가치분석을 통한 휘처 기반의 요구사항 추적 기법

A Feature-Oriented Requirements Tracing Method with Value Analysis

안상임(Sangim Ahn)*, 정기원(Kiwon Chong)**

초 록

추적 링크는 요구사항과 아키텍처 설계서, 소스 코드, 테스트 케이스 등과 같은 시스템 산출물들 사이의 논리적인 연결을 의미한다. 이러한 추적 링크는 요구사항 변경 영향 분석, 요구사항 충돌 분석, 요구사항 일관성 점검에 매우 유용하다. 그러나, 복잡한 소프트웨어 개발 시 많은 다양한 산출물들이 만들어지므로 추적 링크를 생성하거나 운용하는 것은 많은 부담을 초래한다. 본 논문은 가치분석을 근간으로 하고 휘처를 중간 매개체로 활용하는 휘처 기반의 요구사항 추적 기법을 제안한다. 이는 사용자 요구사항과 산출물간의 추적 링크를 생성하기 위한 중간 매개체로 휘처 개념을 적용하였고, 식별된 휘처들의 상세화 정도를 추정하기 위하여 요구사항의 우선순위에 따른 가치 평가를 포함한다. 또한, 본 논문에서 제안한 휘처 기반의 요구사항 추적 기법을 아파트 단지 내 홈 서비스를 통합하는 유비쿼터스 플랫폼에 적용한 사례의 결과를 기술한다.

ABSTRACT

Traceability links are logical links between individual requirements and other system elements such as architecture descriptions, source code, and test cases. These are useful for requirements change impact analysis, requirements conflict analysis, and requirements consistency checking. However, establishing and maintaining traceability links places a big burden since complex systems have especially yield an enormous number of various artifacts. We propose a feature-oriented requirements tracing method to manage requirements with cost benefit analysis, including value consideration and intermediate catalysis using features. Our approach offers two contributions to the study of requirements tracing: (1)We introduce feature modeling as intermediate catalysis to generate traceability links between user requirements and implementation artifacts. (2)We provide value consideration with cost and efforts to identify traceability links based on prioritized requirements, thus assigning a granularity level to each feature. In this paper, we especially present the results of a case study which is carried out in Apartment Ubiquitous Platform to integrate and connect home services in an apartment complex in details.

키워드 : 요구사항 추적성, 휘처 모델링, 휘처기반 요구사항 추적 프로세스, 비용-효과 분석
Requirements Traceability, Feature Modeling, Feature-Oriented Requirements Tracing Process, Cost-Benefit Analysis

---

본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.

* 숭실대학교 대학원 컴퓨터학과
** 숭실대학교 컴퓨터학부 정교수

1. 서 론

소프트웨어 개발 초기에 모든 요구사항을 정의하는 것은 불가능하다. 따라서 요구사항은 소프트웨어 개발이 진행되는 동안에 지속적으로 변경된다. 이러한 요구사항 변경은 개발자가 소프트웨어 구조나 행위를 완벽하게 이해하지 못하거나 변경에 따라 영향을 받는 모든 부분을 식별할 수 없을 경우 많은 오류를 야기 시킨다. 요구사항 추적 링크는 소프트웨어 분석 단계에서 개발 단계까지 요구사항 변경 분석, 요구사항 충돌 분석, 요구사항 일관성 점검하는데 도움을 준다. 그러나 소프트웨어 개발 시 요구사항 명세서, 아키텍처 설계서, 소스 코드, 테스트 케이스 등과 같은 다양한 산출물들이 생성된다. 특히, 복잡한 시스템의 경우는 대량의 요구사항 추적 링크가 생성되기 때문에 소프트웨어 엔지니어들이 이러한 추적 링크를 생성하거나 유지보수를 하는데 많은 부담이 초래된다. 그러므로 비용과 복잡도와 관련한 이슈들 때문에 실제 업무에서 요구사항 추적 링크를 적용하기가 매우 어렵다[1].

본 논문에서는 추적 링크를 비용-효과적으로 생성하기 위한 휘처 기반의 요구사항 추적 기법(Feature-Oriented Requirements Tracing Method)을 제안하고자 한다. 이는 가치분석을 근간으로 하며 재사용 가능한 요구사항 요소들을 생성하기 위해서 소프트웨어 제품 계열에서 중요한 개념으로 사용되고 있는 휘처를 중간 매개체로 활용한다. 즉, 사용자 요구사항과 다른 산출물들간 연결에 휘처 모델링을 통하여 생성된 휘처들이 중간 매개체의 역할을 담당하게 된다. 가치분석의 목적은 비용과 노력을 고려한 요구사항 우선순위를 근거로 해서 비용-효과적인 추적 링크를 식별하는 것이다. 각각의 휘처들의 상세화 정도는 가치분석을 통한 결과를 토대로 결정된다. 휘처 기반의 요구사항 추적을 위해서 요구사항을 포함한 산출물들과 휘처간의 관계를 메타 모델을 통하여 정의하고, 추적 링크가 생성되고 유지보수되는 과정을 설명하기 위해서 전체적인 프로세스를 제시한다. 메타 모델은 요구사항, 휘처, 추적 링크와의 관계를 UML을 이용하여 형식화한 것이다. 휘처 기반의 요구사항 추적 프로세스는 요구사항 정의, 휘처 모델링, 휘처 우선순위화, 요구사항 연결, 추적 링크 평가로 구성된다. 또한, 본 논문에서 제안한 휘처 기반의 요구사항 추적 기법의 타당성을 보여주기 위해서 아파트 단지내 홈 서비스를 통합하는 유비쿼터스 플랫폼에 적용한 사례의 결과를 자세히 기술한다.

제 2장에서는 휘처 모델링 및 요구사항 추적성과 관련된 연구를 소개하며, 제 3장에서는 휘처 기반의 요구사항 추적을 위한 메타 모델링과 전체 프로세스에 대하여 제시한다. 제 4장에서는 아파트 유비쿼터스 플랫폼에서의 사례연구를 통하여 제안된 기법을 실용성을 검증한다. 제 5장에서는 본 연구에서 제안된 기법과 관련된 기법들을 여러 기준 항목으로 비교한다. 마지막으로 제 6장에서는 결론을 기술하고 향후 연구를 제시하며 본 논문을 마무리 한다.

