있는 기준을 제시하고자 한다.

3.1 소프트웨어 프로세스 개선을 위한 위험요소 식별

특정 도메인의 정보를 이용하기 위해서는 영역 전문가의 도움을 받아서 수동적으로 정보를 찾게 된다. 그리고 영역 전문가에게 정보를 찾기 위하여 도움을 요청하게 되고, 요청에 대한 내용은 게시판을 이용하여 질문을 하게 된다. 따라서 그 결과를 응답받기 위해서는 많은 시간을 기다려야 한다. 또한 도메인 전문가도 요청되는 정보를 찾기 위하여 수동적인 절차에 의하여 정보를 찾아서 제공하게 된다. 따라서 수동적인 절차를 자동화 및 웹에서 제공을 하고 필요한 정보를 검색할 수 있게 한다. 그러므로 검색을 위하여 사용자가 사용하기 쉬운 인터페이스를 제공해야 하며, 재사용성을 위하여 활용하고자 하는 분야의 자료를 통합 및 관리하는 방법이 제공되어야 한다. 또한 소프트웨어 프로세스 개선을 하기 위해서는 개선 항목을 지정하여 지속적으로 항목의 변경 및 추가 사항이 존재하는가를 검사해야 한다. 이를 통하여 정확하고 신속한 소프트웨어 프로세스 개선을 수행할 수 있게 된다.

일정관리를 수행하지 않았을 경우에 발생하는 문제점을 예를 들어서 설명하면 다음과 같은 문제점을 발생시킬 수 있다.

- 프로젝트 수행 시, 잠재되어 있는 문제점에 의한 일정 지연 발생 및 예측과 대비 미흡, 추가비용 발생, 전체적인 품질 저하
- 개선되지 않은 프로세스에 의하여 개발되는 소프트웨어에 문제점이 항상 내재됨
- 결함 발생 시, 이를 해결하기 위한 방법 부재 및 원인 파악의 난해성
- 유사한 프로젝트 수행 시, 동일한 내용의 결함 발생

〈표 1〉 프로젝트 계획 및 위험 요소 식별 항목의 내용

내용	중요도
1. 개발환경 구축	Low
2. 시스템의 필요성, 목표, 제약사항 및 요구되는 기술정의	Very High
3. 프로세스 및 생성되는 산출물들 간의 연관성 식별	High
4. 업무 분해도 및 산정치, 기반구조 요소를 근거로 하여 프로젝트 일정 수립	Nominal
5. 프로젝트 계획이 실무자에게 인식되어 활용할 수 잇는 체계 마련	High

따라서 위와 같은 내용을 대비하기 위하여 opportunity tree를 사용하고자 한다.

위험요소 식별을 위하여 많은 방법론과 도구들이 사용되고 있다. 이와 같은 연구를 논문으로서 제시하였다[18, 19]. opportunity tree에서는 각 요소의 연관성의 중요도를 측정하여 위험요소를 정량적으로 제시하였다. opportunity tree의 예는 다음과 같다. 각 중요도는 통계 수치자료를 활용하였으며, 설문서 빈도에 따라서 5개 항목을 많은 빈도부터 Very High, High, Nominal, Low, Very Low로 결정지었다.

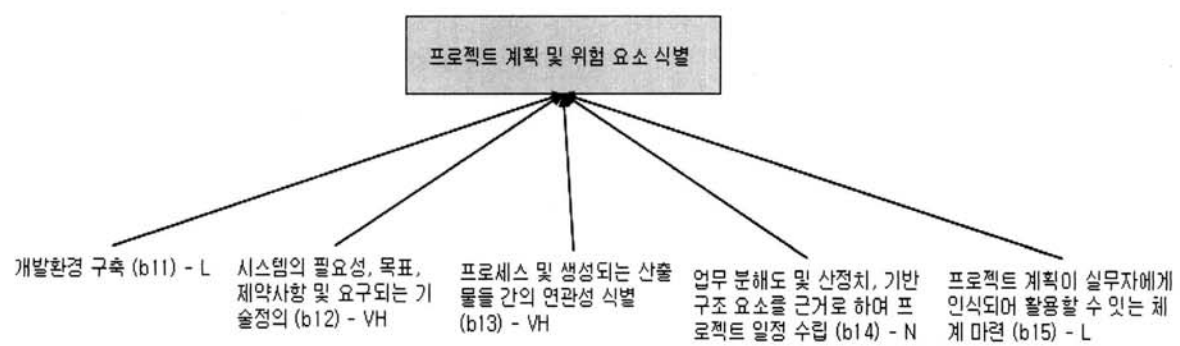
프로젝트 계획 및 위험 요소 식별에 관한 의사결정 구조도와 등급화된 항목은 (그림 1)과 같다.

