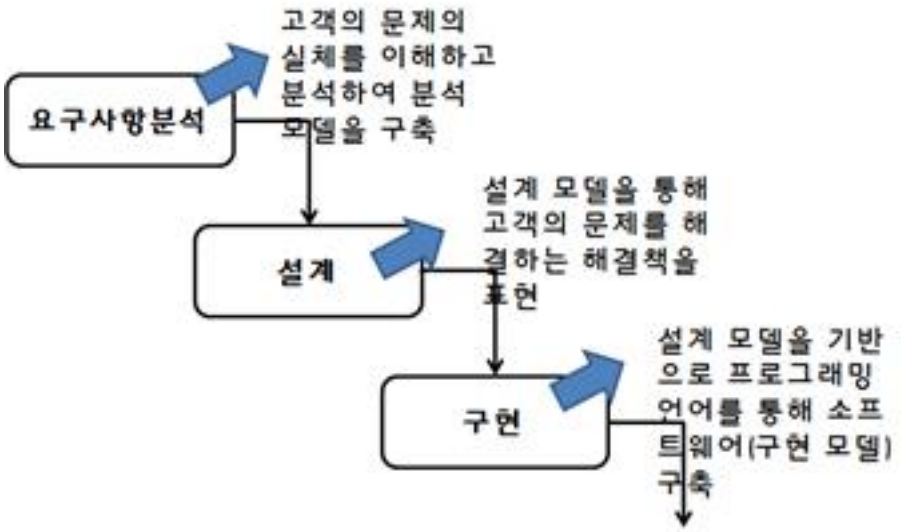
UML-유스케이스 다이어그램

한성대학교 컴퓨터공학부

정인상

개발 모델


요구사항분석
고객의 문제의 실체를 이해하고 분석하여 분석 모델을 구축
설계
설계 모델을 통해 고객의 문제를 해결하는 해결책을 표현
구현
설계 모델을 기반으로 프로그래밍 언어를 통해 소프트웨어(구현 모델) 구축

OOAD?

- 시스템을 일련의 상호작용하는 객체들로 모델링하여 개발하는 소프트웨어공학 접근 방식
- Analysis: 문제 영역에서 개념을 식별하고 기술하는 작업
- Design: 분석 단계에서 식별된 개념들을 소프트웨어 (객체)로 표현하고 구현이 가능하도록 하는 작업

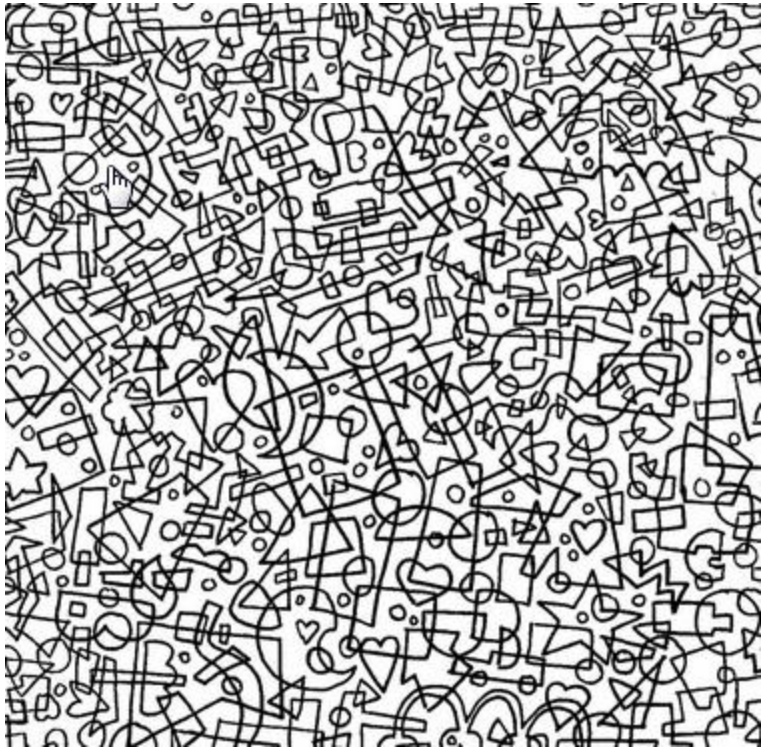
개념(클래스) 모델 작성: 분석 모델

- 초기 클래스 다이어그램
- 유스케이스 명세를 포함한 요구사항명세는 개념 즉, 클래스를 식별하기 위한 정보를 제공한다
- 가장 기본적인 클래스 추출 방법은 쓰임새 명세/문제 기술서에 나타나 있는 명사나 명사구를 추출하는 것이다.

