요구관리산출물과 위험관리산출물의 연계 방안

한소연° 김주영 류성열
숭실대학교 대학원 컴퓨터학과
trony203°@ssu.ac.kr, gogumacake@hanafos.com, syrhew@ssu.ac.kr

The Mapping Method between Requirement Tracebility Table and Risk Management List

SoYeon Han° JuYoung Kim, SungYul Rhew
Soongsill Univ. grad. Dept of computing

요 약

소프트웨어 개발 프로젝트의 위험관리는 프로젝트의 성패를 좌우하는 매우 중요한 영역이다. 그러나 위험관리의 주체가 관리자의 역할을 가진 사람들에 의해 진행되고, 방법론이나 지침에서 제시하고 있는 위험관리 또한 여러 활동들과 산출물이 실제 위험관리정보를 필요로 하는 개발자들에게는 적절하게 연결이 되지 않고 있어서 개발자들이 필요한 정보를 파악하기에 문제점이 있다. 이에 따라 본 연구에서는 개발자들을 위한 위험관리방법으로, 요구사항추적테이블이라는 양식과 위험관리리스트간의 연결방안을 제안한다. 요구사항추적테이블은 소프트웨어 개발 생명주기 전체 과정에서 활용하고 있으며 개발자들이 사용하기 쉬운 양식이므로 요구사항추적테이블과 위험관리리스트를 연결하는 것으로 개발자들로 하여금 위험에 대한 정보를 파악하기 용이하게 하여 설계와 구현 시 관련 위험에 대해 예측이 가능하며 대처방안을 사전에 고려할 수 있다.

1. 서 론

프로젝트의 위험은 발생할 경우, 최소 하나의 프로젝트 목표에-시간, 원가, 범위, 품질 등- 대해 영향을 주는 불확실한 사건이나 조건을 의미한다[1]. 소프트웨어 개발 프로젝트를 성공적으로 이끄는 방법은 개발에서의 실패를 최소화하기 위하여 발생할 수 있는 위험을 사전에 예측하여 관리할 수 있어야 한다. 따라서 소프트웨어 개발 프로젝트에서 위험관리는 매우 중요한 영역이다. 그러나 현재 방법론이나 소프트웨어 개발 프로세스 표준들을 보면 사실상 위험관리는 관리자의 역할을 가진 사람들에 의해서 진행되어 위험에 대한 분석 정보가 실제 위험을 사전에 이해하여 분석 설계에 고려할 수 있어야 하는 개발자들에게는 제대로 연결되지 않고 있다. 개발자가 위험관리를 통해 알 수 있는 정보들을 개발자 수준에서 사용하고 활용할 수 있다면, 개발 과정 중에 발생하는 위험을 미리 대비할 수 있으며 이를 적극적으로 해결하기 위한 노력을 기울일 수 있다.

그렇다면 개발 표준 혹은 방법론 등에서 제시하고 있는 위험관리 프로세스의 정보들을 어떻게 개발자들에게 알려줄 것이며, 어떻게 활용하게 할 수 있을 것인가? 또한 이것을 어떠한 산출물 혹은 어떤 액티비티들을 통해 보여줄 수 있을 것인가?

본 논문은 개발자들을 위한 위험관리 방법으로 산출물에 대한 연구로 시작된다. 새로운 산출물을 개발하기에 앞서, 기존에 사용되고 있던 산출물들을 개선하도록 한다. 개선 방법으로 개발 과정 시 포괄적으로 사용되는 산출물과 위험관리 산출물을 연결하는 방안을 제시하였다.