2. 관련 연구

1.1 가치기반 소프트웨어 공학

많은 기업들은 요구사항 추적, 결함 추적,

형상관리 등과 같은 소프트웨어 공학 실행을 가치-중립(Value-neutral) 방식으로 수행하고 있으며, 이러한 실행들은 자주 비즈니스와 연관된 가치의 이해가 없는 상태로 진행된다. 그러나 IT 환경이 급변하는 오늘날 기업이 성공하기 위해서 투자는 보다 현명하게 이루어져야 하며 더 이상 가치가 고립된 상태로 소프트웨어 공학이나 정보기술이 수행될 수 없다.

이를 해결하기 위해서, Boehm[2]은 기존 또는 새로 만들어지는 소프트웨어 공학 원리 및 실행과 이것들이 상호 조화롭게 강화되기 위한 전반적인 프레임워크를 개발하는데 가치 중요성을 통합하기 위하여 가치 기반 소프트웨어 공학(Value-based software engineering, VBSE)을 제안한다. VBSE는 가치 기반 환경하에서 소프트웨어 공학 행위들이 기본으로 사용하게 되는 Benefits Realization Analysis, Stakeholder Value Proposition Elicitation and Reconciliation, Business Case Analysis, Continuous Risk and Opportunity Management, Concurrent System and Software Engineering, Value-Based Monitoring and Control, Change as Opportunity과 같은 일곱 가지 원리를 포함한다. 또한, Boehm은 시스템 개발 초기에 비즈니스 가치와 연관되어야 할 몇몇 소프트웨어 공학 실행들에 대하여 기술하고 있다. 그 중에서 하나가 가치 기반 요구공학(Value base Requirement Engineering, VBRE)이다. 가치 기반 요구공학은 중요한 이해당사자 식별, 이해당사자의 요구를 시스템의 기능에 지속적으로 집목, 상호 목적에 부합하는 시스템 개발과 관련된 원리 및 실행에 대하여 기술하고 있다. 요구사항이나 이해당사자의 목적은 개발중인 시스템이 고객에게 가치를 제공하고 언제나 고객의 요구를 정확하게 반영한다는 확신을 주기 위하여 관리되고 추적될 필요가 있는 요소이다. 요구사항 추적성은 프로젝트 생명주기 또는 이외에서 요구사항을 추적 및 관리를 가능하게 하는 메커니즘이다. 요구사항 추적의 가치는 프로젝트 실패의 주요 요인이 소프트웨어 개발 생명주기 동안 요구사항과 관련된 문제이기 때문에 매우 중요하다. 따라서 요구사항 추적이 소프트웨어 공학의 주요 행위로 이해되기 위하여 행위 자체로서가 아니라 프로젝트에서 가치를 제공하여야 한다. 즉, 가치를 이해하기 위해서 다양한 요구사항들을 가지고 있는 여러 기업들에게 유용하도록 융통성 있는 방식을 제공해야 한다.

Zemont[3]는 이러한 요구사항 추적의 가치를 보여주기 위하여 VBSE의 일곱 가지 원리를 근간으로 다음과 같은 가치 기반 요구사항 추적 프레임워크(Value-based Requirements Traceability Framework)을 제안한다. 이때, VBSE의 원리는 프로세스로 이용된다.

2.2 휘처 모델링

본 논문에서는 요구사항 추적 링크를 생성하기 위하여 중간 매개체로 소프트웨어 제품계열의 휘처(Feature) 개념을 적용하였다. 휘처는 제품의 주요 특성을 의미한다. 휘처 모델링은 도메인에서 외부적으로 보여지는 제품의 특성을 식별하기 위한 활동으로 이를 통하여 휘처들이 휘처 모델로 조직화된다. Kang[4]에서는 휘처 모델 개념이 기술되어 있고 Lee[5]에서는 소프트웨어 제품계열공학을 위하여 실용적으로 휘처를 모델링하는 가이드라인을 제시하고 있다.

휘처는 성능(Capability), 도메인 기술(Domain technology), 구현 기술(Implementation technique), 운영 환경(Operating environment)로 분류된다. 성능 휘처는 도메인 내의 서비스, 동작 또는 비기능적인 관점에서의 특징들을 나타낸다. 따라서 이 영역은 크게 기능적(Functional) 휘처와 비기능적(Non-functional) 휘처로 구성되어 있다. 기능적 휘처는 서비스(Service)와 그 서비스에 대한 오퍼레이션(Operation)을 결정짓는 휘처로 하위계층의 휘처들을 제어하게 된다. 비기능적 휘처는 시스템의 용도, 성능, 최대 사용자 수 등과 같이 실제 구현할 때 중요한 의미를 지는 것들이다. 도메인 기술 휘처는 도메인에서의 특별한 문제나 서비스 또는 오퍼레이션 휘처를 구현하기 위하여 도메인 분석가나 아키텍처 설계자들이 사용하는 특징 기술들을 나타낸다. 구현 기술 휘처는 도메인 기술 휘처보다는 일반적인 기술로서 어플리케이션에서 사용하는 휘처들을 의미한다. 운영 환경 휘처는 제품계열의 컨텍스트(Context)를 포함하는 휘처들로서 컴퓨터 환경, 서로 다른 타입의 디바이스나 외부 시스템의 인터페이스들을 의미한다. 다른 제품들간의 공통적인 휘처들은 필수 휘처로(Mandatory feature), 상이한 휘처들은 옵션 휘처(Optional feature) 또는 대안 휘처(Alternative feature)로 모델링된다. 휘처 모델링에서는 휘처들간의 관계를 세 가지 종류로 표현한다. 구성(Composed-of) 관계는 한 휘처와 하위 휘처가 전체-부분의 관계에 있을 때 사용되어 지며, 일반화/특별화(Generalization/Specialization) 관계는 어떤 휘처가 하위 피처의 일반화에 해당될 경우 사용되어진다. 구현 (Implemented by) 관계는 특정 휘처가 다른 휘처를 구현하기 위해 꼭 필요할 경우에 사용된다. 세 가지 종류 이외에 상호 의존이나 배제관계를 표현하기 위해 복합 규칙이 있다.

