학술대회 홈페이지
https://www.manuscriptlink.com/society/kips/conference/kips2017fall

제48회 2017

추계학술 발표대회 논문집

THE KIPS FALL CONFERENCE 2017

일 자 2017년 11월 3일(금) ~ 4일(토)

장 소 서울과학기술대학교

주 최 사단법인 한국정보처리학회 KIPS Korea Information Processing Society

주 관 서울과학기술대학교 SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

협 찬 LG히다찌 kpc The Insight KPC 한국생산성본부 IGIS (주)아이지스

Metanet 대우정보시스템

사단법인 한국정보처리학회 KIPS Korea Information Processing Society

클로즈 아키텍처 메커니즘 기반의 요구사항 추적성 매트릭스

변은영*, 손현승, 문소영, 박지훈, 김영철
*홍익대학교 소프트웨어공학연구실
e-mail:{eybyun*, son, msy, pjh, bob}@selab.hongik.ac.kr

Requirement Traceability Matrix Based on Closed Architecture Mechanism

Eun Young Byun*, Hyun Seung Son, So Young Moon, Ji Hoon Park, R. Young Chul Kim
*SE Lab, Dept. of Computer Information Communication, Hongik University

요 약

앞으로의 프로젝트에서는 시장 변화, 신기술, 경쟁업체의 대응, 설계결함, 테스트 실패 등의 다양한 외부적 요인으로 인해, 더욱 빈번한 요구사항 변경이 요청된다. 그 이전에 명료한 요구사항을 정의하기가 매우 어렵고, 소프트웨어 구축 중에도 수시로 요구사항이 변경되고 있는 실정이다. 이런 문제는 요구사항 추적성 및 변경 관리의 미비함에 있다. 이를 해결하기 위해, 소프트웨어 개발 프로세스인 요구사항, 분석, 설계, 구현, 테스트 단계에서의 추적성 관리를 위한 프로세스의 구축과 내재화가 필요하다. 본 논문에서는 클로즈 아키텍처 메커니즘을 기반으로 소프트웨어 각 개발 단계 산출물들 간의 추적성 매트릭스를 제시한다. 이를 통해 프로젝트 과정에서의 잦은 요구사항 변경에 유연하게 대처함으로써 소프트웨어 품질 향상에 기여 할 것으로 본다.

1. 서론

소프트웨어가 대형화, 복잡화되면서 기업은 요구사항 관리에 어려움을 겪고 있다. 요구사항은 소프트웨어 개발 프로세스의 첫 단계로 사용자의 요구를 정의하고 소프트웨어 개발의 로드맵을 제공하는 중요한 역할을 한다. Christopher Lindquist는 "*Fixing the Requirement Mess (요구사항 문제 해결)*" 기사에서 "부실한 요구사항 관리는 소프트웨어 프로젝트의 71%가 실패로 돌아가도록 만들고 있으며, 이는 프로젝트 실패의 가장 큰 원인"이라고 언급한다[1]. 기업이 요구사항 관리에 어려움을 겪고 있는 주요 원인은 시장 변화, 신기술, 경쟁업체의 대응, 설계 결함, 테스트 실패 등의 다양한 외부적 요인으로 인해 고객들로부터 잦은 요구사항 변경 요청을 받고 있는 것이다. 소프트웨어 구현 이후에도 많은 이해관계자들은 지속적으로 변경을 요구한다. 또한, 소프트웨어 개발 과정의 마지막 단계인 테스트에서 요구사항의 충족 여부인 커버리지를 측정해야 한다. 하지만, 요구사항과 테스트의 연계성 부족은 소프트웨어 테스팅을 어렵게 한다.[2,3].

소프트웨어 요구사항 관리는 요구사항 정의 활동과 소프트웨어 개발 프로세스 동안의 요구사항 변경 관리 활동으로 구성된다. 지속적인 요구사항의 변경과 요구사항의 충족 여부를 측정하는 테스트를 효율적으로 수행하기 위해서는 소프트웨어 개발 프로세스인 요구사항, 분석, 설계, 구현, 테스트 단계 간의 추적성을 보장해야 한다[4, 5]. 단, 모든 단계들 간의 추적성을 제공하지 않고, 클로즈 아키텍처 기반으로 연결된 단계들 간의 추적성만을 제공한다. 연결되지 않은 단계는 여러 단계의 추적을 거쳐야 한다. 이를 통해서 기업의 잦은 요구사항 변경에 빠르게 대응하고 소프트웨어 개발의 효율성을 향상시키며 품질을 개선할 수 있다.

