전사적 프로젝트의 Use Case 모델링을 위한 실무 지침

김민선, 김수동
숭실대학교 컴퓨터학과
e-mail : mskim@selab.soongsil.ac.kr

Practical Instructions for Modeling Use Cases of Enterprise Project

Min-Seon Kim, Soo Dong Kim
Dept. of Computing, Soongsil University

요 약

OMG 의 공식 표준으로 자리 잡은 UML 은 9 종류의 기본 모델들을 기초로 하여 다양한 관점에서 대상 소프트웨어 시스템을 모델링 하도록 지원해주고 있다. 특히 Use Case 의 경우 시스템 사용자와의 대화의 수단으로 시스템이 최종 사용자에게 제공하는 서비스들을 잘 표현하고 있다. 그러나 객체 모델이나 순차도 등의 다른 모델들에 비해 설명이 보다 모호하고, 정형화 되어 있지 않아서, 특히 전사적인 규모의 실무에 Use Case 모델 을 적용할 경우, 여러 가지 어려움들이 있다. 개념 단계와 상세 단계에서의 일관성 유지 문제, 하부 시스템으로 나누어 작업 시, 《include》나 《extend》 관계의 Use Case 에 대한 배치, 전체나 일부에서 공통으로 사용되는 《include》 Use Case 의 표현 문제 등이 그것들이다. 본 논문에서는 실무 적용 시 부딪힐 수 있는 Use Case 모델링의 그러한 문제점들에 대한 실무적인 지침들을 사례 연구와 함께 제안하고자 한다.

1. 서론

UML 이 학계나 업계의 모델링 표준으로 자리 잡으면서 분석과 설계 시 유용한 모델링 도구로서 자리잡아가고 있다. UML[1]의 기본 모델(model)들은 구조적모델, 동적 모델, 구조적 모델로 나뉠 수 있다. 구조적 모델에는 클래스 모델 과 객체도(Object Diagram)가 있으며 동적 모델에는 Use Case 모델 , 순차도(Sequence Diagram), 협력도(Collaboration Diagram), 상태도(Statechart Diagram) 및 활동도(Activity Diagram)가 포함된다. 구조적 모델은 컴포넌트 모델과 배치도(Deployment Diagram)를 포함한다.

이 중 Use Case 모델은 분석 초기 단계에서 사용자의 요구사항을 모델링 하는 도구로서, 구축 대상 시스템이 최종 사용자에게 제공하는 서비스들을 사용자의 용어를 기반으로 표현된다. 복잡한 표기법 없이 엑터(Actor)와 Use Case 만의 관계를 보여줌으로써 간결하게 시스템의 기능을 파악할 수 있도록 지원하지만, 서술적인 묘사로 다른 모델들에 비해 모호하다는 단점을 가지고 있다. 특히 전사적인 프로젝트를 수행할 때는 전체 시스템에 대한 초기 버전과 하부 시스템 별 상세 버전으로 나뉘어 작업되며, Use Case 의 복잡도도 증가하여 관리상의 지침들이 요구된다고 하겠다.

본 논문의 2 장에서는 관련 연구로서 UML 에서 지원하는 모델의 확장 기법들과 Use Case 모델 표기법 및 이미 정의된 스테레오타입(Stereotype)들을 살펴본다. 3 장에서는 전사적 프로젝트에 Use Case 모델 을 적용 시 부딪힐 수 있는 문제점들을 정리하고, 각 문제점들에 대한 지침들을 제안하고, 사례를 보인다. 마지막으로 4 장에서 결론을 내리고자 한다

[1] 본 논문에서는 UML 버전 1.3 을 기반으로 한다.

2. 관련 연구

2.1. UML 의 확장 기법

UML 의 확장은 스테레오 타입, 꼬리표값(Tagged Value), 제약조건(Constraints)의 3 가지 기법으로 지원된다. 스테레오타입이란 UML 의 용어(Vocabulary)를 확장하는 기법으로, 기존의 구성 요소들과 유사한 새로운 구성 요소를 만들 수 있도록 한다. 그러나 스테레오타입으로 생성된 구성 요소는 생성된 영역 내에 한해 유효하다. 이중꺽쇄, "《", "》" 사이에 스테레오타입 이름을 지정한 후, 기존의 구성 요소명 위에 위치시키는 것으로 표기되며, 특정 스테레오타입을 위해 아이콘(Icon)을 지정하여 표현할 수도 있다.