항목간 비교 메트릭스는 <표 2>와 같다.

프로젝트 계획 및 위험 요소 식별 항목 간 비교 메트릭스는 b11, b12, b13, b14, b15 열을 기준으로 다른 항목과 중요도를 비교하게 된다. 따라서 b11과 b12는 b12가 4배 많게 분석이 되었다. 그 이유는 b11은 L 이고, b12는 VH이므로 두 항목간의 등급차이가 3이므로 중요도 차이 비율은 4배가 된다.

프로젝트 계획 및 위험 요소 식별을 수행하기 위한 목표 중 어떤 목표가 중요한가를 산출하여야 한다. 따라서 이와 같은 중요한 목표를 산출하기 위하여 기하 평균을 이용하였다. 프로젝트 계획 및 위험 요소 식별 항목의 목표 중요도는 <표 3>과 같이 산출되었다.

b12(시스템의 필요성, 목표, 제약사항 및 요구되는 기술정의) 하위 목표가 프로젝트 계획 및 위험 요소 식별에서 가장 중요한 달성 목표로 분석되었다. 따라서 b12(시스템의 필



(그림 1) 프로젝트 계획 및 위험 요소 식별 원인 등급 구조도

〈표 2〉 프로젝트 계획 및 위험 요소 식별 항목간 비교 메트릭스

	b11	b12	b13	b14	b15
b11	1	1/4	1/3	1/3	1/3
b12	4	1	2	2	2
b13	3	1/3	1	1	1
b14	3	1/3	1	1	1
b15	3	1/3	1	1	1

〈표 3〉 프로젝트 계획 및 위험 요소 식별의 하위 목표 중요도 산출표

항목	산출값
b11	$(1 \times 1/4 \times 1/3 \times 1/3 \times 1/3)^{1/5} = (1/108)^{/5}$ 0.39
b12	$(4 \times 1 \times 2 \times 2 \times 2)^{1/3} = 32^{1/5} = 2$
b13	$(3 \times 1/3 \times 1 \times 1 \times 1)^{1/5} = (1)^{1/5} = 1$
b14	$(3 \times 1/3 \times 1 \times 1 \times 1)^{1/5} = (1)^{1/5} = 1$
b15	$(3 \times 1/3 \times 1 \times 1 \times 1)^{1/5} = (1)^{1/5} = 1$

〈표 4〉 프로젝트 계획 및 위험 요소 식별 목표를 위한 해결책

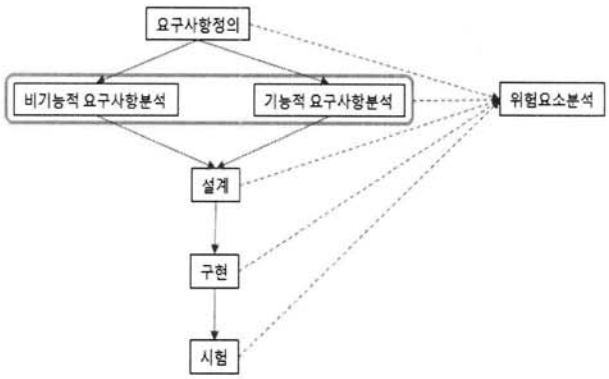
결함 내용	해결책
1. 개발환경 구축	1. 기능적 요구사항과 비기능적 요구사항 분석 2. 개발에 적절한 인력 마련 3. 프로젝트 수행에 필요한 인프라 구축
2. 시스템의 필요성, 목표, 제약사항 및 요구되는 기술정의	1. 프로젝트의 목표 분석 2. 프로젝트의 위험 요소 분석
3. 프로세스 및 생성되는 산출물들 간의 연관성 식별	1. 프로세스별 산출물 파악 2. 업무의 선후관계 분석 3. 작업의 우선순위 정의
4. 업무 분해도 및 산정치, 기반구조 요소를 근거로 하여 프로젝트 일정 수립	1. 프로젝트 일정 산정의 근거 정의 2. 프로젝트 일정 산정의 인프라 구축
5. 프로젝트 계획이 실무자에게 인식되어 활용할 수 있는 체계 마련	1. 프로젝트 계획의 주기적인 확인 2. 일정단위별 진척도 측정

요성, 목표, 제약사항 및 요구되는 기술정의)가 프로젝트 계획 및 위험 요소 식별을 달성하기 위하여 우선순위가 가장 높고, 달성되어야 할 하위목표로 나타났다. 그리고 b11, b12, b13, b14, b15의 목표를 달성하기 위해서는 <표 4>과 같은 해결책을 제시할 수 있다.