외부로 보여지는 제품의 특성을 식별하기 위한 휘처 모델링은 도메인 계획, 휘처 식별, 휘처 조직화, 휘처 정제와 같이 4단계로 수행된다. 첫째, 도메인 계획 단계에서는 도메인을 선택하고 선택된 도메인 내의 범위를 분명히 한 후 도메인 분석팀을 조직하여 도메인 사전을 만드는 작업이 수행된다. 둘째, 휘처 식별 단계에서는 도메인 전문가나 서적, 사용자 매뉴얼, 설계서, 소스 프로그램 등과 같은 문서에서 얻은 도메인 지식을 추상화하는 작업이 수행된다. 셋째, 휘처 조직화 단계에서는 식별된 휘처들간의 관계를 분석하여 휘처 다이어그램을 만드는 작업이 수행된다. 넷째, 휘처 정제 단계에서는 휘처 식별 작업에 참여하지 않은 도메인 전문가나 매뉴얼, 프로그램, 설계서 등과 같은 도메인 산출물들과 함께 작성된 휘처 다이어그램을 점검하는 작업을 수행한다.

2.3 요구사항 추적성관련 기존 연구

요구사항 추적성과 관련된 연구들이 다양한 분야에서 수행되었다. 기본적인 요구사항 추적관련 기법은 주로 상호 참조 스키마(Cross referencing schemes)[6], 키 프레이즈 의존성(Key phrase dependencies)[7], 템플릿, 요구사항 추적 메트릭스, 하이퍼텍스트(Templates, Requirements traceability matrices, Hypertext)[8], 통합 문서(Integration documents)[9], 제약조건 네트워크(Constraint networks)[10] 등이 있다. 이러한 기술은 제어할 수 있

는 정보간의 상호연결성의 수와 지속적인 요구사항 변경에 직면했을 때 요구사항 추적을 유지보수할 수 있는가에 대한 정도에 따른 정보의 양이나 다양성에 있어서 차이가 있다. 이외에도 요구사항 추적 반자동 툴에 대한 연구도 있다[11, 12].

그러나 이러한 기법은 요구사항 추적을 수행을 지원하는 기술은 제공하지만 가치기반 소프트웨어 공학에서 제안하고 있는 가치나 비용에 대한 고려는 하고 있지 않다. Heindl and Biffl[13]은 프로젝트 관리자가 이해당사자 가치, 요구사항의 위험과 변경성, 추적 비용을 근간으로 요구사항 추적의 정밀도나 노력을 조정할 있도록 지원하는 가치기반 요구사항 추적(Value-based requirements tracing, VBRT)을 제안하고 이를 통한 사례연구를 실시하였다. 이 사례연구에 따르면 VBRT는 전체 요구사항 추적의 35% 노력으로 요구사항 추적 수행이 가능함을 확인하였다. VBRT가 추적 링크를 생성하는데 효과적으로 가치기반 소프트웨어공학을 고려하였으나 Heindl and Biffl[13]은 단순히 요구사항을 기능적인 계층 분류를 통해서 그룹핑하는 방법을 사용하고 있다.

반면, Riebisch[14]은 요구사항을 연결하는 모델을 설계하는데 중간 매개체의 개념을 적용하였다. 이는 휘처 모델이 추상화된 상이한 레벨간의 연결에 적절하고 적은 비용으로 요구사항 추적 링크를 생성하거나 유지보수할 수 있다고 제안하고 있다. 그러나 Riebisch[14]의 연구는 단순히 문제영역과 해결영역간의 연결만을 고려하였기 때문에 레벨을 세분화하거나 가치있는 추적 링크를 생성하는 부분이 미흡하다.

3. 휘처 기반의 요구사항 추적

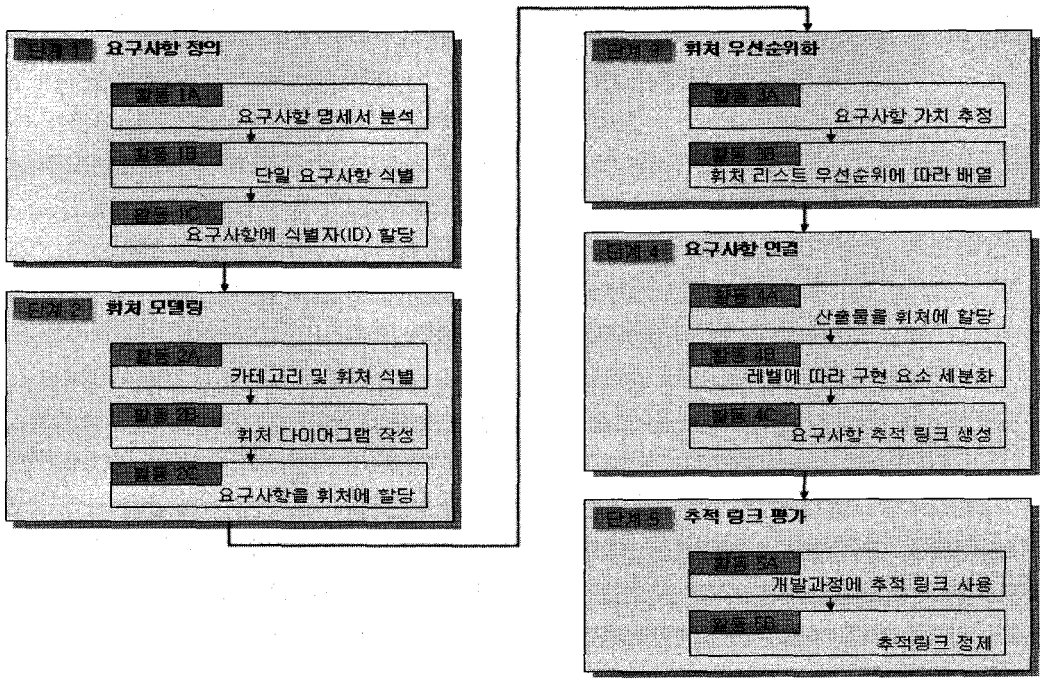
휘처 기반의 요구사항 추적의 목적은 비용과 노력을 고려하여 우선순위화된 요구사항을 기본으로 가치 있는 추적 링크를 식별하는 것이다. 세 3장에서는 휘처 기반의 요구사항 추적을 위한 메타 모델과 전체적인 프로세스를 기술한다.