꼬리표값은 UML 구성 요소들의 속성(Properties)을 확장하는 기법으로 구성 요소 명세 내에서 새로운 정보를 생성하도록 한다. 중괄호 "{", "}" 사이의 문자열의 형태로 표현되며 다른 구성 요소명 아래에 위치한다.

제약조건은 UML 의 구성 요소들의 문법(Semantics)을 확장하는 기법이다. 기존의 규칙을 수정하거나 새로운 규칙을 생성하도록 하며, 꼬리표값과 마찬가지로 중괄호 사이에 관련 규칙을 명시한다. 그리고 그 제약조건이 적용되는 관련 구성 요소 근처에 위치시키거나 의존 관계(Dependency Relationship)를 이용하여 해당 구성 요소와 연결시킨다.

2.2. UML 의 기정의 표준 요소

UML 버전 1.3 에서는 위의 3 가지 확장 기법을 이용하여 일반적으로 적용될 표준 요소들을 정의하고 있다. 표 1은 Use Case 와 관련된 표준 요소들이다. 표 1에 적용된 핵심어들은 모두 스테레오타입이다[1].

표 1. Use Case 과 관련된 표준 요소

핵심어	적용 구성 요소	의미
extend	dependency	대상 Use Case 가 소스 Use Case 의 기능을 주어진 확장 지점에서 확장시킨다.
include	dependency	소스 Use Case 가 소스에 의해 명시된 지점에서 다른 Use Case 의 기능과 명시적으로 협력함을 의미한다.
refine	dependency	소스가 대상에 비해 추상화 수준이 미세함을 의미한다.
Use	dependency	소스 구성 요소의 문법이 대상의 공용 부분의 문법에 의존적임을 의미한다.

3. Use Case 모델링 지침

3.1. Use Case 모델링의 문제점

전사적 규모의 프로젝트에 Use Case 모델을 적용시 다음의 문제점들에 부딪힐 수 있다.

첫째, 요구사항명세서 및 도메인 분석과정의 결과를 토대로 Use Case 를 추출하는 과정에서 가장 논란이 되는 부분으로서 적정 Use Case 의 크기를 정하고, 각 단계의 모델을 구성하는 Use Case 들이 균일한 크기를 유지하도록 시스템에서 제공해야 하는 서비스들을 분류하는 기준을 정하는 일이다. Use Case 모델은 그것이 초기 단계 모델인가, 상세 단계 모델인가에 따라 적정 Use Case 의 크기를 달리할 것이다.

둘째, 모델 간의 일관성이 유지되도록 하는 일이다. 상세 단계 분석과정을 통해 초기 단계의 Use Case 들을 보다 작고 구체적인 Use Case 단위로 분해하는 작업은 전 하는 사적 규모의 프로젝트에 있어 반드시 거쳐야 단계이다. 그 작업에서 초기 단계의 Use Case 모델과 상세 단계의 모델 사이의, 또 분할된 각 Use Case 간의 일관성이 요구된다.

3.2. 상세 단계 Use Case 모델로의 분할 지침

전사적 프로젝트의 Use Case 모델링 시의 가이드로서 다음과 같은 몇 가지 지침을 제안한다.

■ 개념 단계

① 컨텍스트(context) 레벨의 Use Case 모델에서는 서브시스템(subsystem) 단위의 Use Case 를 도출한다.

② 초기 단계의 Use Cases 는 유사한 데이터를 조작하는 단일 혹은 다수의 서비스들을 포함하도록 한다.

■ 상세 단계

① 서브시스템별로 식별번호를 관리하고 서브시스템 단위로 분할된 모델을 작성한다. Use Case 는 하나의 서브시스템에 속하는 것을 기본으로 하지만 서브시스템 내의 Use Case 가 <<include>>나 <<extend>> 의존 관계를 맺는 Use Case 는 다른 서브시스템의 Use Case 일지라도 나타내 준다. 이 경우 두 서브시스템의 식별을 용이하게 하도록 스테레오타입을 정의하여 명시한다던가 모델 상에서 각 서브시스템을 구성하는 Use Case 의 색상을 다르게 지정한다.