또한 개발 프로세스의 전체 과정에서도 특히 위험 발생 빈도가 높은 요구사항 상세 분석단계에서 위험을 식별하고 함께 관리하는 것이 효과적이므로 본 논문에서는 위험의 범위를 요구사항분석 단계에서 식별할 수 있는 위험요소들로 범위를 제한하였다[2][3]. 요구사항분석 단계에서 개발자들이 쓰는 양식은 요구사항 추적 테이블로 사용과 관리가 용이하고, 개발프로세스의 전 단계에서 활용할 수 있다는 점 때문에 가장 많이 사용된다. 이에 따라 요구사항 추적테이블과 위험관리리스트를 연결하여 함께 관리하면 개발자들은 전체 개발 생명주기에 걸쳐 자신들의 추적테이블에 표현된 요구사항들과 발생할 가능성이 있는 위험에 대해 사전에 대비할 수 있다.

2. 관련 연구

2.1 위험요소

2.1.1 DIR(U.S Texas Department of Information Resources)의 소프트웨어 프로젝트 위험요소

미 텍사스 주 연구기관인 DIR에서는 Generic Software Project Risk Factors라는 문서를 통해 소프트웨어 프로젝트에서 발생할 수 있는 위험요소들을 테이블 형식으로 정리하였다. 총 82개의 위험요소가 14개의 카테고리로 구분되어 있고 각 위험요소들이 어떤 상황에서 위험도가 높고 낮은지 판단할 수 있는 예시를 테이블 내에 상세히 기술한다. 위험요소마다 높음, 중간, 낮음을 표시하여 각각의 개수를 세어 해당 프로젝트의 위험요소의 수를 위험이 높은 것이 몇 개 중간인 것이 몇 개, 낮은 것이 몇 개 인지를 결과로 보여준다[4].

2.1.2 NFR 그래프를 이용한 위험분석에서의 위험식별

위험분석을 하기 위하여 위험요소들을 평가하기 위한 목적으로 요소들을 식별한다. 위험 범주를 위험 분할구조(RBS) - 기술, 외부, 조직, 관리-에 따라 분류하고, 일정지연의 원인이 되는 위험요소들을 기술한다[5]. 그러나 이 논문에서 식별한 요소들은 일정지연을 일으키는 위험요소들만으로 한정되어 있다.

2.1.3 CMU의 위험관리 기술문서에서의 위험분류 및 위험요소

CMU에서 제시한 위험관리 기술문서에서는 소프트웨어 개발에서의 위험을 클래스와 엘리먼트, 애트리뷰트의 상하 계층구조로 표현하고 있다.

A. *Product Engineering*
1. Requirements
   a. Stability
   b. Completeness
   c. Clarity
   d. Validity
   e. Feasibility
   f. Precedent
   g. Scale
2. Design
   a. Functionality
   b. Difficulty
   c. Interfaces
   d. Performance
   e. Testability
   f. Hardware Constraints
   g. Non-Developmental Software
3. Code and Unit Test
   a. Feasibility
   b. Testing
   c. Coding/Implementation
4. Integration and Test
   a. Environment
   b. Product
   c. System
5. Engineering Specialties
   a. Maintainability
   b. Reliability
   c. Safety
   d. Security
   e. Human Factors
   f. Specifications

B. *Development Environment*
1. Development Process
   a. Formality
   b. Suitability
   c. Process Control
   d. Familiarity
   e. Product Control
2. Development System
   a. Capacity
   b. Suitability
   c. Usability
   d. Familiarity
   e. Reliability
   f. System Support
   g. Deliverability
3. Management Process
   a. Planning
   b. Project Organization*
   c. Management Experience
   d. Program Interfaces
4. Management Methods
   a. Monitoring
   b. Personnel Management*
   c. Quality Assurance
   d. Configuration Management
5. Work Environment
   a. Quality Attitude*
   b. Cooperation*
   c. Communication
   d. Morale*

C. *Program Constraints*
1. Resources
   a. Schedule
   b. Staff
   c. Budget
   d. Facilities
2. Contract
   a. Type of Contract*
   b. Restrictions
   c. Dependencies
3. Program Interfaces
   a. Customer*
   b. Associate Contractors
   c. Subcontractors*
   d. Prime Contractor*
   e. Corporate Management
   f. Vendors
   g. Politics*

[그림1] CMU 기술문서의 위험분류에 따른 위험요소

대문자로 표시된 A, B, C가 클래스이며 하위 레벨의 엘리먼트와 최하위 레벨의 애트리뷰트를 나타낸 것이다. 관련연구 2.1.1의 Generic Software Project Risk Factors에서 표현하는 방법보다 상세하게 위험을 구분하고 있으나 위험을 측정할 수 있는 기준이나 측정 방식 등은 표현하고 있지 않다[6].