3.1 프로세스와 활동

휘처 기반의 요구사항 추적 프로세스는 <그림 1>과 같이 요구사항 정의, 휘처 모델링, 휘처 우선순위화, 요구사항 연결, 추적 링크 평가의 5단계로 구성된다.

3.1.1 요구사항 정의

요구사항 정의 단계에서 사용자 요구사항은 다양한 종류의 산출물들과 연결되기 위하여 정규화된다. 대부분의 요구사항은 애매모호하게 작성되거나 추상적인 경우가 많다. 따라서 이 단계에서는 우선 사용자 요구사항 명세서를 분석한다. 이때, 유사한 소프트웨어 개발 프로젝트들을 통해서 수집된 일련의 요구사항으로부터 정의된 표준화된 용어집을 이용한다. 그리고 단일 요구사항을 식별한다. 대부분 요구사항 명세서에서는 한 문장이 다양한 의미를 포함한다. 이러한 문장을 기능적 또는 비기능적 접근방법에 의해서 단일 요구사항으로 나눈다. 요구사항 명세서의 모든 문장들이 단일 요구사항 사항으로 분류되면, 모든 단일 요구사항에 식별자를 할당한다. 이 단계의 결과물은 단일 요구사항 및 식별자 리스트이다.



〈그림 1〉 휘처 기반의 요구사항 추적 프로세스

3.1.2 휘처 모델링

휘처 모델링 단계에서는 휘처 다이어그램이 작성된다. 휘처 다이어그램을 작성하기 위해서 우선 목표 시스템의 카테고리와 각 카테고리 내 휘처들을 식별한다. 이때, 공통성과 가변성 개념을 가진 휘처들을 식별하기 위하여 각 카테고리에서 차이점을 찾아야 한다. 모든 휘처들이 식별된 후 다이어그램에 배치한다. 휘처 다이어그램을 작성할 때 휘처들 사이의 관계를 충분히 고려한다. 그리고 요구사항 정의 단계에서 식별된 단일 요구사항을 관련된 휘처에 할당한다. 이 단계의 결과물은 휘처 리스트와 휘처 다이어그램이다.

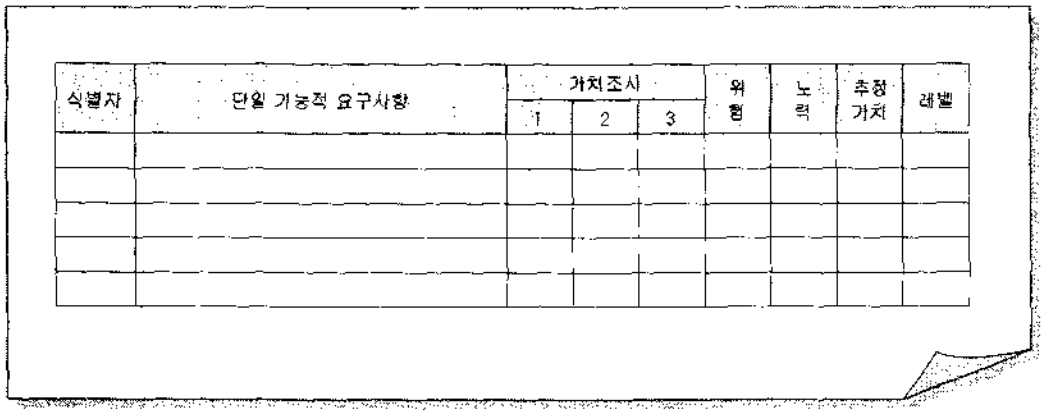
3.1.3 휘처 우선순위화

휘처 우선순위화 단계에서는 휘처의 중요성이 결정된다. 본 논문에서는 <표 1>과 같이 요구사항 추적 링크를 생성하기 위한 우선순위 레벨을 정의한다. 레벨이 올라갈수록 높은 세분화가 이루어진다.

〈표 1〉 우선순위 레벨 및 산출물 분류

우선순위	세분화	산출물 분류
1	낮음	Component
2	중간	Class
3	높음	Method

휘처 우선순위화는 <그림 2>의 가치분석 템플릿을 이용하여 각각 휘처에 대하여 요구사항의 가치, 요구사항의 위험, 추적링크 생성 및 유지보수와 관련한 노력 등에 대한 추정을 수행한다. 그리고 가치분석 매개변수에 대하여 요구사항 추적을 위한 상대적인 가중



식별자	단일 기능적 요구사항	가치조사			위험	노력	추정가치	레벨
		1	2	3				

〈그림 2〉 가치분석을 위한 템플릿

치를 부여 한다. 가중치 범위는 0.0~1.0이며, 요구사항 추적을 위해 해당 매개변수를 전혀 고려하지 않는다면 그것의 가중치는 0이다. 휘처의 우선순위 레벨은 각 가치분석 매개변수의 평가결과에 가중치를 곱한 것의 평균으로 산출한다. 각 휘처들은 추정된 결과에 근거하여 분류된다. 그리고 분류된 휘처들은 레벨에 따라 배열한다. 이렇게 하면 세분화된 추적 링크를 얻을 수 있다. 이 단계의 결과물은 우선순위가 할당된 휘처 리스트이다.