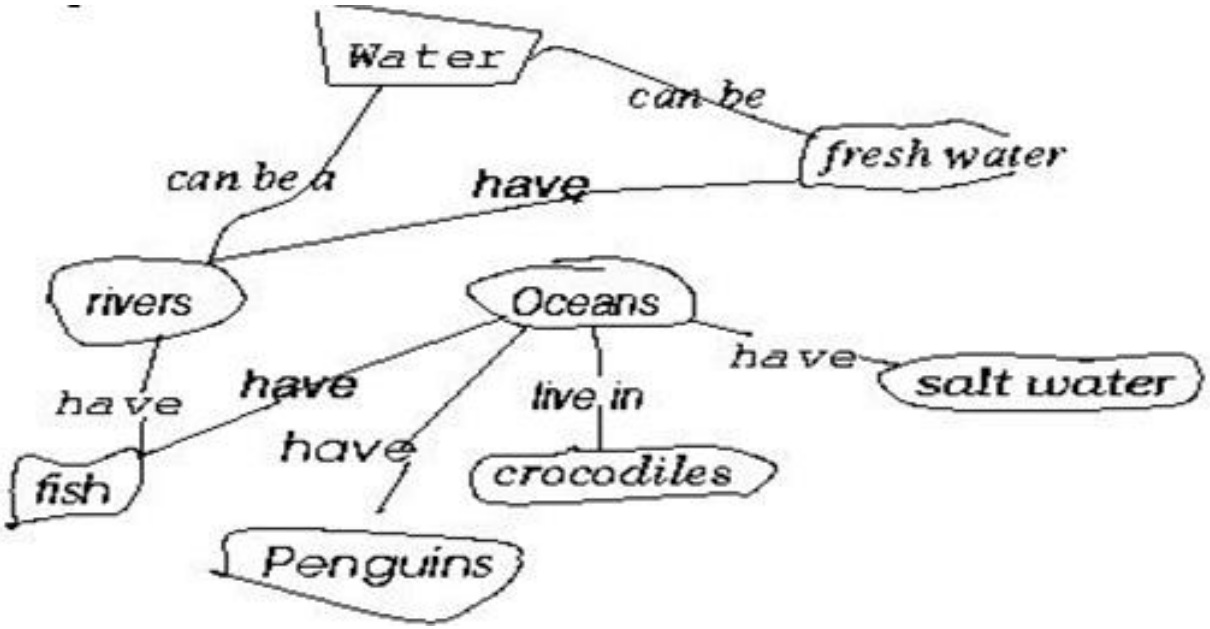
개념 모델을 통해 세상을 본다

- 유아기에는 세상은 윙윙거리는 혼돈으로 가득차있다
- 아주 아렸을 때에는 “파란 하늘”을 “파란” 개념과 “하늘” 개념을 분리된 것으로 보지만 시간이 지남에 따라 “파란 하늘”하나의 복잡한 개념으로 인식한다.
- 더 나이가 들면 개념은 더 정제가 된다.
  - ✓ *the sky is blue only on cloudless days*
  - ✓ *the sky is not really blue*
  - ✓ *it only looks blue from our planet Earth because of atmospheric effects*

뭐가 보이는가?



개념 모델을 통해 세상을 본다

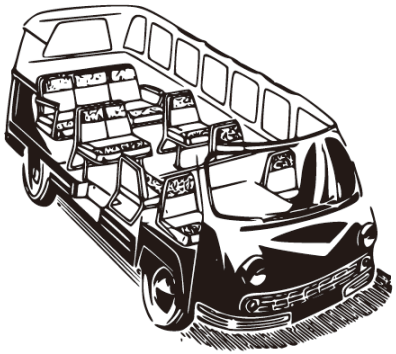


모델링

❖ 모델의 역할

- 서로의 해석을 공유해 합의를 이루거나 해석의 타당성을 검토
- 현재 시스템 또는 앞으로 개발할 시스템의 원하는 모습을 가시화
- 시스템의 구조와 행위를 명세
- 시스템을 구축하는 틀 제공