2.2 위험관리리스트

위험관리 리스트는 소프트웨어 개발 프로젝트에서 발생할 가능성이 있는 위험을 식별하고, 우선순위와 해결 방법을 기술한 양식이다. [그림 2]는 BARRY W. BOEHM 의 Software risk management 에서 나타낸 위험관리리스트이다. 위험 아이템으로 소프트웨어 프로젝트에서 빈번히 발생되는 위험요소를 10 개 선정하여 우선순위, 월별 발생빈도 및 각 위험요소를 해결한 과정을 리스트의 필드로 지정하여 기술한다[7].

Risk item	Monthly ranking: This	Last	No. of months	Risk-resolution progress
Replacing sensor-control software developer	1	4	2	Top replacement candidate unavailable
Target hardware delivery delays	2	5	2	Procurement procedural delays
Sensor data formats undefined	3	3	3	Action items to software, sensor teams, due next month
Staffing of design V&V team	4	2	3	Key reviewers committed; need fault-tolerance reviewer
Software fault-tolerance may compromise performance	5	1	3	Fault-tolerance prototype successful
Accommodate changes in data bus design	6	—	1	Meeting scheduled with data-bus designers
Test-bed interface definitions	7	8	3	Some delays in action items; review meeting scheduled
User interface uncertainties	8	6	3	User interface prototype successful
TBDs in experiment operational concept	—	7	3	TBDs resolved
Uncertainties in reusable monitoring software	—	9	3	Required design changes small, successfully made

[그림 2] 위험관리 양식 작성 예

다음 [표 1]은 다른 위험관리리스트양식으로 국내 기업 L 사에서 실제 사용하고 있는 것으로 [그림 2]에서 제시한 리스트 보다 상세한 정보를 담고 있다. 본 논문에서는 [표 1]의 위험관리리스트를 기반으로 연구를 수행한다.

[표 1] L 사의 위험관리리스트

순번	유형	단계	범주	이슈 번호	이슈 제목/ 상세내역	발생일	해결 예정일	해결일	중요도	긴급성	상태	담당자	처리(진행)내역	비고(참조 파일)
1														
2														
3														
4														
5														

2.3 요구사항 추적테이블

요구사항 추적테이블은 소프트웨어 개발 생명주기의 전 단계에 걸쳐 요구사항의 일관성 유지와 완전성 검증 및 요구사항의 변경을 관리하기 위한 것으로 가장 널리 사용되고 있는 요구관리 방법이다. 요구사항 추적테이블은 표준화된 양식이 존재하고 있는 것은 아니므로 본 논문에서는 기존에 연구된 바 있는 산출물들 간의 일관성과 완전성의 검증을 위해 개선된 요구사항 추적테이블을 참조한다. 제시된 추적테이블에서는 기존의 여러 상용도구와 기관에서 사용하고 있는 요구사항 관리 방법의 문제점을 개선하여 마르미III 방법론에서 요구사항과 함께 추적 관리될 필요가 있다고 판단한 산출물들의 추적관리가 함께 이루어지도록 하였다. 본 논문에서는 이 추적테이블을 기반으로 하여 위험관리를 연계하는 방안을 제시한다[8].

