소프트웨어공학

[소프트웨어 공학] 모델링과 UML, 유스케이스 다이어그램

ljh0326s
2017. 5. 9. 19:29

소프트웨어 개발과 모델링

개발자가 소프트웨어를 개발하는 이유는 고객의 문제를 해결하기 위해서이다. 따라서 소프트 웨어 개발에 있어 가장 우선적으로 해야 할 일은 문제를 이해하는 것인데, 이 때 문제를 잘 이해하기 위해서 하는것이 모델링이다.

언제 이 모델링을 해줄까? 애자일기법이든 폭포수 모델이든 분석, 설계, 구현, 테스트과정을 거치며 프로그램 언어를 이용해서 실제로 개발하는 단계는 구현이란 단계이다. 즉 모델링은 이 구현단계 전에서 해야하며 이 때 사용하는 언어가 바로 UML이다.

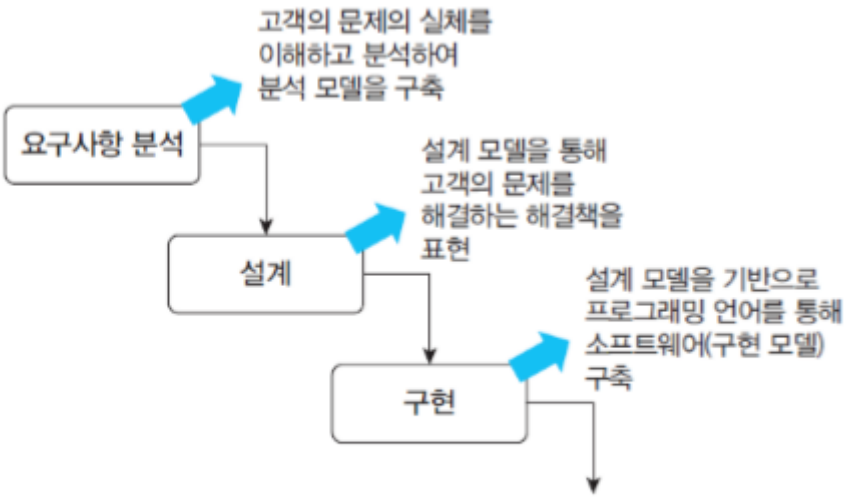


그림 2-1 소프트웨어 개발과 모델

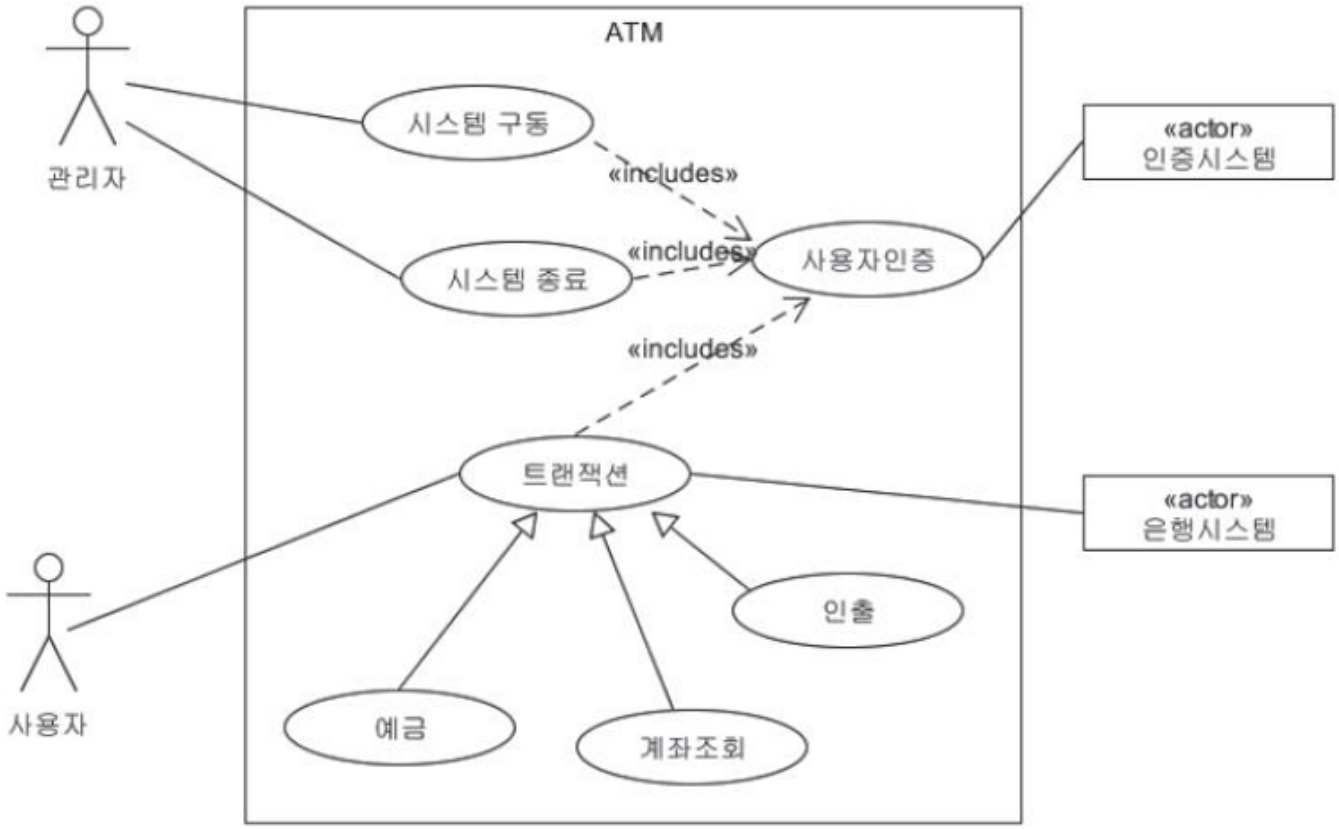
UML (Unified Modeling Language)

은 시스템의 구조와 동작은 표현하는 13개의 다이어그램을 제공해준다. 아래는 이 다이어그램들을 간략하게 정리한 표이며 우리는 대표적으로 많이 사용되는 유스케이스 다이어그램, 클래스 다이어그램, 순차 다이어그램, 패키지 다이어그램에 대해 다룰 것 이다.

참고 : UML의 장점은 분석 모델과 설계 모델을 표현하는 방식이 다르지 않다는 것이다.

분류	다이어그램 유형		목적
구조 다이어그램 (structure diagram)	클래스 다이어그램 (class diagram)		시스템을 구성하는 클래스들 사이의 관계를 표현한다.
	객체 다이어그램 (object diagram)		객체 정보를 보여준다.