그림 1-1 자동차의 전체적 구조를 파악할 수 있는 자동차 모델

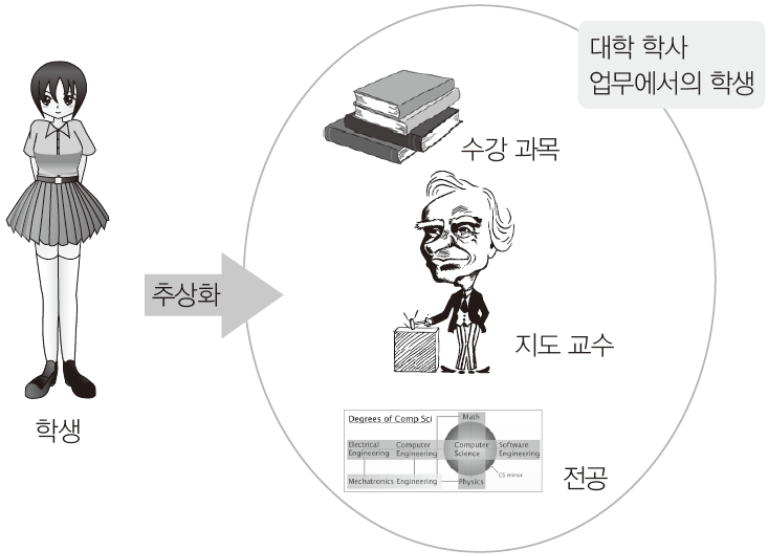


모델링

❖ 모델은 추상화에 바탕을 두고 있다

- 특정 관점에서 관련이 있는 점은 부각 관련이 없는 것은 무시

그림 1-2 대학 학사 업무의 추상화



UML

❖ 대표적인 시스템 모델링 언어

- 제임스 럼버 OMT + 야콥슨 OOSE + 부치 OOAD
- 2012년 UML 2.5

표 1-1 UML 다이어그램의 종류

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.

요구사항이란?

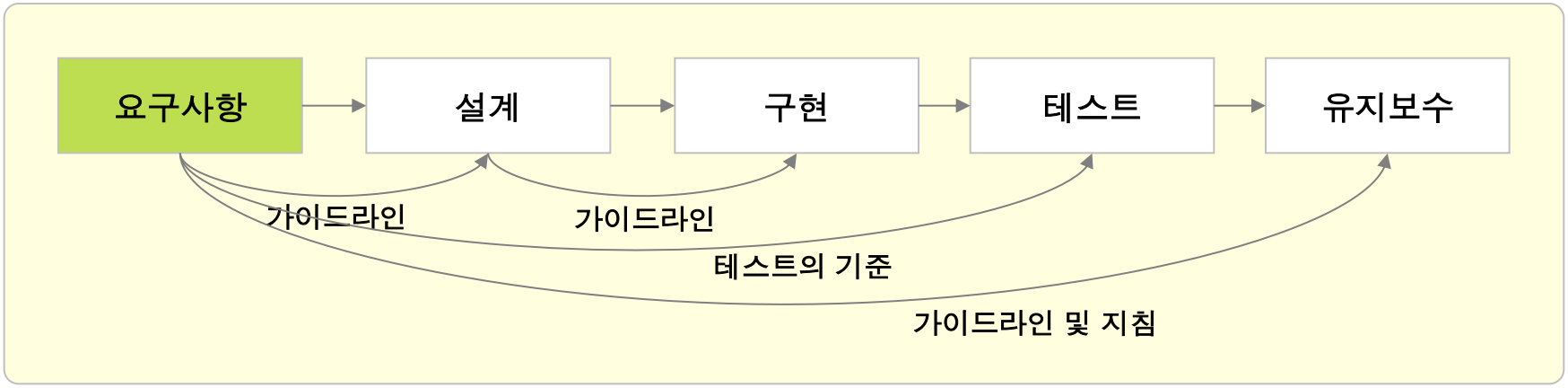
- **정의**
  - 문제의 해결 또는 목적 달성을 위하여 고객에 의해 요구되거나, 표준이나 명세 등을 만족하기 위하여 시스템이 가져야 하는 서비스 또는 제약사항
  - 고객이 요구한 사항과 요구하지 않았더라도 당연히 제공되어야 한다고 가정되는 사항들

if you get the requirements wrong, the resulting application won't solve the users' problems.
You'll be like a tourist in Boston with a broken GPS.