[표2] 마르미 산출물간의 연관성을 포함한 요구사항추적테이블

요구사항ID	프로젝트 기안서 프로젝트 정의서	프로젝트 제안서	프로젝트 수행계획서	요구사항	사용자/관련 부서	구현방안/해결방안						연수기준ID
요구사항(Re)1					A							

3. 요구사항테이블과 위험관리리스트의 연결

3.1 요구사항추적테이블의 위험도필드 작성

요구사항 추적테이블과 위험관리 리스트를 연결하기 위하여 각각에 링크 필드를 추가한다. 즉 요구사항추적테이블의 추적필드에는 위험도를 추가로 포함하며, 위험관리리스트에는 요구사항 필드를 포함하여 이들 간에 연결성을 부여한다. 요구사항 추적테이블에 위험도필드를 작성하기 위해서는 위험도 값을 결정해야 하며, 위험도 값은 위험을 일으키는 위험요소들을 예측하여 추출한 다음, 산정방식에 따라 측정하여 결정된다.

3.1.1 위험요소 추출

요구사항추적테이블에 위험도필드를 추가하기 위해서는 필드를 구성하는 레코드들의 값을 정해야 한다. 요구사항 수집, 분석 단계의 작업을 마친 후 수집한 각각의 요구사항들이 개발 프로젝트를 진행하는 과정 중에 어떤 위험요소들이 발생할 수 있을 것인가를 찾는다. 개발자 본인이 예측하여 위험요소들을 찾을 수도 있지만 관련연구에서 언급했듯 여러 기관의 문서와 논문에서 위험요소들에 대한 리스트를 제공하고 있으므로 이런 문서들을 참조하여 위험요소들을 추출할 수 있다. 본 논문에서는 관련연구 2.1.1 에서 언급한 위험요소들을 PMBOK 에서 제시하는 위험요소 식별 기법 중 델파이 기법을 활용하여 우선순위가 높은 핵심 위험요소들을 추출한다[1][4].

[표 3]Generic software risk factors 에서 추출한 위험요소

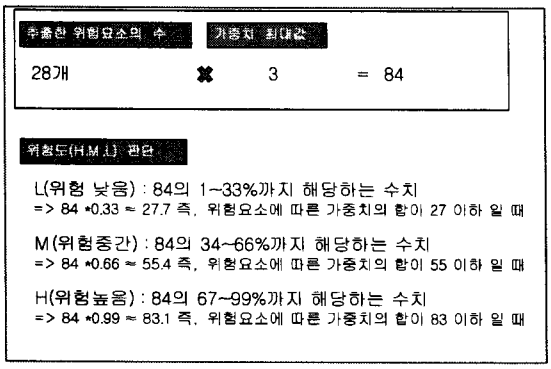
위험 ID	위험요소(Risk factors)	위험 ID	위험요소(Risk Factors)
category: Mission and Goals			외부의 하드웨어 또는 소프트웨어 인터페이스
	고객의 지각	category: Development Environment	
category: Decision Drivers			사용 가능한 툴
	일맞은 일정		벤더지원
	기술선정		계약 일치
category: Customer/User		category: ProjectTeam	
	사용자 참여		이용 가능한 팀 멤버
	사용자의 수락		어플리케이션 경험
category: Product Content			프로젝트 하드웨어 소프트웨어 경험
	요구사항의 안정성		프로세스의 경험
	요구사항의 완벽함과 명확함		어플리케이션 영역 (도메인) 전문적 지식
	테스트가능성	category: Technology	
	설계의 어려움		프로젝트 팀의 기술적 경험
	구현의 어려움		기술에 대한 전문지식의 사용가능성
	시스템 의존성	category: Maintenance	
category: Deployment			설계의 복잡성
	응답 시간 그밖에 성능관련요소		개별적인 지원
	고객 서비스 효과		벤더지원
	요구된 데이터의 마이그레이션	합계	28개의 위험속성

3.1.2 위험도 측정

추출한 위험요소들은 개발이 진행되는 중에 실제로 발생할 가능성이 높다고 판단되는 것을 가장위험(3 점)으로 보고 그 다음은 보통위험(2 점), 위험낮음(1 점), 위험없음(0 점)으로 가중치를 주는 3 점 척도 계산 방식으로 표현하도록 한다. 위험요소들 자체는 사실상 추적테이블 상에 한번에 보여지기 어렵기 때문에 이를 효과적으로 표현할 수 있도록 위험도라는 필드로 작성한다. 위험도는 위험요소 별 가중치를 측정하여 결과에 따라 3 가지 범주(High, Medium, Low)로 구분한다.