Vaule Analysis = (Rv, Rr, Te)

- 요구사항의 가치(Requirements value)는 이해당사자에게 있어서 요구사항에 대한 중요 정도를 의미한다. 높은 가치를 가지고 있는 요구사항의 경우 시스템의 핵심 기능을 담당하고 있고 향후 추적 정보가 중요하게 사용되므로 낮은 가치를 가지고 있는 요구사항의 경우보다 상세하게 추적링크를 생성할 필요가 있다.
- 요구사항의 위험(Requirements risk)은 향후 요구사항의 변경 가능성을 의미한다. 변경 가능성이 많은 요구사항은 변경영향 분석을 위해서 자주 추적을 수행해야 하므로 보다 상세한 요구사항 추적링크 생성이 필요하다.
- 추적 식별 및 유지보수의 노력(Traceability link efforts)은 요구사항 추적링크를 식별하고 유지보수하기 위해서 필요한 노력을 의미한다. 요구사항 추적링크를 상세화 한 경우 노력은 그 만큼 더 많이 필요하다. 그러나 가치가 낮은 요구사항의 경우 추적링크를 배제한다면 그 만큼 적은 추적링크가 생성되고 결과적으로 비용효과적으로 요구사항 추적이 가능하다.

3.1.4 요구사항 연결

요구사항 연결 단계에서는 모든 추적 링크가 생성된다. 우선 모든 산출물들이 관련된 휘처에 연결된다. 이때, 휘처가 요구사항과 산출물들을 연결하는 중간 매개체의 역할을 담당한다. 특히 구현 산출물 요소들은 우선순위 레벨에 따라 세분화되어 연결된다. 그러면, 요구사항 사항 추적 링크가 생성된다. 이 추적 링크는 요구사항, 휘처, 산출물들 사이

의 관계를 표현한다. 따라서 중요한 요구사항은 보통 요구사항보다 보다 자세히 추적이 가능하다. 이 단계의 결과물은 추적 링크 리스트이다.

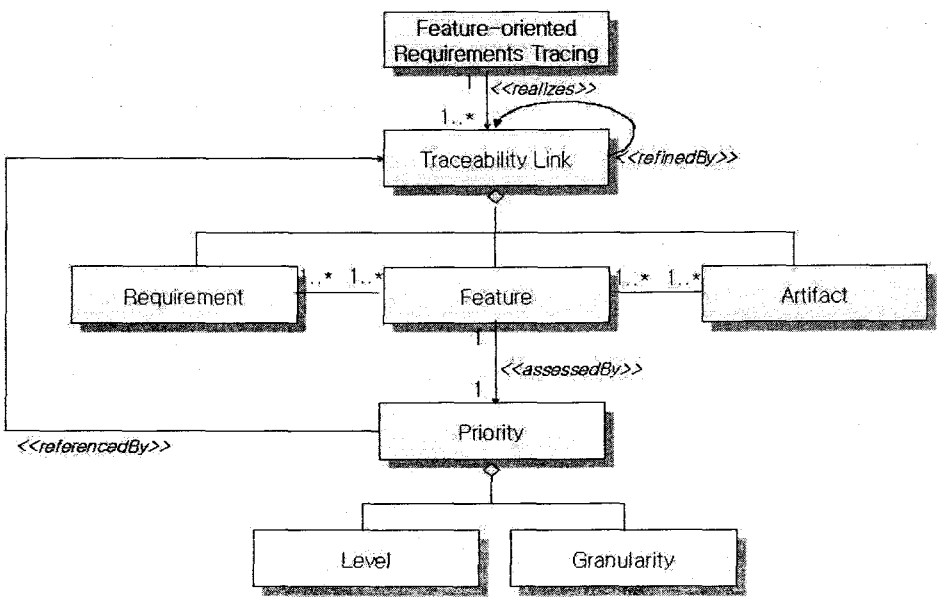
3.1.5 추적 링크 평가

추적 링크 평가 단계에서는 생성된 추적 링크가 소프트웨어 개발 과정에서 요구사항 변경 분석, 요구사항 충돌 분석, 요구사항 일관성 점검하는데 사용된 후 반복적으로 정제된다.

3.2 메타 모델

<그림 3>과 같이 휘처 기반의 요구사항 추적(FoRT)관련 메타 모델은 UML 형식으로 표현된다. 이 메타 모델에서는 휘처 기반의 요구사항 추적이 우선 사용자 요구사항이 입력되면, 이를 휘처로 그룹핑한 후 휘처와 구현 산출물들을 연결하기 위한 추적 링크를 생성하는 과정을 표현하고 있다.

휘처들은 각각 고유한 이름을 가지고 있으며 성능, 운영환경, 도메인 기술, 구현 기술과 같은 휘처 카테고리에 의해서 분류된다. 각 휘처 카테고리는 해당되는 요소들로 다시 세분화되며, 세분화된 휘처들은 구성, 일반화, 특별화, 구현, 옵션, 대안 관계에 의해서 연결된다. 이는 관련연구의 휘처 모델링을 근거로 한다. 이렇게 식별된 휘처들을 소프트웨어 개발 단계에서 생성된 산출물과 추적 링크를 통해서 연결된다. 이때, 각 휘처들은 가치, 위험, 노력 등의 기준에 의해 평가된 결과인 우선순위를 가지고 있는데, 우선순위 레벨에 따라 낮음, 중간, 높음으로 세분화가 되고 이는 가치 있는 추적 링크를 생성하기 위한 기준으로 사용된다. 즉, 추적링크는 이 우선순위를 참조하여 생성된다.