요구사항의 중요성

❖ 요구사항의 중요성

- 참여자들로 하여금 개발되는 소프트웨어 제품을 전체적으로 파악하도록 하여 의사 소통 시간을 절약하게 해 주는 것
- 상세한 요구사항이 있어야만 산정이 가능하고, 이를 기반으로 계획을 세울 수 있기 때문



요구사항의 분류

❖ 기능적 요구사항(Functional Requirements)

- **수행될 기능과 관련되어 입력과 출력 및 그들 사이의 처리과정**
- **목표로 하는 제품의 구현을 위해 소프트웨어가 가져야 하는 기능적 속성**
  - 예) 워드 프로세서에서 파일 저장 기능, 편집 기능, 보기 기능 등

❖ 비기능적 요구사항(Non-Functional Requirements)

- **제품의 품질 기준 등을 만족시키기 위해 소프트웨어가 가져야 하는 성능, 사용의 용이성, 안전성과 같은 행위적 특성**
- **시스템의 기능에 관련되지 않는 사항을 나타냄**
  - 예) 성능(응답 시간, 처리량), 사용의 용이성, 신뢰도, 보안성, 운용상의 제약, 안전성 등
- **아키텍처 결정에 많은 영향을 준다**
- **품질속성이라고도 한다**

예

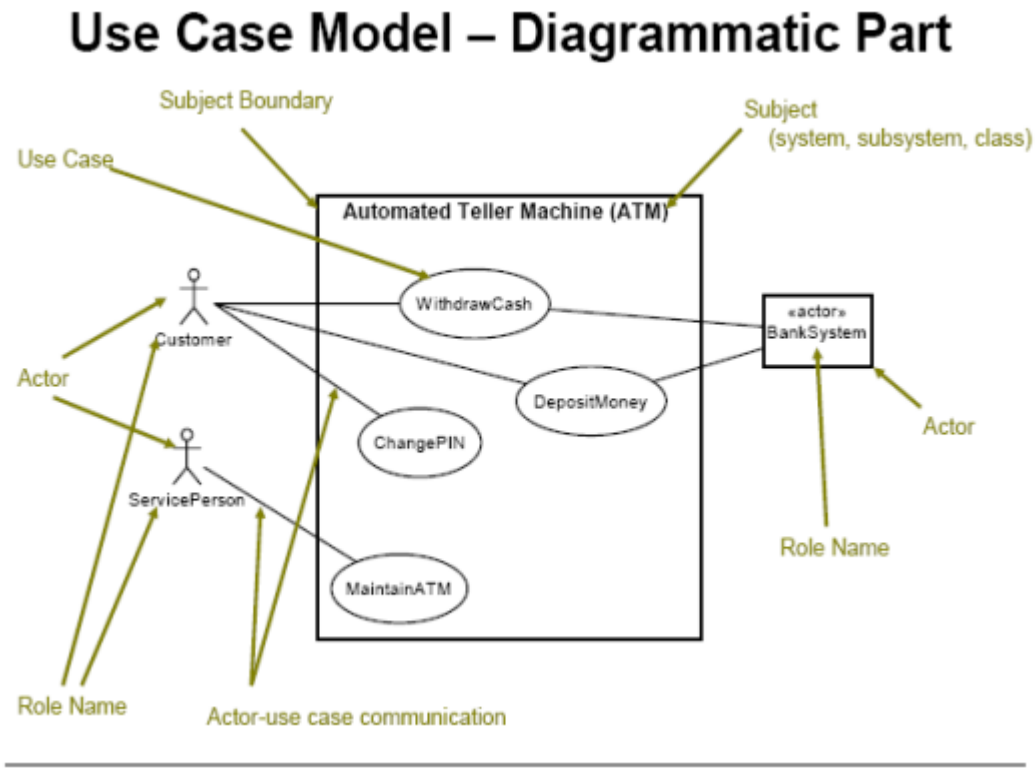
"Allow users to reserve a hovercraft online."

"The application must support 20 users simultaneously making reservations at any hour of the day."

또 다른 형태의 요구사항

- 표준준수
- 인터페이스 요구사항
- 물리적 요구사항
  - require a minimum amount of processing power,
  - a maximum amount of electrical power,
  - Easy portability (such as a tablet or smartphone), touch screens, or environmental features (must work in boiling acid).