본 논문에서는 특정 요구사항에 대한 위험도를 측정하기 위하여 이전 단계에서 예측하여 추출한 다수의 위험요소에 3 점 척도 방식을 활용하여 위험도를 결정한다. [표 3]의 요구사항 별 위험도 산정표를 활용하여 해당 요구사항에 위험요소 1 에 대한 값이 위험높음으로 판단되면 3 점, 위험중간으로 판단되면 2 점, 낮음 이면 1 점, 없으면 0 점으로 가중치를 부여한다. 이것을 결정하는 기준은 관련연구 [4]의 문서에서 제공하는 위험요소에 따라 3 가지의 기준으로 해당 위험요소를 가지고 있는 위험상황을 표현하고 있는 것을 참조한다. 이 문서에서 제공하는 3 가지 기준은 높음, 중간임, 낮음이며 높음일 때를 3 으로

낮음일 때를 1 로 판단하였고 관련 위험요소에 대한 위험이 없으면 0 으로 표현하였다. 이 부분은 3.5 절 사례연구에서 예를 들어 자세히 설명하도록 한다.부여된 가중치를 합한 결과는 값에 따라 위험도 범주인 H/M/L 즉 High, Medium, Low 의 형태로 요구사항 추적테이블의 위험도 필드에 표현된다. 위험도의 3 가지 범주는 사용자가 임의로 정해놓은 기준값에 의해 결정된다. [표 4]는 가중치 산정 결과에 따라 위험도 범주를 H/M/L 중 결정하여 요구사항 추적테이블에 표현한 것이다. 본 논문에서는 위험도 판단의 편의를 위해 다음과 같이 기준을 정하였다. 위험요소 가중치의 최대값인 '3'을 위험요소의 수와 곱한 수를 통해 이 값의 약 1~33% 까지를 L, 34~66%를 M, 나머지 67~99% H 로 표현한다. 다음의 [그림 2]는 위의 내용을 간단히 표현한 것이다.



[그림 2] 위험도 판단

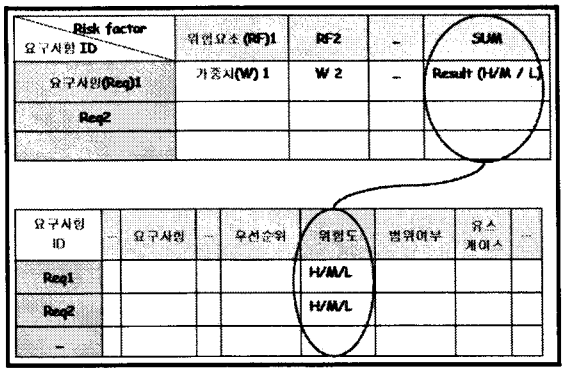
[표 4] 요구사항 별 위험요소의 위험도 산정표

요구사항별 위험요소 가중치를 측정

Risk factor / Requirement ID	위험요소 (RF)1	위험요소 (RF)2	-	SUM
요구사항(Req)1	가중치 1	가중치 2	...	Result (H/M/L)
Req2				

[표 5] 위험도 필드를 추가한 요구사항추적테이블

요구사항 ID	...	요구사항	...	우선순위	위험도	범위여부	유스케이스	...
Req1					H/M/L			
Req2					H/M/L			
-								