다음 장은 관련 연구로 기존에 요구사항 추적성을 제공하는 도구들을 기술·비교한다. 3장은 요구사항 추적성 매트릭스로 소프트웨어의 각 개발 단계에서의 산출물들과 그들의 추적성을 나타내고 구축 사례 설명한다. 마지막으로 4장은 결론 및 향후 연구를 언급한다.

2. 관련 연구

효율적인 요구사항 관리를 위한 다양한 요구사항 관리 도구들이 있다. 대표적인 상용 제품은 CASE Spec, Polarion, IBM Requisite Pro, IBM DOORS, Caliber RM, Aligned Elements 이다. 이 도구들은 요구사항 관리, 협업 시스템, 문서 관리 및 자동 생성, 추적성, 다이어그램, 리스크 관리 기능들을 제공한다. 표 1은 요구사항 관리 상용 도구의 확장 비교이다[6].

CASE Spec은 명세, 요구사항 추적, 분석, 디자인, 테스팅을 위한 라이프 사이클 솔루션을 제공한다. 단일 또는 다중 계층으로 관리되는 아이템, 다이어그램, 파일 간의

<표 1> 요구사항 관리 도구의 확장 비교[6]

Feature	CASE Spec	Polarion	IBM Requisite Pro	IBM DOORS	Caliber RM	Aligned Elements
Requirements Classification	✓	✓	✓	✓	✓	✓
Custom Attributes	✓	✓	✓	✓	✓	✓
Document Management	✓	✓	x	x	x	✓
Multi-dimensional Traceability	✓	x	✓	x	x	✓
Change Management	✓	✓	x	x	✓	✓
Custom Link Types	✓	✓	x	x	x	x
Attributes for Links	✓	x	x	x	x	x
Custom Reporting	✓	✓	x	x	x	✓
Diagramming	✓	x	x	x	x	✓
Includes Enterprise Database Licenses	✓	x	x	x	x	x
Current License (Floating license)	$500.00	$3200.00	$3880.00	$2500.00	$3000.00	$2300.00

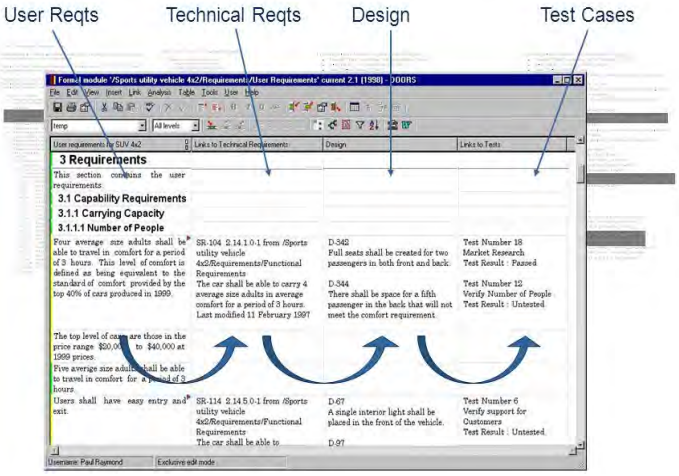
링크를 제공한다.

Polarion[7]은 프로젝트 팀이 익숙한 환경에서 작업할 수 있는 옵션을 제공하면서 효과적이고 투명하며 안전한 협업 시스템을 제공한다. 또한, 문서 재사용을 통해서 제품의 공통점을 관리한다.