	복합체 구조 다이어그램 (composite structure diagram)		복합 구조의 클래스와 컴포넌트 내부 구조를 표현한다.
	배치 다이어그램 (deployment diagram)		소프트웨어, 하드웨어, 네트워크를 포함한 실행 시스템의 물리 구조를 표현한다.
	컴포넌트 다이어그램 (component diagram)		컴포넌트 구조 사이의 관계를 표현한다.
	패키지 다이어그램 (package diagram)		클래스나 유즈 케이스 등을 포함한 여러 모델 요소들을 그룹화해 패키지를 구성하고 패키지들 사이의 관계를 표현한다.
행위 다이어그램 (behavior diagram)	활동 다이어그램 (activity diagram)		업무 처리 과정이나 연산이 수행되는 과정을 표현한다.
	상태 머신 다이어그램 (state machine diagram)		객체의 생명주기를 표현한다.
	유즈 케이스 다이어그램 (use case diagram)		사용자 관점에서 시스템 행위를 표현한다.
	상호작용 다이어그램 (interaction diagram)	순차 다이어그램 (sequence diagram)	시간 흐름에 따른 객체 사이의 상호작용을 표현한다.
		상호작용 개요 다이어그램 (interaction overview diagram)	여러 상호작용 다이어그램 사이의 제어 흐름을 표현한다.
		통신 다이어그램 (communication diagram)	객체 사이의 관계를 중심으로 상호작용을 표현한다.
		타이밍 다이어그램 (timing diagram)	객체 상태 변화와 시간 제약을 명시적으로 표현한다.