위에서 제시한 방법을 토대로 하여 각 주제별로 위험요소를 식별하고자 한다.

3.2 위험요소 분석을 위한 추이분석 방법론

개발 단계 전반에 걸쳐서 위험요소 분석을 위하여 이를 추출하기 위한 노력이 계속되고 있다. 따라서 위험요소 분석단계는 프로젝트나 개발의 전반적인 생명주기에 연관되어 있으며, 좋은 품질과 신뢰성을 확보하기 위하여 위험요소의 관리가 필수적이다. 다음 그림은 위험요소가 생명주기에 전반적으로 연관되어 있으며, 위험요소 관리의 필요성을 나타내기 위한 것이다.



(그림 2) 생명주기와 위험요소 분석단계의 연관성

(그림 2)에서와 같이 위험요소 분석은 개발 초기의 요구사항정의에서부터 시험단계까지 생명주기 전반에 필요한 필수 사항이다. 위험요소 분석을 채택하여 개발에 적용을 함으로서, 생산품이나 프로세스의 신뢰성과 위험요소로 전이되는 것을 예방할 수 있게 된다.

위험요소는 요구사항정의 단계에서부터 발생하여 구현 단계에 까지 영향을 미치게 된다. 요구사항에서 위험요소를 내포하고 분석된 사항들은 설계단계에 영향을 미치게 되어 위험요소를 내포한 상태로 설계가 이루어지게 된다. 이와 같은 상황에서 위험요소가 발생되거나 발견되지 않는다면, 개발자와 관리자는 아무 문제가 없는 것으로 판단하고 위험요소가 있는 상태를 그대로 구현 단계로 전달하게 된다. 구현 단계에서는 프로그래머의 성숙도 부족으로 인식하고 근본적인 위험요소의 제거가 이루어지지 않고, 발생된 위험요소만 수정해서 생산품이 출시된다. 이와 같은 상황은 위험요소를 포함하여 생산품이 생산이 되므로, 후에 리콜이나 많은 문제를 발생시켜서 기업이나 프로젝트에 상당히 악영향을 끼치게 된다. 따라서 프로젝트 전반에 걸친 위험요소의 관리는 필수적인 요소가 되어야 한다.

3.3 위험요소 전이단계

추이분석을 위하여 선행되어야 할 사항은 위험요소로 전이되는 단계를 정의하는 것이다.

전이되는 단계를 정의함으로서, 위험요소로 전이되는 상황을 판단할 수 있으며, 이를 활용하여 위험요소로 전이되는 것을 예방하고 대비할 수 있게 된다. 또한 관리자나 개발자는 현재 자신과 연관된 위험요소를 파악하여 연관된 요소들을 관리하여 위험요소의 확산이 되지 않도록 할 수 있게 된다.

본 논문에서는 다음과 <표 5>와 같은 전이 단계를 제시한다.

* 이슈단계는 요구사항과 연관된 모든 사항을 이슈사항으로 분석을 하게 된다. 따라서 요구사항을 시스템으로 기능화하기 위하여 필요한 세부기능의 사항도 포함하게 된다. 이를 위해서는 영역분석을 수행하면서

〈표 5〉 위험요소 식별을 위한 전이단계

위험 활동내용 전이단계	상세 내용
1. 이슈단계	요구사항과 연관된 모든 사항을 이슈사항으로 분석함
2. 이슈사항의 연결그룹 분석단계	분석된 이슈사항을 요구사항과 연관지어 카테고리별로 그룹핑함
3. 문제요소 분석단계	분석된 이슈 사항에서 요구사항 실현에 영향을 미치는 요소
4. 문제요소의 연관성 분석단계	분석된 문제요소에서 서로의 연관성 분석
5. 결함요소 분석단계	문제요소가 실제 구현되는 기능상에서 실행을 저해하는 요소
6. 결함요소의 인과관계 분석단계	결함요소간의 인과관계 분석
7. 위험요소 분석단계	일정, 비용, 품질에 영향을 미치는 요소

요구사항 분석이 병행되어야 한다. 또한 각 이슈사항을 비 기능적 및 기능적으로 분류를 해서 추출을 하고 이를 활용하여 위험요소가 시스템에 연관된 사항인지 고객의 요구사항에 연관된 사항인지를 구분해야 한다. 이와 같은 구분은 하드웨어와 관련된 위험요소인지 소프트웨어와 연관된 위험요소인지를 구분하는 중요한 자료로 활용되게 된다.