② 상세 단계에서는 트랜잭션(transaction) 단위로 Use Case 를 추출한다. 단, 예외적으로 쿼리(qurery) 서비스의 경우는 별개의 Use Case 로 분리해내지 않고 기존의 Use Case 에 통합하는 것이 가능하다.

③ 상세 단계 모델링을 진행하는 과정에서 하나 이상의 서브시스템에서 공통적으로 사용하는 Use Case 가 추출될 수 있는데, 그 경우 해당 Use Case 에 <<global>>이라는 스테레오타입을 정의하여 적용시킨다. 또한 단일 서브시스템 내에서 특정 Use Case 에로의 <<include>> 혹은 <<extend>> 의존 관계가

두 번 이상 나타난다면, 그 특정 Use Case 에는 <<local common>>이라는 스테레오타입을 정의한다.

④ <<extend>> 의존 관계로 도출되는 Use Case 는 필요하다면 별개의 서브시스템으로 분리할 수 있다. 그러나 <<include>> 의존 관계로 분리되는 상세 수준 Use Case 의 경우에는 여러 서브시스템에서 공통적으로 요구되는 서비스(<<global>> 및 <<local common>> Use Case)에 제한적으로 독립된 서브시스템으로 분리하는 것이 가능하다. <<global>> Use Case 의 경우 그것이 속해있는 메인(main) 서브시스템을 정하는 것이 곤란하다면 해당 Use Case 를 독립된 서브시스템으로 분리한다.

4. 사례 연구

4.1. 사례 연구 도메인 정의

은행 도메인 중 계좌 관리와 수신 관리 업무의 모델링에 실무 지침들을 적용한 예이다.

계좌관리 (Account Management) 업무는 은행에서 다루는 금융 상품을 관리하고 신규 계좌 개설과 기존 계좌의 해지를 담당한다. 그리고 계좌 원장과 고객별 계좌 목록 등의 조회 업무도 포함된다.

수신관리 (Deposit Management) 업무는 개인이나 기업이 그들의 자금을 은행에 저축 또는 출납대행 등의 목적으로 예금을 하거나 필요한 때 언제든지 환급을 받을 수 있으며 은행이 그 자금을 합리적으로 운용하여 얻어진 이익의 일부를 이자로 지급 받을 수 있는 업무이다.

4.2. 상세 단계 Use Case 로의 분할

■ 초기단계

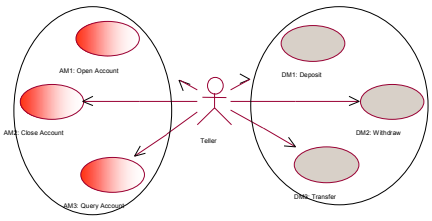

그림 1. 초기 단계 사용사례도

■ 상세단계

□ 계좌관리(AM) 서브시스템의 분할

신규 계좌 개설(AM1) 시 계좌입금(DM1)이 일반적으로 같이 발생한다. 계좌 해지(AM2) 시에는 해당 계좌에 약정된 규칙에 따라 이자 계산 작업을 하게 되며, 계좌출금(DM3)이 반드시 수행되고, 때로는 계좌이체(DM3)가 발생한다. 모든 작업 시 Journal 에 작업 내용을 남긴다(SM2).

계좌관리 서브시스템을 구성하는 Use Case(AMx)들과 <<include>>및 <<extend>> 관계에 있는 수신관리 및 시스템관리 서브시스템에 속한 Use Case 들까지 나타내주었다. 그림 2. 계좌 관리 부문의 분할와 같이 서브시스템 별로 식별번호를 부여하고 표기 색상을 달리하였다.

계좌관리:Amx 수신관리:DMx 시스템관리:SMx

그림 3. 수신관리 부문의 분할, 그림 4. 시스템관리 부문의 분할의 수신관리, 시스템관리 서브시스템에 속한 Use Case 이면서 계좌관리 서브시스템의 Use Case 와 <<include>> 및 <<extend>> 의존 관계로 모델 상에 존재하는 DM1, DM2, DM3, DM4, SM2 Use Case 에는 <<global>>의 스테레오 타입을 명시하였다.