[그림 4] 위험도 산정표와 요구사항 추적테이블의 관계

3.2 위험관리리스트

위험관리리스트는 앞서 언급한 관련연구에서 L 사의 양식을 참조하였다. 위험관리리스트에는 위험 ID 번호와 유형, 제목/상세내역, 해결예정일, 해결일, 중요도, 긴급성과 같은 필드가 포함되어있다. 요구사항 추적테이블과의 연결을 위하여 위험관리리스트의 필드에 요구사항 추적테이블의 추적필드 중 요구사항 ID 혹은 요구사항 번호를 추가로 포함한다. 위험관리리스트에 포함되는 요구사항 ID 필드는 요구사항 추적테이블의 필드를 그대로 기록하는 것이 아니며, 해당 위험을 가지고 있는 요구사항의 ID 를 기록하도록 한다. 위험관리리스트에 위험정보를 기록할 때에는 요구사항추적테이블에서 위험도가 H 인 요구사항들을 선택하여 이러한 요구사항들에서 발생할 수 있는 위험정보를 기록한다.

[표 6] 요구사항 ID 를 포함한 위험관리리스트

위험 ID	...	제목/상세내역	발생일	해결예정일	해결일	중요도	긴급성	...	요구사항 ID
Risk1									Re1
Risk2									Re2
...									...

3.3. 요구사항추적테이블과 위험관리리스트 연결

3.1 과 3.2 절에서 요구사항추적테이블에는 추적필드 중 위험도를 추가하였고 위험관리리스트에는 요구사항 ID 를 포함하였다. 그러나 이와 같은 경우에는 위험관리리스트에서 요구사항추적테이블로의 추적은 가능하나 역으로 요구사항 추적테이블에서 위험관리리스트로의 추적은 불가능하다. 요구사항관리와 위험관리 산출물간의 연결은 상호간의 자유로운 추적이 가능할 정도로 긴밀해야 의미가 있다.

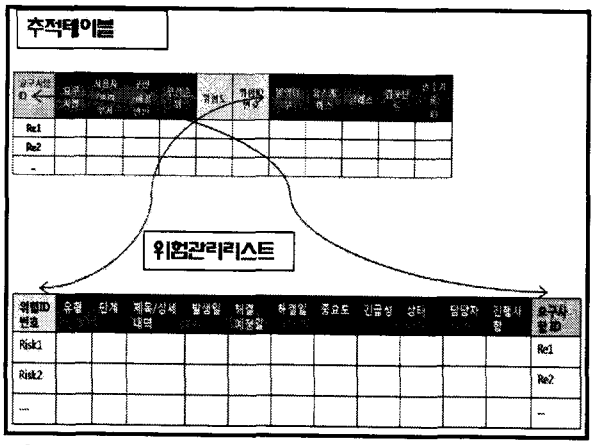
따라서 요구사항 추적테이블에 해당 요구사항이 가지고 있는 위험에 대한 ID 를 필드에 추가적으로 포함한다. 아래의 [표 7]은 일방적인 추적만 가능했던

이전 [표 5]에서의 요구사항 추적테이블에 위험 ID 를 추가적으로 포함한 것이다. 앞 절의 위험관리리스트와 마찬가지로 위험 ID 도 위험관리리스트의 위험 ID 필드를 그대로 가져오는 것이 아니라 해당 요구사항이 가지고 있는 모든 위험의 ID 를 기록한다.

[표 7] 위험 ID 를 포함한 요구사항추적테이블

요구사항 ID	...	요구 사항	...	우선 순위	위험도	위험ID 번호	범위여부	유스케이스	...
Req1									
Req2									
-									

연결을 위한 링크 필드를 추가한 두 산출물간의 연결관계는 다음 [그림 4]와 같다.



[그림 4] 추적테이블과 위험관리리스트간의 연결관계

요구사항 ID, 위험 ID 번호와 같은 특정 필드들을 링크필드로 사용하여, 두 산출물 간 연결이 긴밀해 짐을 알 수 있다. 이로써 개발자들은 두 가지 산출물을 한번에 보고 관리할 수 있게 된다.