유스케이스 다이어그램


Use Case Model – Diagrammatic Part
Subject Boundary
Subject
(system, subsystem, class)
Use Case
Automated Teller Machine (ATM)
WithdrawCash
«actor»
BankSystem
Customer
Actor
DepositMoney
ChangePIN
Actor
ServicePerson
Role Name
MaintainATM
Role Name
Actor-use case communication

요구사항 검증

❖ 요구사항 타당성 검증 사항

검증 사항	설명
무결성(correctness) 및 완전성(completeness)	사용자의 요구를 에러 없이 완전하게 반영하고 있는가?
일관성(consistency)	요구사항이 서로간에 모순되지 않는가?
명확성(unambiguous)	요구분석의 내용이 모호함 없이 모든 참여자들에 의해 명확하게 이해될 수 있는가?
기능성(functional)	요구사항 명세서가 “어떻게” 보다 “무엇을”에 관점을 두고 기술되었는가?
검증 가능성(verifiable)	요구사항 명세서에 기술된 내용이 사용자의 요구를 만족하는가? 개발된 시스템이 요구사항 분석 내용과 일치하는지를 검증할 수 있는가?
추적 가능성(traceable)	시스템 요구사항과 시스템 설계문서를 추적할 수 있는가?

Clear

- **Bad example:**
  - UR2: All screens must appear on the monitor quickly.
  - **How long is quickly?**

- **Bad example:**
  - UR2: All screens must appear on the monitor quickly.
  - **How long is quickly?**

- **Good example:**
  - UR2: When the user accesses any screen, it must appear on the monitor within 2 seconds.

Unambiguous

- “Find the best route from a start location to a destination location.”

complete

- **Bad example:**
  - UR3: On loss of power, the battery backup must support normal operations.
  - **For how long?**

Complete

- **Bad example:**
  - UR3: On loss of power, the battery backup must support normal operations.
  - **For how long?**

- **Good example:**
  - UR3: On loss of power, the battery backup must support normal operations for 20 minutes.

Consistent

- 요구사항 간에는 서로 모순되지 않아야 한다.
- 문제를 해결할 수 없을 정도로 너무 많은 제약이 주어지면 안된다.
- 개개의 요구사항도 일관성을 가져야 한다. (In other words, it must be possible to achieve.)

Reduce appointment start windows to no more than 2 hours while meeting 90 percent of the scheduled appointments.

Feasible

- **Bad example:**
  - The replacement control system shall be installed with no disruption to production.
  - **This is an unrealistic expectation.**

- **Bad example:**
  - The replacement control system shall be installed with no disruption to production.
  - **This is an unrealistic expectation.**

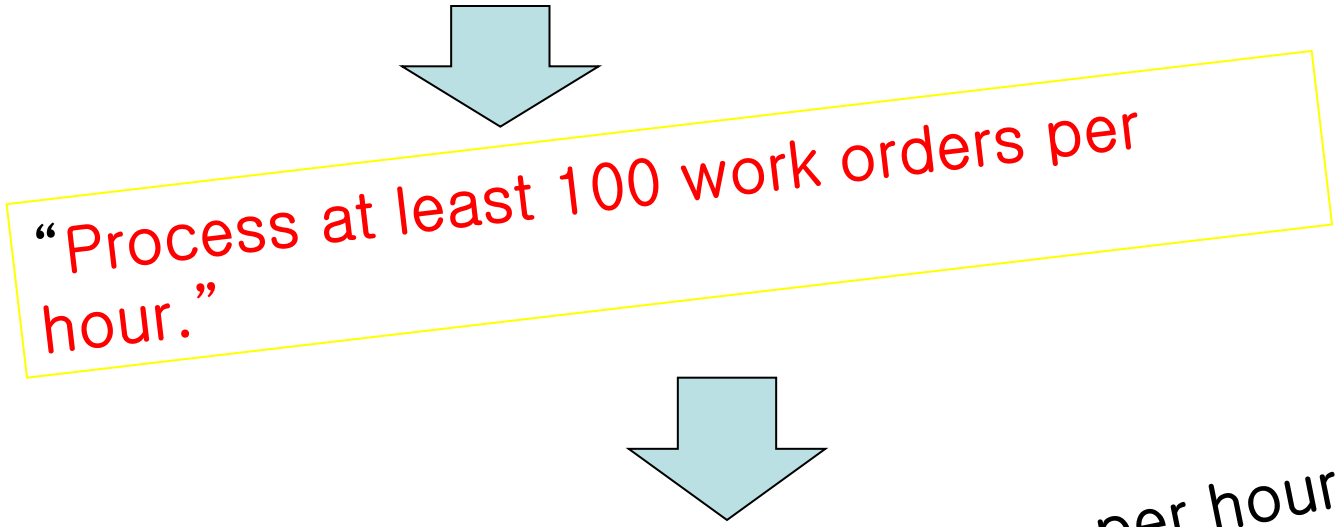
- **Good example:**
  - The replacement control system shall be installed causing no more than 2 days of production disruption.