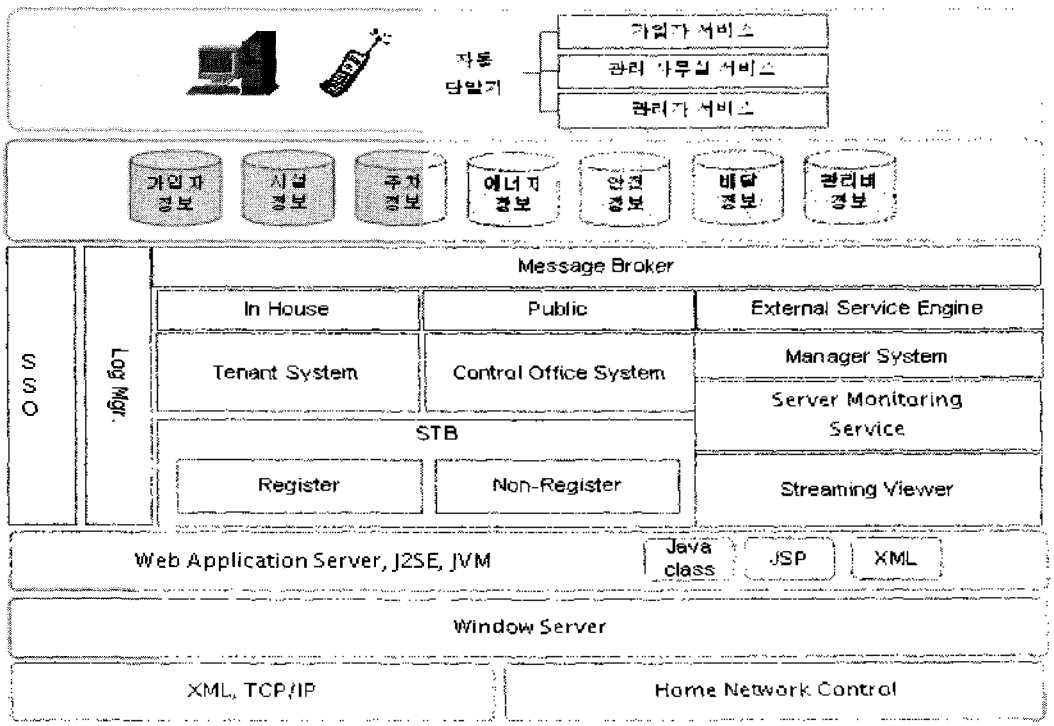
〈그림 3〉 휘처 기반의 요구사항 추적 메타 모델

4. 사례 연구

이 장에서는 제안된 휘처 기반의 요구사항 추적 프로세스와 결과물을 아파트 단지 내 홈 서비스를 통합하는 유비쿼터스 플랫폼에 적용한 사례를 보여준다. 아파트 유비쿼터스 플랫폼(AUP)은 아파트 단지 내 홈 네트워크 서비스를 효율적으로 관리 할 수 있는 입주민 지원서비스, 관리사무소 지원서비스, 시스템관리자 서비스 등을 통합한 아파트 단지 통합 솔루션이다. 사용자 인터페이스는 홈 단말, STB/TV, PC, 휴대용 단말기로 접속이 가능하도록 하였으며, 통합되는 시스템은 원격검침시스템, 관리비지원시스템, CCTV/DVR 시스템, 출동경비시스템, 무인경비시스템, 주차관제시스템, 설비제어시스템, 무인택배시스템 등이다. 단, 금번 사례에서는 휘처 모델링에 기능적 요구사항만을 고려하였으며, 추적 링크를 생성하는데 중복성 처리는 배제하였다.

4.1 단계 1 : 요구사항 정의

단계 1의 주요 목적은 사용자 요구사항 명세서를 분석하여 단일 요구사항을 식별하는 것이다. 아파트 유비쿼터스 플랫폼의 요구사항에는 홈 제어, 홈 정보관리 및 관리비, 시설 사용 예약, CCTV 이미지 운영을 위한 서비스 등이 포함된다. <그림 4>는 아파트 유비쿼터스 플랫폼의 주요 컴포넌트들의 구조를 설명하는 아키텍처이다. 아키텍처 구조의 프



〈그림 4〉 아파트 유비쿼터스 플랫폼 아키텍처

론트엔드는 입주민 서비스, 관리사무소 서비스, 시스템관리자 서비스로 분리되어 있고, 미들웨어인 웹 어플리케이션 서버(Web Application Server, WAS)는 톰캣을 사용하였으며 백엔드인 데이터베이스는 MS 2003 Server를 이용하였다. 그리고 AUP의 비즈니스 로직은 J2EE 기반의 JAVA를 이용하여 개발되었으며, 프리젠테이션 로직을 위해서는 JSP, XML 등을 사용하였다.

4.1.1 활동 1A : 요구사항 명세서 분석

텍스트 문서로 작성된 사용자 요구사항 명세서를 분석한다. 요구사항 명세서는 지면의 한계로 나타내지 않는다

4.1.2 활동 1B : 단일 요구사항 식별

요구사항 명세서를 분석 후, 단일 요구사항을 추출한 결과 124개 기능적 요구사항과 31개의 비기능적 요구사항을 얻었다.

4.1.3 활동 1C : 요구사항 식별자 할당

활동 1B에서 추출된 요구사항에 고유한 식별자를 부여하여 <표 2>와 같이 결과를 얻었다.

4.2 단계 2 : 휘처 모델링

단계 2의 주요 작업은 휘처 다이어그램을 작성하는 것이다.

4.2.1 활동 2A : 카테고리 및 휘처 식별

요구사항 정의 단계에서 추출된 요구사항 리스트로부터 휘처를 식별한 결과 성능 계층에 91개, 운영 환경 계층에 21개, 도메인 기술 계층에 휘처에 15개, 구현 기술 단계에 11개의 휘처를 얻었다. 요구사항에서 휘처를 식별할 때는 공통성과 가변성의 개념을 고려해야 한다. 그리고 식별된 휘처에 고유한 식별자를 할당한다.

4.2.2 활동 2B : 휘처 다이어그램 작성

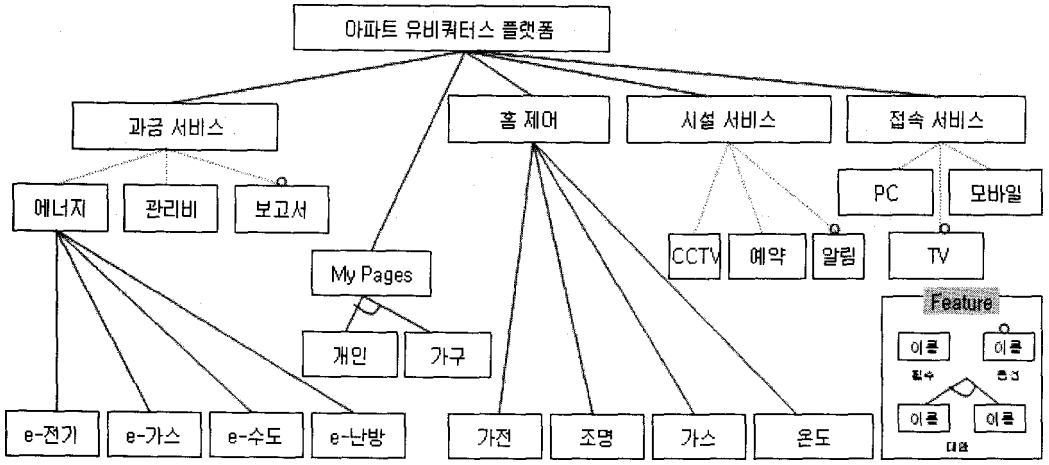
이렇게 식별된 각 휘처들의 관계를 고려하여 <그림 5>와 같은 휘처 다이어그램을 작성한다.