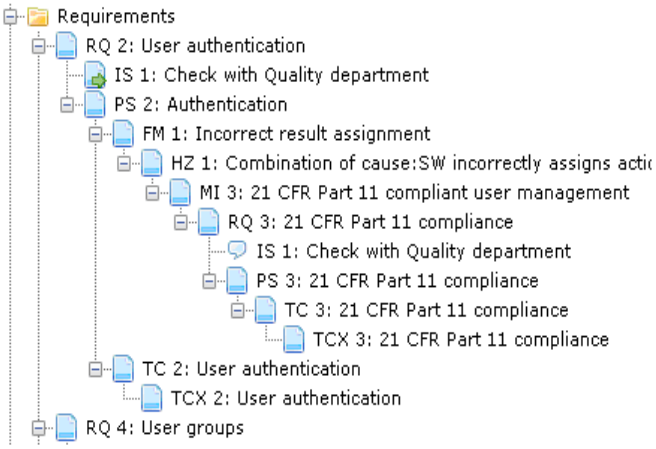
IBM Rational DOORS[8]는 시장 점유율 1위의 요구사항 관리 도구로 Hard Real time 시스템을 개발하는 곳에서 체계적이고 효율적인 요구사항 관리를 위해서 많이 사용되고 있다. DOORS의 가장 큰 장점은 워드와 엑셀과 비슷한 사용자 인터페이스로 편리하게 추적성을 제공한다. 요구사항부터 서브 요구사항, 아키텍처, 설계, 테스트까지 자세한 추적이 가능하게 해준다.

Aligned Elements[9]는 의료 디바이스 개발 문서를 관리하는 특화 시스템이다. 실시간 영향 분석이 가능한 추적성, 무결성, 완전성을 위한 추적성 검사, 리뷰 검사를 제공한다. 또한, 리스크 관리, 자동화된 문서 생성, 높은 재사용성의 특징을 갖는다.

다양한 요구사항 관리 도구가 있지만, 현재까지 추적성을 제공하는 도구는 현저히 적고, 효율성이 부족하다. DOORS에서 추적성을 제공하는 방법은 그림 1과 같다. 다른 도구들과의 차이점은 모든 라인이 식별자(ID)를 갖기 때문에 관리 쉽다는 장점이 있지만, 반대로 관리 항목들이 너무 많아 복잡하다. 또한, 소프트웨어 설계에 중요한 다이어그램과의 추적성을 제공하지 않는다. Aligned Elements에서 추적성을 제공하는 방법은 그림 2와 같다. Aligned Elements는 DOORS와는 다르게 각 요소들의 식별자(RQ, IS, PS, FM, HZ, TC, 등)를 사용하여 트리 형태로 추적성을 나타낸다. 이 방법은 여러 요구사항이 여러 개의 요소(N : M 관계)와 추적성을 갖는다면 이를 구분하기가 쉽지 않다.



(그림 1) DOORS의 추적성[7]



(그림 2) Aligned Elements의 추적성[8]

3. 요구사항 추적성 매트릭스

소프트웨어의 개발 프로세스에서 요구사항 추적성을 위한 클로즈 아키텍처 기반의 매트릭스를 제안한다. 표 2는 소프트웨어 개발 프로세스의 각 단계와 해당 단계에서의 산출물을 나타낸다.