Fast, Good, Cheap

- Build something quickly with high quality and high cost.
- Build something quickly and inexpensively but with low quality.
- Build with high quality and low cost but over a long time.

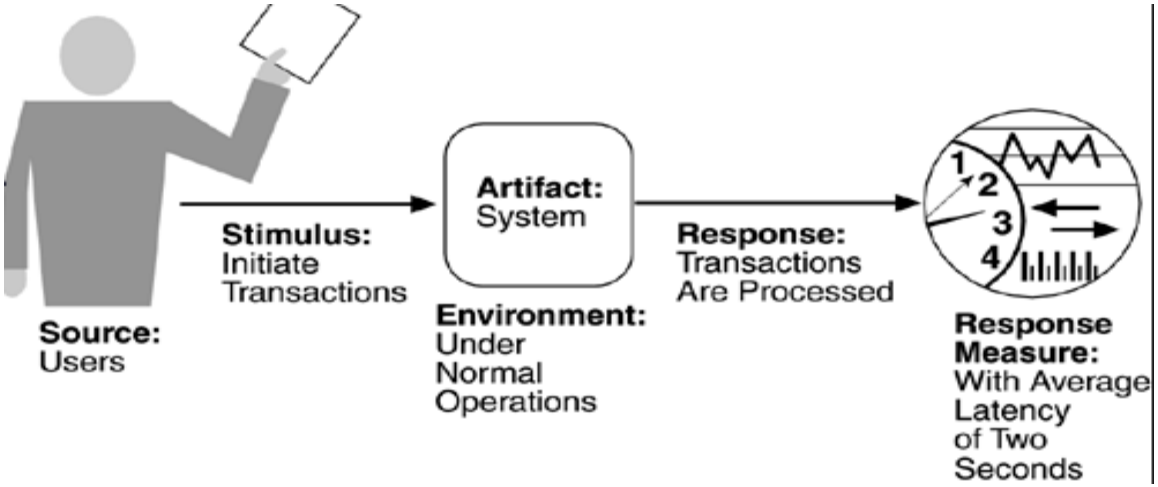
Verifiable

- "Process more work orders per hour than are currently being process



품질속성을 검증가능한 시나리오로

- **품질속성템플릿**
  - 소스(Source) : 자극을 생성하는 개체
  - 자극(stimulus): 시스템에 영향을 주는 조건
  - 대상 또는 산출물(artifact) : 자극에 의해 영향받는 시스템(부분)
  - 환경(environments): 자극이 발생된 상황
  - 반응(response): 자극으로 야기되는 시스템 행위
  - 반응척도(measure): 시스템의 반응을 평가하는 척도

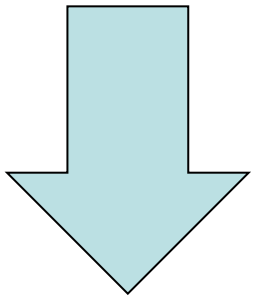


예제

- **사용자**: 등록된 회원이
- **자극**: 주소 변경을 요구했을 때
- **대상**: 회원 정보 갱신 시스템
- **환경**: 시스템이 최대 부하가 걸린 상태에서
- **반응**: 회원계정 갱신 트랜잭션이 수행되며
- **척도**: 이 트랜잭션은 0.15초 내에 완료되어야 한다

What, not on How

- 토핑폼은 선택가능한 토핑들을 보여줄 것이다. 사용자는 토핑 옆에 있는 체크박스를 체크하여 원하는 피자 토핑을 추가한다.



The toppings form will allow the user to select the toppings put on the pizza.

MOSCOW

알아둡시다

우선 순위를 매길 때 널리 알려진 방법으로 MOSCOW 방법이 있다. 이 방법은 다음과 같이 4가지의 우선순위 등급을 갖는다:

- M (Must): 성공적인 제품이 되기 위해서는 반드시 제공해야하는 필수적인 기능이다. 이 기능이 제공되지 않으면 제품으로써의 존재가치가 상실되지만 일단 제공되면 제품의 가치를 상승하는데 더 이상 기여를 하지 않는다.
- S (Should have): 중요하고 유용한 기능이며 가능하다면 포함되었으면 하는 기능이다. 만약 일정에 여유가 없거나 대안이 있는 경우에는 다음 일정에 포함되도록 개발을 마룰 수 있다.
- C (Could have): 만약 제공된다면 사용자의 만족도를 올릴 수 있는 바람직한 기능이다. 현재 개발 일정에 여유가 없어 제공되지 않는다 할지라도 크게 문제가 되지 않는 기능을 의미한다.
- W (Wont have the time): 현재 개발 일정에서 누락되어도 아무 상관이 없는 기능을 의미한다.