4.2.3 활동 2C : 요구사항을 휘처에 할당

활동 1C에서 추출된 요구사항과 이를 근거

〈표 2〉 기능적 요구사항과 식별자

식별자	단일 기능적 요구사항
RF_001	PC와 모바일 기기로 댁내 가스밸브 제어가 가능해야 한다.
RF_002	PC와 모바일 기기로 댁내 전기 스위치 제어가 가능해야 한다.
RF_003	PC와 모바일 기기로 댁내 난방 제어가 가능해야 한다.
RF_004	PC와 모바일 기기로 댁내 수도 제어가 가능해야 한다.
RF_005	방문자 이미지를 적어도 32개 이상 저장해야 한다.
RF_006	가전기기의 상태정보(알람, 제어신호 등)와 상호 연동할 수 있어야 한다.
.....	.....



〈그림 5〉 아파트 유비쿼터스 플랫폼의 휘처 다이어그램(일부)

로 추출된 휘처간의 매핑은 마이크로소프트사의 엑셀을 이용하여 요구사항 메트릭스를 작성한다.

4.3 단계 3 : 휘처 우선순위화

단계 3의 주요 작업은 식별된 휘처의 중요도를 결정하는 것이다.

4.3.1 활동 3A : 요구사항 가치 추정

모든 이해당사자는 단계 1에서 식별된 결과물에 대하여 가치, 위험, 노력 등에 따라 요구사항의 가치를 추정한다.

4.3.2 활동 3B : 우선순위에 따라 휘처 배열

활동 3A의 결과를 종합한 후 요구사항의 우선순위 레벨은 각 항목의 평균치로 산출한다. 그 결과는 23% 높음(우선순위 레벨 3), 40% 중간(우선순위 레벨 2), 37% 낮음(우선순위 레벨 1)이다. 이를 참조하여 휘처와 구현 산출물을 연결할 때 세분화 정도가 결정된다.

4.4 단계 4 : 요구사항 연결

단계 4의 주요 작업은 요구사항 추적 링크를 생성하는 것이다.

4.4.1 활동 4A : 산출물을 휘처에 연결

식별된 휘처에 해당되는 소스 및 구현 프로그램, 설계서, 테스트 케이스 등과 같은 산출물을 연결한다.

4.4.2 활동 4B : 우선순위에 따라 휘처 세분화

휘처에 연결된 구현 산출물을 우선순위에 따라 컴포넌트, 클래스, 메소드와 같이 세분화하여 다시 연결한다. 아파트 유비쿼터스 플랫폼에서는 11개 컴포넌트, 1002개 클래스, 3009개의 메소드를 식별하였다.

4.4.3 활동 4C : 요구사항 추적 링크 생성

활동 4B의 작업 결과, 요구사항, 휘처, 산출물을 연결하는 1004개의 추적 링크를 얻었다.

〈표 3〉 요구사항 추적 링크

추적링크 식별자	요구사항 식별자	회처 식별자	구현 산출물		
			Component	Class	Method
T001	RF_008	F001	Energy	EngergyIsepection.class	getIatHousehold()
T002	RF_008	F001	Energy	EngergyIsepection.class	getEnergyElectricty()
T003	RF_008	F001	Energy	EngergyIsepection.class	getMeterElectricty()
…	…	…	…	…	…
T052	RF_001	F015	HomeControl	GasRemote.class	requestConnection()
T053	RF_001	F015	HomeControl	GasRemote.class	setGasOn()
…	…	…	…	…	…
T121	RF_030	F017	FacilitityBook	-	-
…	…	…	…	…	…

5. 평 가

관련연구에서 기술한 요구사항 추적 링크와 관련된 기존 연구와 본 논문에서 제안하는 회처 기반 요구사항 추적 기법과의 비교를 위하여 「Guide to the Software Engineering Body of Knowledge, SWEBOK」에서 제시하고 있는 요구사항 관리와 관련된 주요 항목과 프로젝트 수행중 효과적인 요구사항 추적을 위해 필요한 항목을 평가항목으로 선정하였다.

- 잘 정의된 프로세스 : 요구사항 추적 링크를 식별하거나 생성하기 위한 프로세스가 잘 정의되었는지에 대한 평가 기준
- 가치기반 추적, 노력 절감 : 추적 링크를 생성하거나 유지보수하기 위하여 비용이나 노력을 절감하기 위한 가치분석을 고려하였는지에 대한 평가 기준
- 회처 모델링, 가변성 고려 : 요구사항과 다양한 산출물들을 연결하거나 요구사항 변경을 효과적으로 처리하기 위하여 중간 매개체를 활용하는지에 대한 평가 기준
- 툴 제공 : 추적 링크 관리를 위한 자동화를 제공하는지에 대한 평가 기준

〈표 4〉 기존 연구들과 비교 평가

평가항목 \ 기법	Egyed[11]	Pinheiro[12]	Heindl[13]	Riebisch[14]	본 논문의 기법
잘 정의된 프로세스	△	△	△	X	○
가치 기반 추적	X	X	△	X	○
회처 모델링	X	X	X	○	○
가변성 고려	X	X	X	△	○
노력 절감	Medium	low	High	Medium	High
툴 지원	○	○	○	△	X

주) ○ : 제공, △ : 부분 제공, X : 미흡.