* 이슈사항의 연결그룹 분석단계는 요구사항 분석과 연관되어서 기능별로 같은 범주에 속하는 것을 구별하는 것이다. 이와 같은 분석과정은 기능에 문제가 발생한 경우에, 문제가 발생한 부분을 손쉽게 찾아서 유지 및 보수를 원활히 할 수 있게 된다. 따라서 기능의 범주를 추출하기 위하여 기능별로 그룹화를 수행해야 하며, 이를 토대로 추가되는 기능에 대하여서는 주기적으로 범주와 기능이 추가되었는지를 확인해야 한다.
* 문제요소 분석단계는 요구사항을 시스템으로 기능화 하여 구현하는 과정에서 구현에 중요한 요인이 무엇인지를 분석하고 추출하는 과정이다. 이와 같은 분석과정은 기능을 완성하는 과정에서 중요한 기능이 아님에도 프로젝트나 개발에 치명적인 요소가 발생하는 경우, 이를 대비하기 위한 단계이다.
* 문제요소의 연관성 분석단계는 요구사항을 시스템으로 기능화 하여 구현하는 과정에서 발생하는 영향에 대하여 분석을 하는 단계이다. 시스템에 구현하는 과정에서 문제가 발생하는 경우, 다른 기능에서 그 영향을 받는다면 위험요소를 그대로 다른 기능에 전이하게 된다. 이와 같은 상황을 예방하기 위하여 문제요소 분석단계를 거쳐 예방을 하기 위함이다.
* 결함요소 분석단계는 각 단계에서 가장 중요시하고 노력승수를 많이 투여해야 하는 단계이다. 그 이유는 이전단계에서는 위험요소로 전이되는 것이 명확하게 나타나지 않게 된다. 위험요소로 전이되는 것은 결함요소로 발생하면서 실제적인 기능상의 오류나 문제를 야기하게 된다. 따라서 결함요소 분석단계에서부터는 좀 더 명확하고 상세한 결함 관리가 수반되어야 한다. 결함관리를 위하여 결함종류를 분석하고 결함의 영향 정도를 파악하여 주 기능을 저해하는 결함인지 주 기능과 연관성이 없는 결함인지를 파악하여 결함정도가 미치는 영향을 정량화하여 관리하여야 한다.
* 결함요소의 인과관계 분석단계는 실제로 위험요소 전이가 되는 결함간의 연관성을 분석하는 단계이다. 결함의 연관성을 분석하여 발생하는 결함의 추적성과 수정을 동시에 수행할 수 있게 된다.
* 위험요소 분석단계는 발생된 요소가 일정을 지연시키고, 제한된 개발비용이나 프로젝트 비용을 초과하며, 초기 개발 목적에서 요구되었던 품질에 미치지 못하게 되는 경우가 발생하지 않도록 하기 위한 단계이다.

위험요소 식별을 위한 단계는 위와 같이 제시하였으며, 이를 토대로 하여 사례연구를 수행하였다.

3.4 위험항목 분석을 위한 정량적인 측정방법

제시된 각 단계는 위험요소를 관리하기 위하여 제시하였다. 또한 단계별로 분석하여 관리하기 위하여 세분화 하였다. 각 전이단계에서 결함으로 전이되는 요소를 측정하기 위하여 수학적인 접근 방법을 사용하였다.