유스케이스 기반의 요구사항 분석

유스케이스 다이어그램

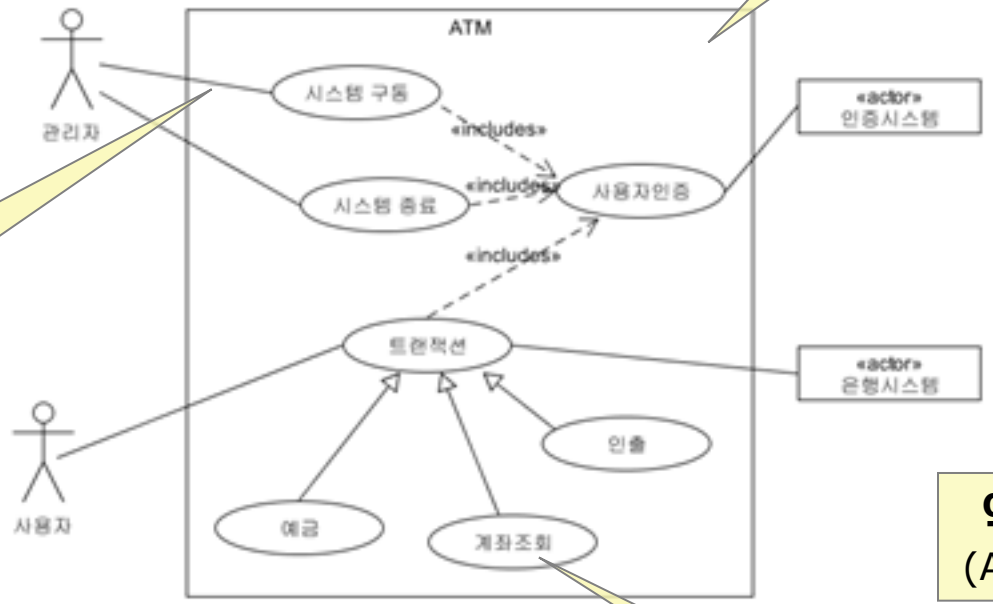
- 시스템을 사용하는 목적들, 즉 쓰임새들을 사용자 관점에서 기술한 다이어그램이며 이 목적 달성을 위한 사용자와 시스템 사이의 상호작용을 보여준다.
- 시스템이 제공하는 기능이나 서비스 등을 정의하고, 시스템의 범위를 결정

유스케이스 다이어그램 구성요소

primary

액터
(Actor)

시스템
(System)