시스템에서 제공해야하는 기능이나 서비스를 명세하는 단계로 상용자와 시스템 사이의 상호작용을 보여준다.
아래 그림은 유스케이스 다이어 그램의 한 예이다. 한번 살펴보자.



유스케이스 다이어그램의 구성요소

위 예제를 통해 유스케이스를 구성하는 요소들을 알아보자

시스템 범위(scope)

자 위의 예제를 살펴봤을때 많은 사람들이 아 위 유스케이스 다이어그램이 ATM시스템을 모델링하고 있는 중이다. 라는 것을 알 수 있을 것이다. 왜냐하면 커다란 네모에 ATM으로 쓰여져 있기 때문이다. 즉 우리가 개발하고자 하는 시스템은 사각형으로 표시한다는 것을 자연스럽게 알 수 있다.

유스케이스(usecase)

자 그러면 사각형 안에있는 동그라미는 뭘까? 이것이 바로 유스케이스다. 시스템이 어떤 서비스or 기능을 제공하는지를 명세해 주는 것으로 타원형으로 표시하며 유스케이스의 이름은 상호작용을

5

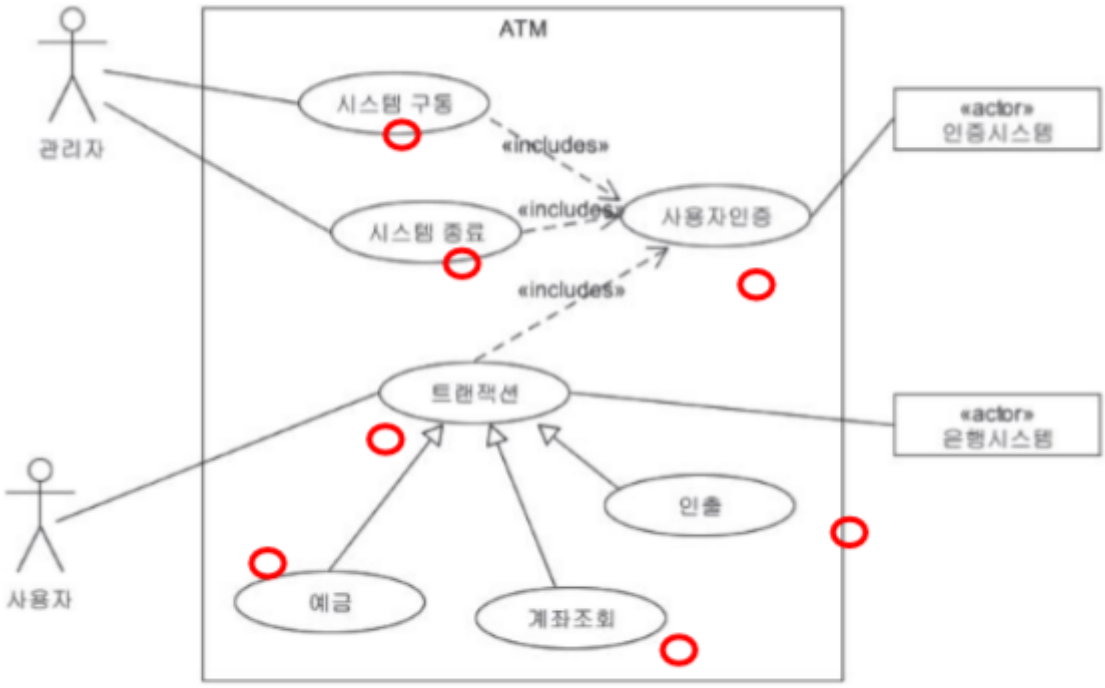


그림 2-2 ATM 시스템 유스케이스 다이어그램 예제

액터(actor)

액터는 시스템 외부에 존재하며 시스템과 상호작용을 하는 모든 것들을 나타낸다. 시스템을 사용하게 될 사람은 물론이고, 다른 외부에 있는 시스템도 포함된다. 액터에는 2가지 종류가 있다.