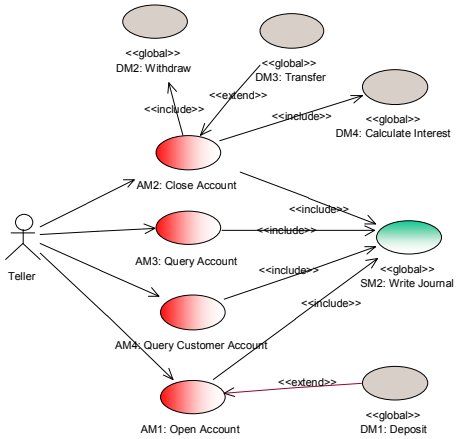

그림 2. 계좌 관리 부문의 분할

□ 수신관리 부문의 분할

입금(DM1), 출금(DM2), 이체(DM3),과 같은 고객의 수신업무와 계좌 조회를 서비스 해주는 서브시스템으로 관련된 트랜잭션이 발생할 경우 그 결과는 Bookkeeping 의 해당 계정에 반영된다. 계좌관리 업무에서와 같이 각 작업의 로그(Log)를 기록하는 저널링(Journaling) 작업이 포함된다 (SM2).

새로이 도출된 상세 수준 Use Case 인 DM5, DM7 가 수신관리 서브시스템 내에서 DM1 과도 <<include>>관계에 있고 DM2 와도 같은 관계에 있으므로 <<local common>> 이란 스테레오 타입을 표기하였다.

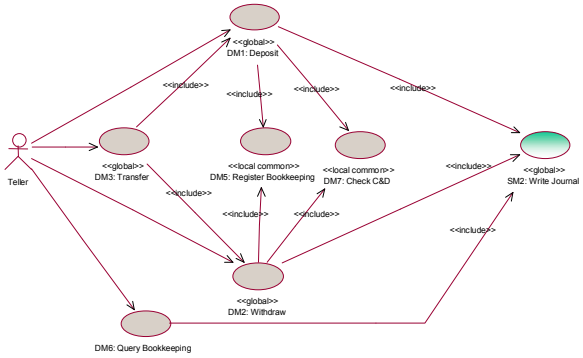

그림 3. 수신관리 부문의 분할

□ 시스템관리(SM) 서브시스템의 분할

전체 업무에 공통적으로 적용되는 관리 기능을 담당한다. 작업을 시작하기 전에 Teller 의 신분을 확인하는 작업과 각 작업의 로그(Log)를 기록하는

저널링(Journaling) 작업이 포함된다

계좌관리 및 수신관리 서브시스템의 상세수준 Use Case 모델을 작성하는 과정에서 도출된 <<global>> Use Case 인 SM2 는 모든 서브시스템과 <<include>> 관계를 지니므로 SM1 의 서비스를 추가로 추출하여 독립된 서브시스템으로 분리하였다.

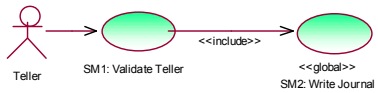

그림 4. 시스템관리 부문의 분할

5. 결론 및 향후 연구 과제

본 논문에서는 은행 시스템을 사례 연구 도메인으로 하여, 전사적인 시스템 개발 시 적용할 수 있는 Use Case 모델링의 실무적인 지침들을 제안하였다. 본 논문에서 제시된 방법은 사례 연구를 통해 얻어진 기법들이지만 다른 다양한 개발 영역과 환경에도 적용이 가능하며, 본 논문에서 제시된 방법 외에 다른 지침들도 요구될 수 있다.

본 논문에서는 가장 모델링 지침이 부족하고 정형화 수준이 낮은 Use Case 모델에 적용 가능한 지침들을 제안하였으며. 향후 연구 방향으로는 Use Case 모델과 UML 의 타 모델들 간의 일관성 보장을 위한 실무적인 지침 개발을 들 수 있다.

참고문헌

[1] Grady Booch, James Rumbaugh, and Ivar Jacobson, The Unified Modeling Language User Guide, Addison – Wesley,1999.

[2] Martin Fowler with Kendall Scott, UML Distilled, Addison –Wesley,1997

[3] Richard C.Leee, UML and C++, William M.Tepfenhart AT&T

[4] Rational Software, UML Notation Guide, 1997

[5] Rational Software, UML Semantics, 1997