<표 2> 소프트웨어 개발 프로세스의 단계와 산출물

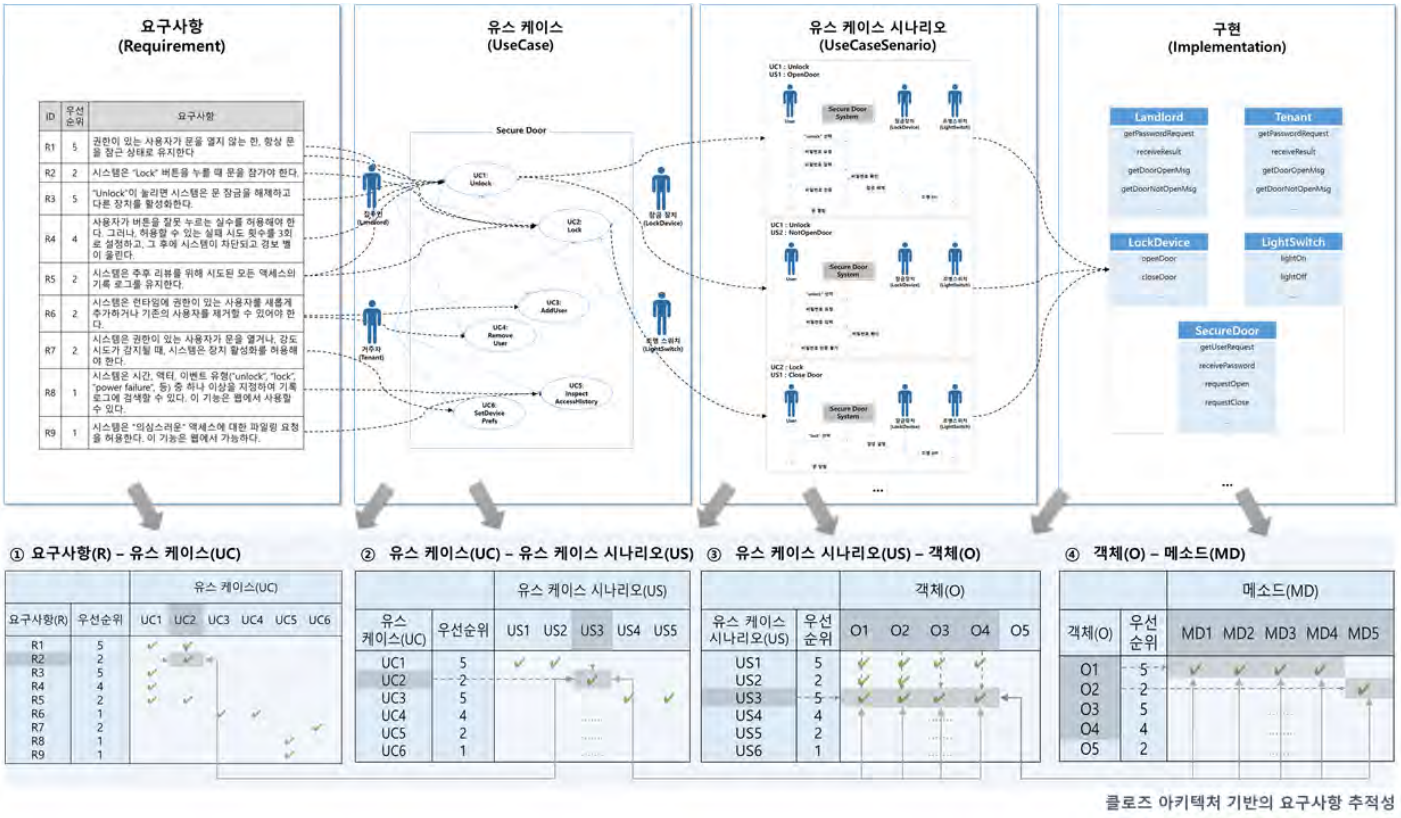
Software Development Process		
Step	Name	Output
1	Requirement	Requirement(R)
2	Analysis	UseCase(UC)
3	Design	UseCaseScenario(US)
4	Implementation	Object(O)
		Method(MD)
5	Test	TestCase(TC)
		Test Script(TS)

총 5가지 단계로 소프트웨어 개발 프로세스가 구성된다. 각 단계들은 순차적으로 실행되고, 인접한 단계의 산출물들 간의 추적성을 하나의 매트릭스로 나타낸다. 표 3은 매트릭스의 대한 예시로 표 2에서 정의한 바와 같이 각 단계의 산출물들의 식별자로 행과 열을 구성한다. 요구사항 명세(Requirement Elicitation) 단계와 분석(Analysis)단계에서 각각의 산출물인 요구사항(R)과 유스 케이스(UC) 간의 추적성 매트릭스를 나타낸다. 각각의 항목들이 서로 연

<표 3> 요구사항과 유스 케이스 간의 추적성 매트릭스

		유스 케이스(UC)					
요구사항 (R)	우선순위	UC1	UC2	UC3	UC4	UC5	UC6
R1	5	✔	✔				
R2	2		✔				
R3	5	✔					
R4	4	✔					
R5	2	✔	✔				
R6	1			✔	✔		
R7	2						✔
R8	1					✔	
R9	1					✔	