1. 프라이머리 액터(primary actor) : 시스템을 사용함으로서 이득을 얻는 엑터로 보통 외부 객체가 사람일때를 말하며 졸라맨을 이용해 표기한다. 보통 시스템의 왼쪽에 그려준다. 위 예제에서는 ATM을 이용하는 액터가 사용자와 관리자가 있다는 얘기다.

2. 세컨더리 액터(secondary actor) : 프라이머리 액터가 이득을 얻기위해 도움을 주는 액터로 보통 외부 시스템을 의미하며 박스에<<actor>>를 입력하여 표기한다. 보통 시스템의 오른쪽에 그려준다. 위 예제에서는 ATM을 도와주는 외부 시스템으로 인증시스템과 은행시스템이 있다는 얘기다.

참고 : 액터이름은 구체적이면 안된다. 예를들어 액터이름을 홍길동으로하면, 홍길동만 ATM을 사용한다는 의미가 되며 이런 상황을 방지하기위해 보통 그 조직이 수행하는 역할 이름을 사용해준다.

5

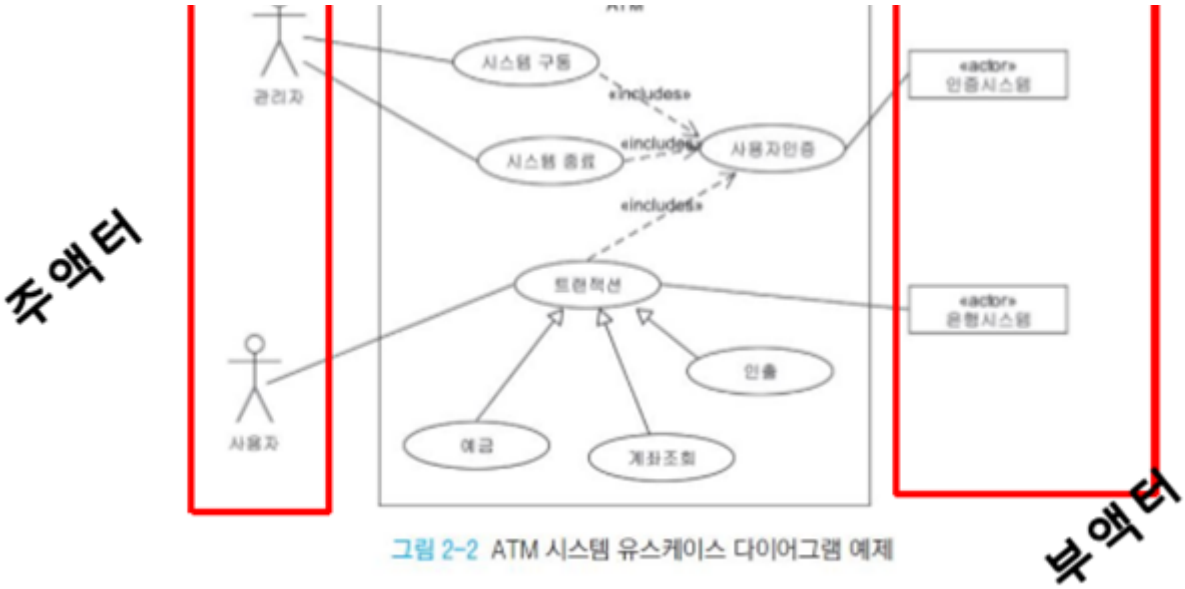


그림 2-2 ATM 시스템 유스케이스 다이어그램 예제

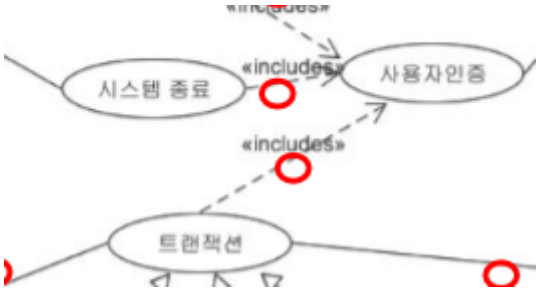
관계(relationship)

유스케이스 다이어그램에서는 관계는 액터와 유스케이스, 유스케이스와 유스케이스 사이에서 나타날 수 있으며 서로 상호작용을 한다는 의미로 해석해주면 되며 3가지 종류의 관계가 있다.