$$RD(t,p) = IF(t,p) + \prod_{i=1}^{7} C(i)$$

RD : 위험정도 산출
t : 형태
p : 단계
IF : 위험요소별 영향
C : 결함 개수 생성 함수
i : 각 단계의 색인

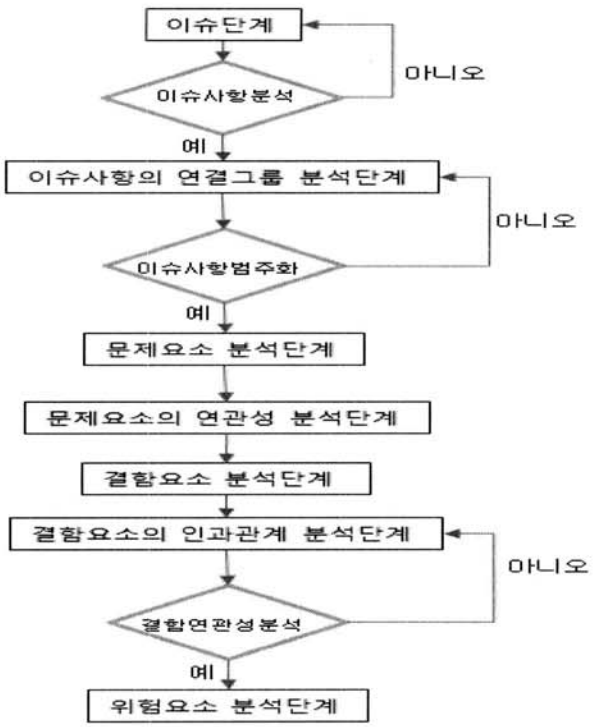
위험요소로 전이 되는 것을 정량적으로 측정하기 위하여 위와 같은 식을 제시하였다.

수식에서는 위험요소별 측정을 단계별로 산출하게 된다. 이와 같은 이유는 각 요소별로 위험요소로 전이되는 것을 측정하기 위함이다. 수식을 구성하는 각 요소를 설명하면 다음과 같다.

t는 위험요소 대상의 형태를 구분하기 위함이다. t에 의하여 위험요소가 어떤 형태인지 구분하고 분류할 수 있다. 형태는 영역에 따라서 형태가 달라지므로 영역과 개발하는 프로젝트의 형태에 따라서 달라진다.

p는 위험요소가 전이되는 단계를 나타낸 것이다. 따라서 현재 요소가 어디까지 전이되었는지 단계를 제시하여 어느 정도까지 전이가 되었는지 인지할 수 있다.

IF는 각 요소마다 프로젝트에 영향을 주는 정도이다. 이는 형태와 단계에 연관이 되며, 프로젝트와 영역의 특성에



(그림 3) 위험요소 분석단계의 순서도

따라서 주 요인이 될 수도 있고 그 반대가 될 수도 있다. 예를 들면, 시간우선으로 결과물이 나와야 하는 프로젝트는 비용보다는 시간이 우선되어야 하므로 시간에 연관된 요소들의 가중치가 높아지게 된다.

C는 요서별로 단계에 영향정도를 정량적인 방법으로 접근하기 방법으로 결함의 개수를 산출하기 위한 함수이다. 함수의 산출방법은 결함 개수 산출을 위한 연구에서 제시하였다[18, 19]. 각 요소별로 중요도를 측정하여 단계에서 발생하는 요소 중에 결함이 되어 영향을 줄 정도를 기하평균을 활용하여 산출하는 방법이다. 이와 같은 방법은 3.1절에서 제시하였다. 기하평균을 사용한 이유는 각 요소의 중요도를 측정하기 위하여 전체 결함의 개수 중 차지하는 비중을 측정하기 위해서이다.

위의 수식을 기반으로 각 요소의 위험정도가 어느 단계까지 전이되었는지 정량적으로 제시할 수 있다.

제시된 수식으로 정량적인 산출을 하기 위하여 IF와 C의 산출방법을 제시하고자 한다.

IF의 수준은 상, 중, 하와 각 단계별로 상중하를 다시 세분화 하였다. IF의 수준별 가중치 값은 다음과 같다.

각 단계에서 상으로 갈수록 위험정도에 영향을 많이 미치는 경우를 뜻한다. 여기서 상, 중, 하의 기준은 각 요소가 다른 요소에 영향을 주는 정도로 측정을 한다. 따라서 IF의 영향 정도를 측정할 경우에는 요구사항분석단계에서 다른 요구사항에 영향을 주는 연관성 분석을 수행하여 영향 정도를 파악하게 된다. <표 6>에서의 적용 범위는 "(다른 요구사항에 전이되는 요구사항개수/전체요구사항 × 100 × IF 가중치)"에 의하여 산출되게 된다. 이와 같은 내용은 발생되는 개발단계(p)와 형태에 의하여 영향정도를 측정하게 된다.

C는 결함 제거률을 통하여 산출하도록 한다. 결함제거률은

〈표 6〉 위험요소 영향도 산출기준

IF기준	하부기준	적용 범위
상	상(0.9)	90%이상
	중(0.8)	89~80%
	하(0.7	79~70%
중	상(0.6)	69~60%
	중(0.5)	59~50%
	하(0.4)	49~40%
하	상(0.3)	39~30%
	중(0.1)	29~20%
	하(0.1)	19%이하

"|(발생된 결함개수 - 제거된 결함개수)| / 전체 결함개수 × 100"에 의하여 산출한다. 결과 값은 개수에 의한 값이므로 절대 값으로 산출한다. 이는 다음의 경우의 수를 갖게 된다.