<표 4>와 같이 기존 연구들은 추적링크를 생성하거나 툴로써 관리하기 위한 단순 기법을 제시하고 있는 반면 휘처 기반의 요구사항 추적기법은 비용과 노력을 절감하면서 효과적으로 요구사항을 추적할 수 있는 방법을 보여준다. 또한, 대부분의 기존 방법이 가치중립적으로 수행된 반면 본 논문에서는 가치분석을 수행하고 동 결과를 추적링크를 생성하는데 반영토록 하고, 휘처를 중간 매개체로 활용함으로써 요구사항과 산출물들을 연결할 때 발생하는 추상성에 대한 이슈를 해결할 수 있도록 도움을 줄 수 있다.

6. 결론 및 향후 연구

본 논문에서는 요구사항 추적 링크를 생성하거나 운영할 때 많은 노력이 필요한 문제를 해결하기 위하여 휘처 기반의 요구사항 추적 기법을 제안하였으며, 이의 실현성을 검증하기 위하여 아파트 유비쿼터스 플랫폼에서 사례연구를 수행하였다. 동 기법은 가치분석과 휘처를 중간 매개체로 활용한다. 가치분석의 목적은 우선순위가 부여된 요구사항으로부터 가치 있는 추적 링크를 생성하는 것이다. 이것은 요구사항과 구현 산출물들 사이의 연결을 위하여 중간 매개체로 휘처를 두고, 해당 휘처에 중요성을 부여하여 세분화를 근거로 실현된다. 이것은 우선 우선순위가 높게 평가된 요구사항의 경우 추적링크를 상세하게 만들고 그렇지 않을 경우 최소의 추적링크만을 생성하므로 비용·효과적으로 보다 적은 수량의 요구사항 추적 링크를 생성할 수 있고 휘처를 매개체로 활용함에 따라 관련 지식이 없더라도 쉽게 요구사항부터 산출물에 이르는 검증이 용이하다. 그리고 산출물이 많은 대용량 시스템의 경우 보다 쉽게 유지보수가 가능하며, 연결된 산출물들 사이의 관계를 이해하고 기억하기가 용이하다. 또한, 재공학, CBD, 컴포넌트 연관관계 점검, 재사용 컴포넌트 작성 등을 유용하게 지원할 수 있다.

향후 본 연구는 자동화와 같은 시스템적인 접근으로 확장이 필요하다. 또한, 대량의 산출물이 작성되는 복잡한 소프트웨어 개발 프로젝트의 요구사항 변경이나 일관성 검증에 활용하여 심도 깊은 실무적인 분석이 수행되어야 한다.

참 고 문 헌

[1] Karl E. Wiegers, Software Requirements 2/E, Microsoft Press, 2003.

[2] Biffl, S., Aurum, A., and Boehm, B. W., et al, Value-based Software Engineering, Springer Verlag, 2005.

[3] Grant Zemont, Towards Value-Based Requirements Traceability, thesis, DePaul University, Chicago Illinois, 2005.

[4] Kang, K., Kim, S., Lee, J., Kim, K., Shin E., Huh, M., "FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures," Annals of Software Engineering, 1998, pp. 143-168

[5] Kwanwoo Lee, Kyo C. Kang, and Jae-Joon Lee, "Concepts and Guidelines of Feature Modeling for Product Line Software Engineering," Proceedings of the Seventh Reuse Conference (ICSR7), Austin, U.S.A., Apr.15-19, 2002, Springer Lecture Notes in Computer Science Vol. 2319, 2002, pp. 62-77

[6] Evans, M. W., "The Software Factory," John Wiley and Sons, 1989.

[7] Jackson, J., "A Keyphrase Based Traceability Scheme," IEE Colloquium on Tools and Techniques for Maintaining Traceability during Design, 1991, pp. 2-1-2/4.

[8] Kaindl, H., "The Missing Link in Requirements Engineering," ACM SigSoft Software Engineering Notes, Vol. 18, No. 2, 1993, pp. 30-39.

[9] Lefering, M., "An Incremental Integration Tool between Requirements Engineering and Programming in the Large," Proceedings of the IEEE International Symposium on Requirements Engineering, San Diego, California, Jan. 1993, pp. 82-89.

[10] Bowen, J., O'Grady, P., and Smith, L. "A Constraint Programming Language for Life-cycle Engineering," Artificial Intelligence in Engineering, Vol. 5, No. 4, 1990, pp. 206-220.

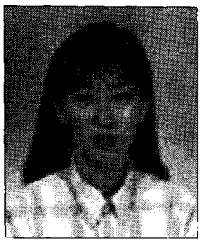
[11] Egyed, A. "A Scenario-Driven Approach to Traceability," Proceedings of the 23rd International Conference on Software Engineering (ICSE), Toronto, Canada, May 2001, pp. 123-132

[12] Pinheiro, F. A. C., Goguen, J. A. "An Object-Oriented Tool for Tracing Requirements," IEEE Software, Vol. 13, No.2, 1996, pp. 52-64.

[13] Matthias Heindl, Stefan Biffl, "A Case Study on Value-based Requirements Tracing," Proceedings of the European Software Engineering Conference and Foundations of Software Engineering (ESEC/FSE), 2005, pp. 60-69.

[14] Matthias Riebisch, "Supporting Evolutionary Development by Feature Models and Traceability Links," Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS '04), May 2004, pp. 370-377.

저 자 소 개



안상입 (E-mail : siahn69@ssu.ac.kr)
1992년 숭실대학교 전자계산학과 졸업
2004년 연세대학교 공학대학원 컴퓨터공학 공학석사
2005년 ~ 현재 숭실대학교 대학원 컴퓨터공학과 박사과정
관심분야 요구공학, 소프트웨어개발방법론, 품질보증

정기원 (E-mail : chong@ssu.ac.kr)
1967년 서울대학교 공과대학 전기공학과 졸업
1981년 미국 알라바마 주립대학(헌츠빌) 전산학과 공학석사
1983년 미국 텍사스 주립대학(알링턴) 전산학과 공학박사
1990년 ~ 현재 숭실대학교 컴퓨터학부 정교수
관심분야 소프트웨어 개발 프로세스, 방법론, 모델링, 실시간 응용, 전자거래, 정보시스템 개발 및 평가