ex) 관리자와 시스템 구동간의 관계를 해석해보자 : 관리자가 시스템을 구동할 목적으로 시스템과 상호작용 한다.

1. 포함관계 (필수적 관계)

자 생각해 보면 시스템 구동이나 시스템 종료 나 시스템의 여러 기능을 사용하기 전 사용자 인증을 받아야한다.
관리자인지 확인도 안됐는데 시스템을 구동하거나 종료하게 하면 안돼지 않는가? 그러면 사용자 인증이란 기능은 공통적으로 많이 쓰이는데 이렇게 여러 유스케이스에서 중복되는 경우 따로 떼어내어 새로운 유스케이스를 만드는 경우를 포함관계라고 한다.

표기방법 : 원래의 유스케이스에서 새롭게 만들어진 유스케이스 방향으로 화살표를 점선으로 연결하고 <<include>>를 표기한다.
ex) "시스템 구동을 하기 위해 관리자가 ATM 시스템과 상호작용할 때 사용자 인증이 필요하다"라고 해석해준다.



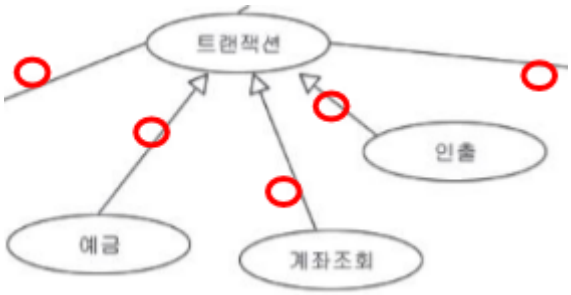
이부분

2. 일반화 관계

말 그대로 보다 보편적인 유스케이스와 구체적인 유스케이스 사이에 존재한다.

표기방법 : 구체적인 유스케이스에서 일반화된 유스케이스 방향으로 끝부분이 삼각형의 테두리로 표현된 화살표를 실선으로 연결하여 표현한다.

ex) 사과, 배 , 수박은 과일로 일반화 시켜줄 수 있다.



일반화 관계

3. 확장관계 (선택적 관계)