1) 발생된 결함개수 > 제거된 결함개수(발생된 결함을 모두 제거하지 못한 경우)
2) 발생된 결함개수 < 제거된 결함개수(이전 단계에서 발견하지 못한 결함을 이후 진행하는 단계에서 결함이 발생하는 경우이며, 이는 이전단계에서 예측과 분석을 못한 경우)
3) 발생된 결함개수 = 제거된 결함개수(발생된 결함을 완전히 제거하여 이후 진행되는 단계에서 발생하지 않는 경우이며, 연산을 위하여 1을 부여함)

따라서 위에서 제시한 연산 과정에 의하여 위험정도를 분석하고자 한다.

3.5 위험항목 관리 방법

위험항목을 관리하기 위하여 위험활동내용 전이단계를 기반으로 관리방안을 제시한다. 관리 방법은 (그림 3)과 같은 순서도에 의하여 관리된다.

위험항목 전이단계는 3.2절에서 제시하였다. 순서도에서 세분화된 단계를 측정하는 과정에서 위험요소가 어느 수준까지 전이되었는지 진척도를 측정할 수 있게 된다. 또한 각 단계에서 필수적으로 측정해야 할 단계를 이슈사항의 범주화와 결함연관성 분석에 있다. 그 이유는 발생된 위험항목의 원인 및 예방을 찾기 위하여 추적성을 제공하는 것이다. 추적성을 제공하기 위하여 이슈사항의 범주화를 수행하게 된다. 수행된 범주화는 위험요소 발생에서 해당 기능을 찾을 수 있게 된다. 또한 결함연관성 분석은 결함간의 영향도를 측정하여 발생된 결함과 해결하는 것이 아니라, 발생된 결함의 영향을 받은 모든 사항을 관리하는 것이다. 이와 같은 목적을 달성하기 위해서는 이슈사항의 범주화가 선행이 되어야 하며, 이를 기반으로 결함연관성 분석이 수행되어야 한다.

4. 사례연구

3장에서 제시된 위험정도 산출 식에 의하여 각 단계의 정량적인 결과를 얻을 수 있다. 각 단계는 3.3절에서 제시된 단

계를 기초로 하고 있다. 적용 분야는 업무의 IT 전산화 작업 과제이며, 개발 생명주기는 점층적인 기법을 적용하였다. 점층적인 방법에 의하여 각 단계는 네 번의 생명주기를 갖고 있으며, 정량적인 산출을 하기 위하여 네 번의 생명주기에서 위험정도가 가장 많이 발생한 생명주기의 식을 제시하였다.

산출된 단계에서 이슈단계는 세 번째 점층적인 방법에서 최대치가 산출되었으며, 문제요소분석단계와 결함요소분석단계는 전체 생명주기에서 일정한 위험정도를 나타내었다.

전체 단계에서 가장 큰 영향을 주는 이슈단계와 문제분석단계 및 결함요소분석단계의 위험정도 산출내역은 다음과 같다. 연산은 소수점 첫째 자리에서 반올림하였다.

1) 이슈 1단계
$\{0.2 \times (410 - 110) \times 100 / 650\} + (20 \times 100 / 150) \fallingdotseq 22$

2) 이슈 2단계
$\{0.5 \times (410 - 220) \times 100 / 650\} + (20 \times 100 / 150) \fallingdotseq 28$

3) 이슈 3단계
$\{0.9 \times (410 - 350) \times 100 / 650\} + (134 \times 100 / 150) \fallingdotseq 97$

4) 이슈 1단계
$\{0.2 \times (30 - 10) \times 100 / 40\} + (4 \times 100 / 30) \fallingdotseq 23$

이슈 1~3단계까지는 이전의 분석 및 요구사항 관리가 잘 이루어지지 않아서 3단계에서 위험정도가 절정을 이루고 있다. 4단계의 경우, 이전 단계에서 발견 못한 새로운 결함이 발생한 경우이다.