관성이 있을 경우 체크 표기한다. 다중 체크도 가능하므로 추적성의 관계가 1 대 다(one to many) 일 때 이를 쉽게 추적할 수 있다. 뿐만 아니라 유스 케이스 다이어그램이나 시퀀스 다이어그램과의 추적성 또한 제공할 수 있다. 예제로써 *'건물 도어락 시스템'*의 개발 프로세스에 이 매트릭스를 적용한다. 이 시스템에서는 건물주와 주거 자가 도어락 비밀번호를 통해 건물에 출입하며, 출입 시에는 입구 조명이 제어된다. 그림 3은 도어락 시스템의 각 개발 단계의 산출물들을 간략히 표현하고 그들 간의 추적성 매트릭스들을 보여 준다. ①요구사항(R)과 유스 케이스(UC), ②유스 케이스(UC)와 유스 케이스 시나리오(US), ③유스 케이스 시나리오(US)와 구현에서의 매트릭스가 생성되는데 구현 단계에서의 산출물들은 ④객체(O)와 메소드(MD)로 구분할 수 있으므로 총 4개의 매트릭스가 생성된다. 모든 단계의 산출물들이 연결되지 않고 인접된 단계의 산출물들



(그림 3) 소프트웨어 개발 프로세스의 추적성 매트릭스

을 연결하는 클로즈 아키텍처이다[10]. 이러한 매트릭스는 각각 단계별 산출물들 간의 추적을 가능하게 하고, 소프트웨어를 개발하는 어떤 단계에서든 수정 시에 프로세스 산출물들의 추적을 통한 효율적인 유지보수가 가능하다. 단, 인접해 있는 단계로만 추적이 가능하기 때문에, 인접해 있지 않은 단계는 여러 단계의 추적을 거쳐야 한다. 이를 기반으로 그림 4와 같은 요구사항 추적 시스템을 구축했다. 일감 추적을 통해서 각 개발 프로세스의 항목들을 전체 개수와 진행 중, 완료된 항목들을 구분할 수 있고, 이들은 링크를 통해 상세한 내용을 확인할 수 있다. 프로젝트의 전체 일정 관리는 Gantt 차트를 통해서 수행한다. 또한, 정의한 매트릭스를 적용하여 클로즈 아키텍처 기반의 추적성을 보장한다. 이를 통해 잦은 요구사항 변경에 유동적으로 대처할 수 있으며 소프트웨어 테스팅과 유지보수를 효율적으로 수행하여 소프트웨어 개발의 효율성과 품질을 개선할 수 있다.

4. 결론

소프트웨어 기업들은 이해관계자들의 잦은 요구사항 변경 요청이나 시장 변화, 설계결함, 테스트 실패에 의한 반복적인 요구사항 변경을 피할 수 없다. 이런 상황에서의 유동적인 대처와 효율적인 소프트웨어 개발을 위해 요구사항 추적성 관리가 필요하다. 현재까지 많은 도구들이 요구사항 관리 목적으로 사용되지만, 소프트웨어 설계에서 중요한 요소인 다이어그램과의 추적성을 제공하지 않을 뿐만 아니라 한눈에 확인하기 어렵고, 여러 개의 산출물들이 연결(N : M 관계)되어 있을 때 이를 구분하는 것 또한 어려운 실정이다.

이런 문제의 해결을 위해 소프트웨어 개발 프로세스에서의 추적성 매트릭스를 제안한다. 이는 클로즈 아키텍처 기반으로 인접한 두 단계에서의 산출물들 간의 연관성을 매트릭스로 나타낸다.