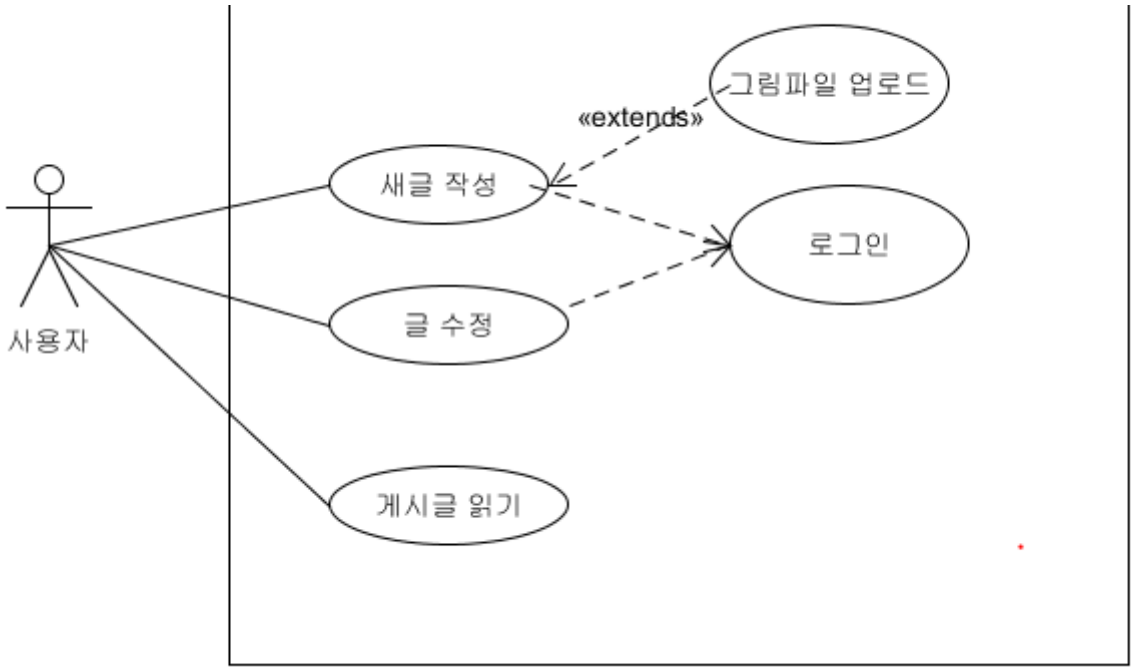
유스케이스가 특정한 조건이 만족되는 경우에만 실행되는 단계를 확장관계로 표현하며

표기방법 : 특정한 조건이 만족하는 경우에만 실행되는 단계를 따로 유스케이스로 만들고 원래의 유스케이스 방향으로 화살표를 점서능로 연결한 후 <<extend>>를 표기해준다.

예제

아래 문장을 유스케이스 다이어그램으로 표현해보자

1. 게시판 시스템을 만드려고한다. 액터는 사용자가 있다.
2. 사용자는 글을 만들수도 수정할수도 있다.
3. 글을 쓰거나 수정하는 경우 로그인을 해야한다.



유스케이스 해석방법

유스케이스 다이어그램을 봤을 때 아래와같은 방법으로 해석하거나 작성하면 유용하다.

1. S시스템에서 액터A가 U라는 유스케이스와 연관이 있는경우 "A는 U라는 목적을 달성하기 위해 S와 상호작용한다"라고 해석하자
2. U유스케이스가 U1유스케이스와 포함관계에 있는 경우 "U를 하기위해서는 U1을 반드시 수행한다"라고 해석하자
3. U유스케이스가 U2유스케이스와 확장관계에 있는 경우 "U를 수행하는 도중에 경우에따라 U2가 수행된다"라고 해석하자

주의사항(잘못된 유스케이스 다이어그램)