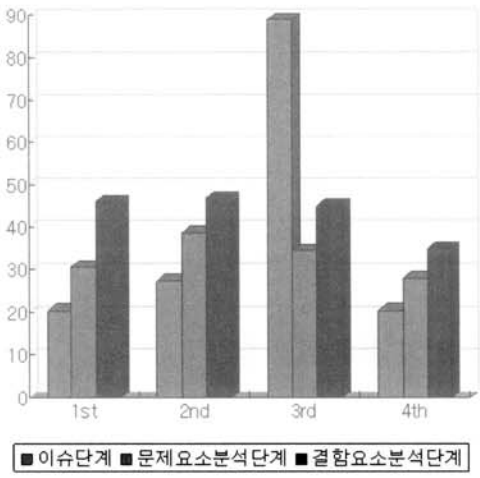
본 논문에서 위험항목 관리방법을 적용하기 위하여 제조업과 연관된 회사의 프로젝트에 적용을 하였다. 적용 프로젝트는 점층적 방법의 생명주기를 사용하였고, 위험항목을 식별하기 위하여 네 번의 주기로 생명주기를 적용하였다. 각 단계요소의 결함 내용은 기업의 정보이므로 세밀한 사항은 제공하지 않고 결함개수만 제공하여 분석을 수행하였다.

각 주기마다 위험요소를 제거하기 위한 활동을 수행하였으며, 각 단계를 전이단계의 중요한 마일스톤이 되는 3개의 단계를 기준으로 위험항목의 결함 개수를 정의하였다.

네 번의 생명주기에 대한 결함 개수에 대한 내용은 다음과 같다.

(그림 4)에서와 같이 생명주기는 네 번의 점층적 방법을 수행하였다. 네 번의 주기에서 이슈단계와 문제분석단계 및 결함요소분석단계에 대한 각 단계의 개수를 추출하였다.

첫 번째 생명주기에서는 이슈단계에 대한 문제점이 초기 단계에서 발생빈도가 적게 되었다. 그러나 문제분석단계와



(그림 4) 위험요소 관리의 통계

결함요소분석단계에서는 위험요소를 그대로 가지고 프로젝트를 진행했기 때문에 결함의 영향을 그대로 받은 상태로 문제점들이 발생하고 있다. 이와 같은 결과는 요구사항에 의한 이슈정의가 미흡하고, 개발초기이므로 요구사항의 변화가 빈번이 발생하여 문제분석단계와 결함요소분석단계에 영향이 그대로 전이된 상태이다. 따라서 이슈단계에서는 오히려 결함이 덜 발생한 반면에 문제분석단계와 결함요소분석단계에서는 결함이 많이 발생하게 되었다.

두 번째 생명주기에서는 결함분석단계의 결함 개수가 이전과 동일한 수준이며, 이슈단계와 문제분석단계의 결함이 증가하였다. 첫 번째 생명주기에 비하여 결함 개수는 증가하였지만, 위험요소 분석관리를 통하여 결함을 더 찾아낸 것으로서 향후 결함발생의 원인 및 예방의 기초 자료로 활용이 가능하게 된다. 따라서 위험요소 관리를 위한 과도기에 해당되는 단계가 된다.

세 번째 생명주기에서는 이전 생명주기 적용 단계와 특이사항으로 이슈단계의 결함이 증가하였다. 세 번의 생명주기를 점층적으로 적용하는 단계에서 위험요소가 추출되고 그 원인의 대부분이 요구사항 분석단계에서 오류 및 기능 분석 수행에 문제점 때문에 이슈단계의 결함이 증가하였다. 이슈단계의 결함이 증가한 다른 원인은 앞에서 적용한 두 단계의 점층적인 생명주기 단계에서 발생한 문제분석단계 및 결함분석단계의 결함의 원인이 이슈단계가 원인이 되어서 결함개수가 증가하게 되었다. 따라서 본 단계의 결과를 통하여 초기에 요구사항의 올바르고 정확한 분석이 전체 프로젝트 개발에 많은 영향을 미치게 된다는 것을 알 수 있게 된다. 또한 초기단계 분석의 중요성을 인식시켜주는 자료가

〈표 7〉 위험요소 산출 자료표

단계	IF 가중치	발생된 결함개수	제거된 결함개수	전체 결함개수	전체요구사항	다른 요구사항에 전이되는 요구사항 개수
1	0.2	410	110	650	150	20
2	0.5	410	220	650	150	20
3	0.9	410	350	650	150	134
4	0.2	30	10	40	30	4

되기도 한다.