즉 개발자들이 개발 과정 중에 지속적으로 사용하는 요구사항추적테이블을 통해 위험관리리스트를 볼 수 있으므로 위험에 대한 정보를 얻고 사용하기 용이해 진다.

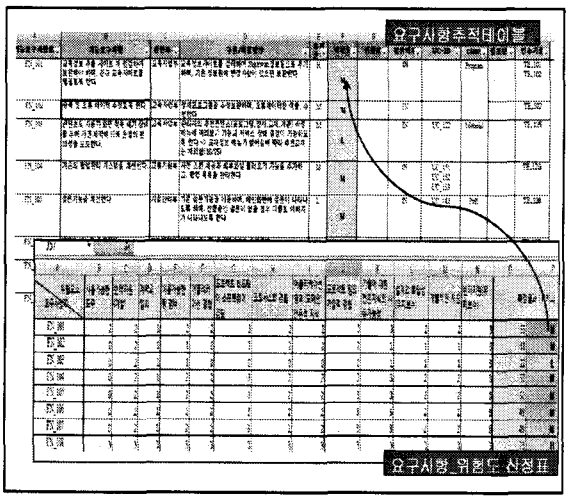
3.4 요구사항관리와 위험관리의 연결절차

본문 3.1 에서부터 3.3 의 내용에 따라 요구사항관리와 위험관리의 연결절차 중 요구사항추적테이블과 위험관리리스트를 연결하는 방법을 정리하면 다음과 같다.

① 요구사항추적테이블의 추적필드에 위험도와 위험요소 ID 를 추가한다.
② 위험관리리스트에는 요구사항 ID 필드를 추가한다.
③ 개발과정에서 발생할 가능성이 있는 위험요소를 위험도 산정표와 위험관리리스트에 기록한다.
④ 위험도 산정표에 따라 요구사항 별 위험요소들에 대한 가중치를 측정, 합산하여 위험도 값을 결정한다.
⑤ 해당 요구사항의 위험도 값에 따라 H/M/L 을 요구사항 추적테이블에 표현한다.
⑥ 위험도가 H 인 요구사항에 한하여 해당 요구사항이 가지고 있는 위험정보를 위험관리리스트에 작성한다.
⑦ 작성한 위험정보의 위험 ID 번호를 요구사항 추적테이블에 기록하고 위험관리리스트에는 해당 위험 ID 에 대한 요구사항 ID 를 기록한다.

3.5 사례연구

[그림 5]와 [그림 6]은 요구사항 추적테이블과 위험관리 리스트의 연결 관계를 나타내기 위하여 실 사례를 적용한 것이다. 요구사항 추적테이블은 서울시 산하의 공공기관의 업무시스템 구축 시 분석한 요구사항을 예로 제시하였다. 요구사항을 분석한 후 추적테이블에 기술된 요구사항은 관련연구[4]에서 추출한 위험요소를 포함하여 위험도산정표에 작성한다. [그림 5]는 위험도산정표에서는 위험요소 별 가중치를 측정하여 그 결과를 H 혹은 M, L 으로 작성하여 요구사항 추적테이블의 위험도 필드에 작성한 내용이다.

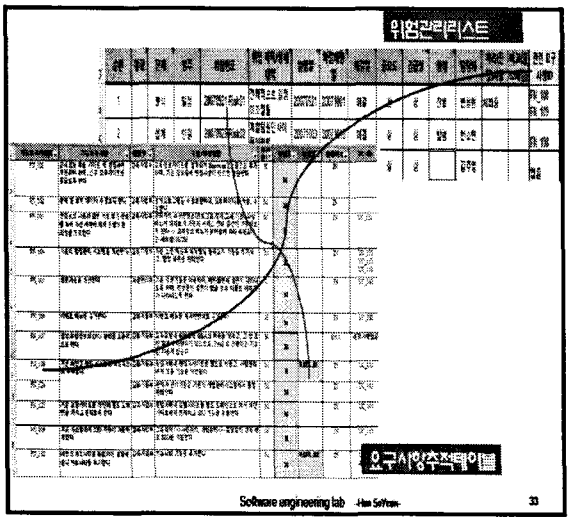


[그림 5] 추적테이블의 위험도 필드 내용 작성

이 후 작성된 위험도가 H 인 것을 선별하여 위험관리 리스트의 위험정보에 기록하는데 이때 요구사항의 ID 도 함께 등록하여 개발자들이 특정 요구사항에 어떤 위험 정보가 존재 하는지를 파악할 수 있게 한다. 즉 위험관리리스트와 요구사항 추적테이블을 링크필드를