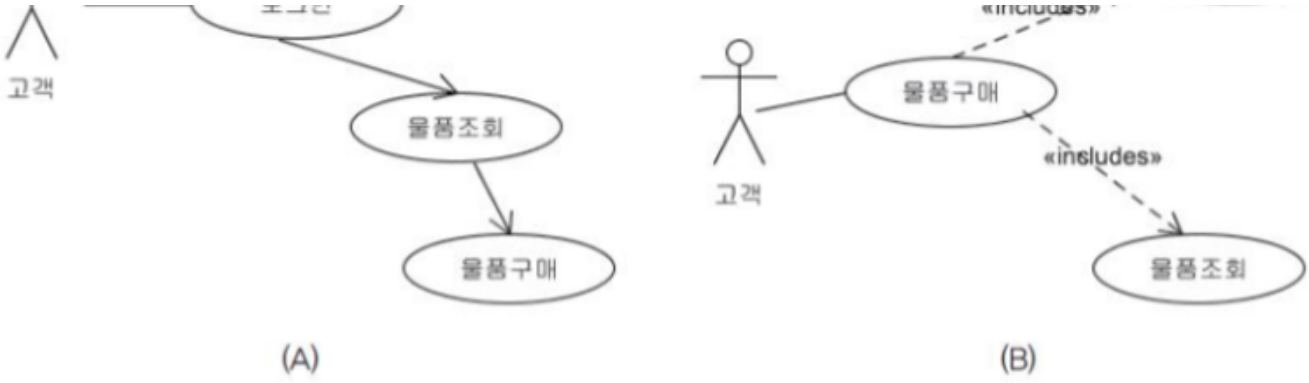


그림 2-3 잘못 작성된 유스케이스 다이어그램 예

위와 같은 식으로 유스케이스 다이어그램을 사용하면 무엇이 잘못된 걸까?

그림 A는 유스케이스다이어그램을 흐름도처럼 그린 경우이다.

유스케이스 다이어그램을 작성할 때 흐름도를 작성하는 것 처럼 그려서는 안된다. 유스케이스간의 관계는 포함, 확장, 일반화 관계만이 존재하고 실행순서 관계는 나타내지 않는다는 점을 명확히 알아야 한다.

그림B는 포함관계를 과도하게 사용하는 경우이다.

의도는 물품 구매가 로그인 단계와 물품조회 단계를 포함하기 때문에 포함관계로 연결하였으나 일반적으로 포함 관계는 여러개의 유스케이스가 서로 공통적인 부분을 갖고있는 경우에 중복된 부분을 유스케이스로 분리하여 표현하고자 할 때 사용된다. 예젯럼 물품 조회가 다른 물품 구매를 제외한 다른 유스케이스에서 전혀 사용되지 않는다면 이를 포함관계로 할 필요가 없다.

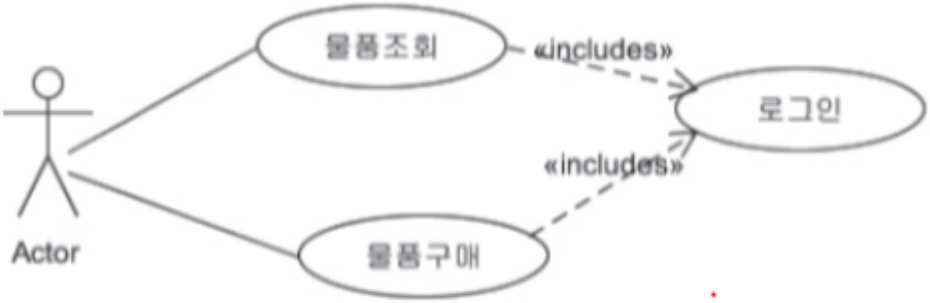


그림 2-4 올바르게 수정한 유스케이스 다이어그램

유스케이스 기술서는 액터가 해당 유스케이스의 목적을 달성하기 위하여 시스템과 상호작용 하는 과정을 구체적으로 묘사한 과정을 기술한 것으로 유스케이스 다이어그램에 있는 각 유스케이스에 대해 유스케이스를 작성해야 한다.