네 번째 생명주기에서는 이전 생명주기와 비교하여 이슈 단계, 문제분석단계, 결함분석단계의 결함개수가 줄어드는 것을 확인할 수 있다. 그 이유는 이전 세 번의 점층적 생명주기 적용단계를 거치면서 각 단계의 기능의 연관성을 분석하여 추적성을 제공하게 되었다. 이는 결함이 발생을 해도 해당 기능을 찾아 원인을 제거하고, 그 영향이 파급되는 것을 막을 수 있기 때문에 결함개수가 줄어든 것이다.

5. 결론 및 향후 연구 방향

본 논문에서는 결함이 전이되어서 위험요소로 전이되는 것을 예방하고 원인을 찾아 결함을 해결하기 위하여 위험요소 전이 단계를 제시하였다. 또한 위험요소 관리 방법을 적용하여 각 전이 단계의 결함의 개수가 줄어드는 것을 확인할 수 있다. 위험요소로 전이되는 것을 예방하여 전체 개발에서 일정이 지연되거나 품질이 저해되거나 비용이 추가되는 상황을 예방할 수 있게 되게 된다. 이는 프로젝트 개발 시 신뢰성 있는 개발을 통하여 고객과 개발자 및 관리자에게 만족도를 높일 수 있게 된다.

향후 연구로는 현재 제시된 전이 단계의 측정을 정량적으로 산출하기 위한 연구가 필요하다. 이를 위해서는 정량화 요소를 선정하고, 개발 영역에 따른 각 요소에 대한 가중치 결정해야 한다. 또한 결정된 요소의 가중치를 여러 프로젝트에 적용을 하여서 신뢰성을 높이는 연구를 계획하고 있다.

참 고 문 헌

[1] 윤청, 성공적인 소프트웨어 개발 방법론, 생능출판사, 1999. 08.

[2] 허원실, 시스템 분석과 설계, 한빛미디어, 2006.05.

[3] Software process improvement forum, KASPA SPI-7, 2002.

[4] 신경애, "결함중요도 단계를 고려한 소프트웨어 신뢰도 성장 모델에 관한 연구," 컴퓨터 산업교육기술 학회 논문지, Vol.03 No.07 pp.0837-0844, 2000.

[5] Robert h. dunn, "Software defect removal," Mcgraw-hill, 1984.

[6] Ram chillarrege, Kothanda r. prasad, "Test and development process retrospective a case study using ODC triggers," IEEE computer society, 2002.

[7] McCall, P.K. richards and G.F. walters, "Factors in software quality," Vol.1, 2 and 3. Springfield VA, AD/A-049-014/015/055, 1997.

[8] Wohlin, Runeson, "Defect content estimations from review data," Proceedings international conference on software engineering ICSE pp.400-409, 1998.

[9] Gaffney, John, "Some models for software defect analysis," Lockheed martin, 1996.

[10] L.hatton, "Is modularization always good idea," Information and software technology, Vol.38, pp.719-721. 1996.

[11] B. compton and C. withrow, "Prediction and control of ada software defects," J. systems and software, Vol.12, pp.199-207, 1990.

[12] Roger s. pressman "Software engineering" Mcgraw-hill international edition, 1997.

[13] Tim kasse, "Actin focused assessment for software process improvement, Artech house, 2002.

[14] Victor basili, G. caldiera and D. rombach, "The goal question metric approach," Wiley, 1994.

[15] Victor basili, "The experience factory and its relationship to other improvement paradigms," Lecture notes in computer science, 1993.

[16] Paulish, and Carleton, "Case studies of software process improvement measurement," Computer, Vol.27, No.9, 1994.

[17] Roger s. pressman, "Software engineering: A practitioner's approach," Mcrwa-hill international editions, 1997.

[18] 이은서, 이경환, "소프트웨어 결함 분석을 위한 트리거 설계," 한국정보처리학회 논문지, 2003.

[19] 이은서, 이경환, 소프트웨어 결함 처리를 위한 Opportunity Tree 및 알고리즘 설계, 제11-D권 제4호 PP873, 정보처리학회논문지D, 2003. 08.

이 은 서

e-mail : eslee@andong.ac.kr

2001~현 재 ISO/IEC 15504 국제 심사위원

2004년 중앙대학교 컴퓨터공학과(박사)

2004년~현 재 임베디드 산업협회 전문위원

2004년~현 재 한국정보통신기술협회위원

2005년~2007년 숭실대학교 정보미디어기술 연구소 연구교수

2008년 3월~현 재 안동대학교 컴퓨터공학과 조교수

관심분야 : CBD, Formal method, Quality model, SPI(Defect Analysis)