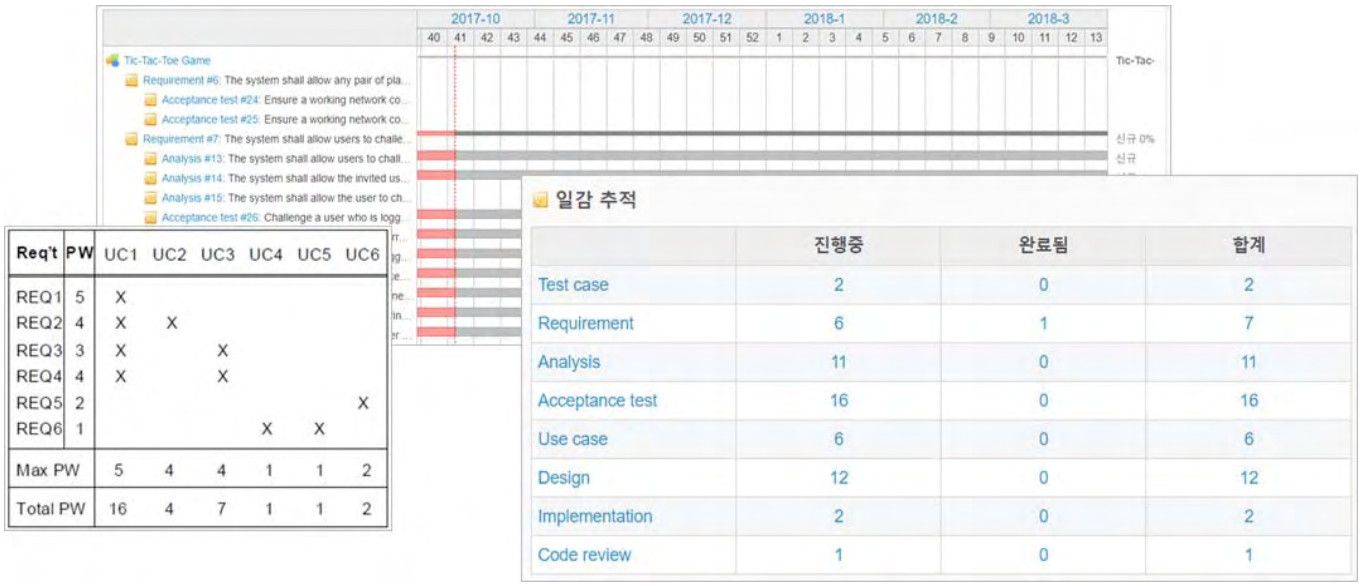
그 결과, 기업들은 잦은 요구사항 변경 요청에 효과적인 대응과 비용/시간의 절감이 가능하다. 따라서 소프트웨어 개발의 효율성과 품질을 향상시킨다. 향후에는 소프트웨어 전체 개발 프로세스에서의 추적성을 오픈 아키텍처로 개선하여 인접한 단계뿐만 아니라 전 단계에서 제공하고자 한다.

ACKNOWLEDGE

이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2017R1D1A3B03035421)

참고문헌

[1] Michelle Boucher, “6 Reason Your Projects Fail: How Requirements Management Can Help”, PTC.
[2] 박수진, “분석모델 자동생성 기반의 비즈니스 어플리케이션 서비스 추적 방안”, 서강대학교, 2007.
[3] Christopher Lindquist, “Fixing the Requirements Mess”, CIO Magazine.
[4] 서채연, 박보경, 문소영, 김영철, “비즈니스 프로세스 프레임 구축 상에서 기능적 요구사항으로부터 효율적 비즈니스 프로세스 추출”, 2014 한국 인터넷 방송통신학회 종합학술대회, 2014.
[5] 박보경, 권하은, 문소영, 이유진, 김영수, 이상은, 박용범, 김영철, “요구사항 추적성을 위한 요구사항 추적 모델”, 한국정보처리학회, 2015.
[6] PSEG, “INCOSE 요구사항 관리 도구 비교 조사”, 2014.
[7] ALM, “https://polarion.plm.automation.siemens.com/”.
[8] IBM, “Rational DOORS Documentation Library”.
[9] Aligned, “Aligned elements - Feature”.
[10] Ivan Marsic, “Software Engineering”, Rutgers.



Req't	PW	UC1	UC2	UC3	UC4	UC5	UC6
REQ1	5	X					
REQ2	4	X	X				
REQ3	3	X		X			
REQ4	4	X		X			
REQ5	2						X
REQ6	1				X	X	
Max PW		5	4	4	1	1	2
Total PW		16	4	7	1	1	2

	진행중	완료됨	합계
Test case	2	0	2
Requirement	6	1	7
Analysis	11	0	11
Acceptance test	16	0	16
Use case	6	0	6
Design	12	0	12
Implementation	2	0	2
Code review	1	0	1

(그림 4) 요구사항 추적성 매트릭스 구현 결과