통해 두 산출물 간 상호 연결이 가능해 짐으로 개발자들이 요구사항 별 위험정보를 기록하거나 확인하는 것이 가능해 지며 역으로 위험관리리스트에 존재하는 위험이 어떤 요구사항과 관련이 있게 되는지를 알 수 있다.

[그림 6]은 위험도가 높은 요구사항의 ID 를 위험관리리스트에 기록하고 그 요구사항에서 발생할 수 있는 위험정보들을 위험관리리스트에 기록하는 것이다.



[그림 6] 추적테이블과 위험관리리스트

4. 결론 및 향후 연구

본 논문은 현재 소프트웨어 개발 프로젝트에서의 위험관리가 관리자 레벨에서 관리되어 실제 위험정보가 필요한 개발자들에게는 적절하게 보여주고, 전달되고 있지 않고 있음을 개선하기 위하여 개발자들을 위한 위험관리방법에 대한 연구를 수행하였다. 개발자들에게 효과적으로 보여줄 수 있는 위험관리 방법으로는 산출물들간의 연결을 제안하였다. 산출물로는 개발자들이 실제 개발생명주기 전반에 걸쳐 사용되고 있는 요구사항 추적테이블을 선정하였다. 이를 위험관리 산출물인 위험관리리스트와 링크 필드를 통해 연결한다. 두 산출물의 연결을 통해 개발자들은 요구사항 분석 후 예측할 수 있는 모든 위험을 찾아 요구사항과 함께 프로젝트가 종료되는 시점까지 추적, 관리할 수 있다. 사실상 요구사항관리와 위험관리는 현재 각기 다른 형태의 프로세스와 작업, 산출물을 가지고 따로 진행이 되고 있어서 프로젝트 성공률을 오히려 저하시키고 있다는 문제점을 내포하고 있는 바, 향후 연구로 산출물들 간의 연결을 통하여 개발자들을 위한 새로운 위험관리산출물 양식을 개발하고, 나아가 방법론 등에 적용할 수 있도록 액티비티로 정립하는 연구가 진행될 것이다.

5. 참고문헌

[1] 한국 프로젝트 관리 협회, "프로젝트 관리지식체계 지침서 3 판 (PMBOK)", pp239~270, 2004

[2] 황만수, "요구공학 기반의 통합된 소프트웨어 요구사항 관리 프레임워크 설계", 2001, 숭실대 대학원 컴퓨터학과 박사학위논문

[3] HENRI BARK, "Toward and Assessment of software development risk", Journal of Management Information Sysunrr 1 Fall 1993. VoL 10. No. 2 pp. 203~225

[4] Texas Project Delivery Framework, "Generic Software Project Risk Factors 1.0", DIR(U.S. Texas department of information resources), 2006

[5] 엄충용, "NFR Graph 를 이용한 위험분석 및 프로젝트 일정예측 기법", 2005, 숭실대 대학원 컴퓨터학과 석사학위논문

[6] Ronald P. Higuera, Yacov Y. Haimes, "Software Risk Management", CMU/SEI technical Report, pp19~37, 1996

[7] BARRY W. BOEHM, Software risk management: principles and practices, IEEE computer society, 1991

[8] 김주영, "산출물의 일관성과 완전성 검증을 위한 추적테이블의 경험적 연구", 한국정보과학회 논문지 제 34 권 제 5 호, pp. 419~430, 2007.05