표 2-3 유스케이스 기술서 형식

유스케이스명	액터가 시스템을 통해 달성할 목적을 명확하게 하나의 문장으로 표현한다.
액터명	실제 사람의 이름이나 시스템의 이름을 사용하지 않고 시스템에서 수행하는 역할이름을 사용한다.
개요	유스케이스를 수행하는 개요를 기술한다.
사전조건	유스케이스의 기본 흐름이 올바르게 동작되기 위하여 사전에 충족되어야하는 조건을 기술한다.
사후조건	유스케이스가 실행된 후에 만족해야 하는 조건을 기술한다.
기본흐름	시스템과 액터 사이에 목적을 달성하기 위한 기본적인 상호작용 흐름을 기술한다. 기본흐름을 수행할 때에는 어떠한 오류나 예외가 발생하지 않고 모든 것이 완전하게 수행되는 것을 전제로 한다. 기본흐름의 첫 번째 단계는 해당 유스케이스를 시작하는 사건을 기술한다. 이를 트리거(trigger)라고 한다.
대체흐름	기본 흐름으로부터 경우에 따라 선택적으로 실행되고 다시 기본흐름으로 돌아오는 흐름이나 오류나 예외가 발생한 경우에 이를 처리하는 흐름을 기술한다.

기본흐름은 보통 정상적 흐름을 얘기하며, 대체흐름에는 오류적, 예외적인 상황을 기술한다.

5

유스케이스명	인출
액터명	주 액터: 사용자 부 액터: 인증시스템, 은행시스템
개요	사용자가 자신의 계좌로부터 예금 출금을 하기 위해 ATM을 이용한다.
사전조건	• 사용자가 은행 카드를 소지하고 있어야 한다. • 은행시스템과 인증시스템과의 네트워크 연결이 정상적이어야 한다. • 시스템은 사용자가 요구한 출금액만큼의 현금을 가지고 있어야 한다.
사후조건	• 사용자가 카드를 돌려받는다. 사용자가 돈을 출금하고 은행계좌에 출금액이 반영된다. • 사용자가 카드를 돌려받는다. 사용자가 돈을 출금하지 못하고 은행계좌는 변동이 없다. • 사용자가 카드를 돌려받지 못한다. 사용자가 돈을 출금하지 못하고 은행계좌는 변동이 없다. 또한 은행을 접촉하라는 메시지를 받는다.
기본흐름	1. 사용자는 은행 카드를 시스템에 제시한다.(트리거) 2. 시스템은 카드 정보를 읽고 비밀번호를 요구한다. 3. 사용자는 비밀번호를 입력한다. 4. 「사용자인증」 유스케이스를 실행한다. 5. 시스템은 사용자에게 트랜잭션 타입을 요구한다. 6. 사용자는 출금 트랜잭션을 선택한다. 7. 시스템은 출금액을 사용자에게 요구한다. 8. 사용자는 출금액을 입력한다. 9. 시스템은 은행시스템에 출금을 요구한다. 은행시스템은 사용자의 계좌에 출금을 반영한다. 10. 시스템은 돈을 사용자에게 내주고 출금 사실을 로그기록에 반영 한다 11. 시스템은 은행카드를 사용자에게 되돌려 준다.
대체흐름 1	2a. 시스템이 인식하지 못하는 카드가 입력되는 경우 2a.1 시스템은 사용자에게 카드 판별할 수 없다는 사실을 알리고 유스케이스를 종료한다.
대체흐름 2	9a. 잔금이 모자라는 경우 9a.1 사용자에게 알리고 단계 7을 다시 수행한다.

#유스케이스 #UML #모델링언어 #다이어그램 #소프트웨어공학 #유스케이스기술서

5

5

ljh0326s

인문대생의 컴퓨터공학 도전!

+ 이웃추가

이 블로그 소프트웨어공학 카테고리 글

#유스케이스 다른 글

Find Your Matching SIEM #3.SIEM의 필수 기능들은 무엇인가?

SCK · 2019. 7. 1.

♡ 0 💬 0

소프트웨어공학

분당캐드학원 · 2019. 5. 29.

💬 0

[소프트웨어 공학] 유스케이스 다이어그램 with starUML

이현 · 2019. 4. 2.

♡ 1 💬 0

커피메이커 유스케이스 별 세부 설계

Benny · 2019. 3. 31.

💬 0

[소프트웨어 공학] 유스케이스 다이어그램 with starUML

이현 · 2019. 3. 28.

♡ 1 💬 0

5

맨 위로

PC버전으로 보기