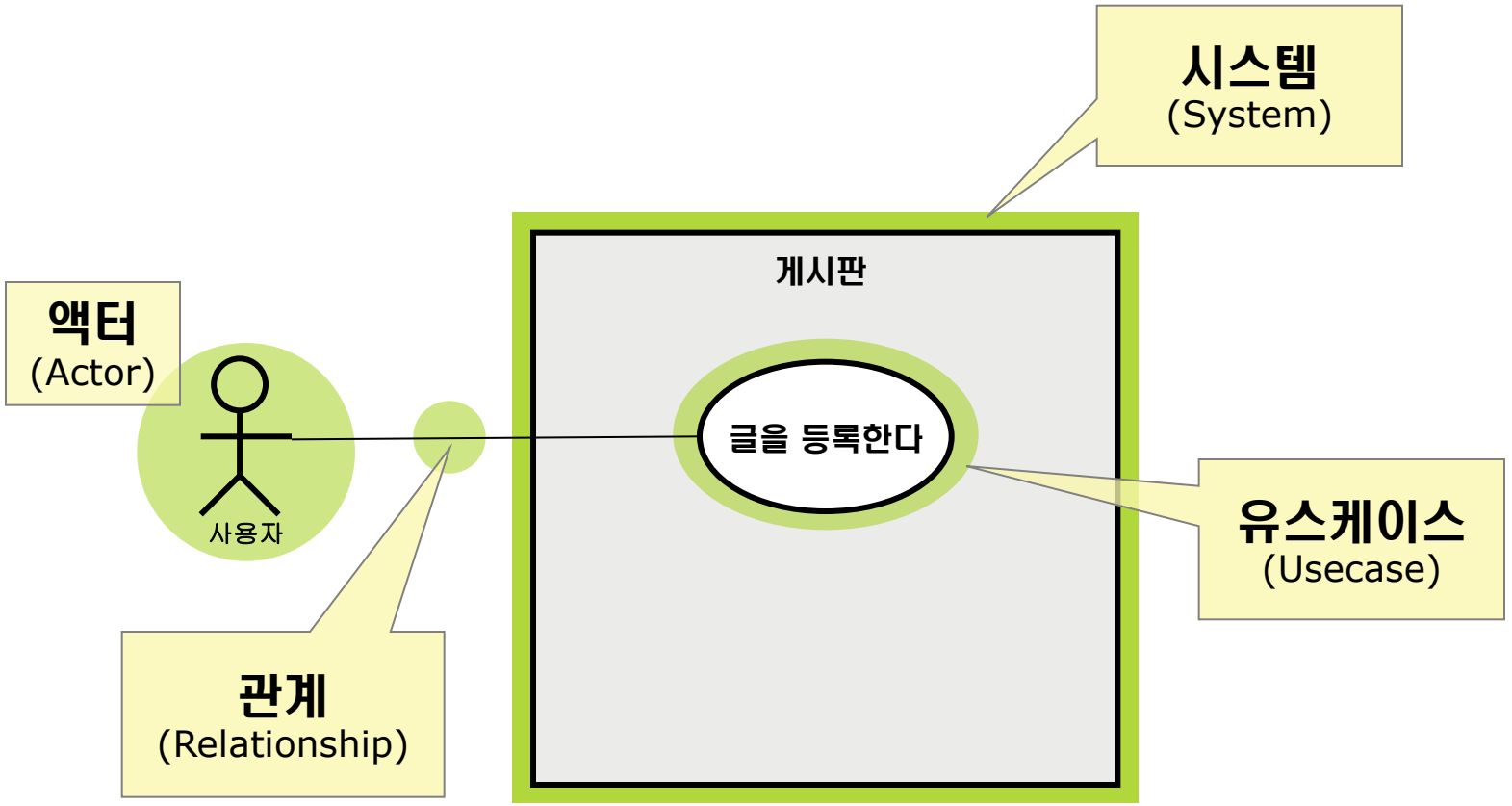
관계
(Relationship)

secondary

액터
(Actor)

유스케이스
(Usecase)

유스케이스 다이어그램의 구성요소



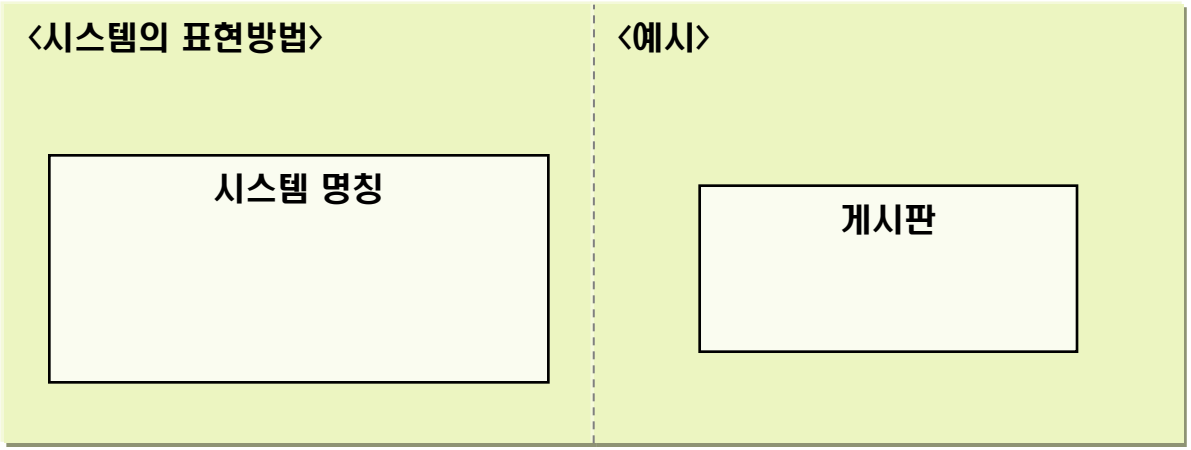
시스템(System)

❖ 의미

- 만들고자 하는 어플리케이션

❖ 표기법

- 유스케이스를 둘러싼 사각형의 틀을 그리고, 시스템 명칭을 사각형 안쪽 상단에 기술



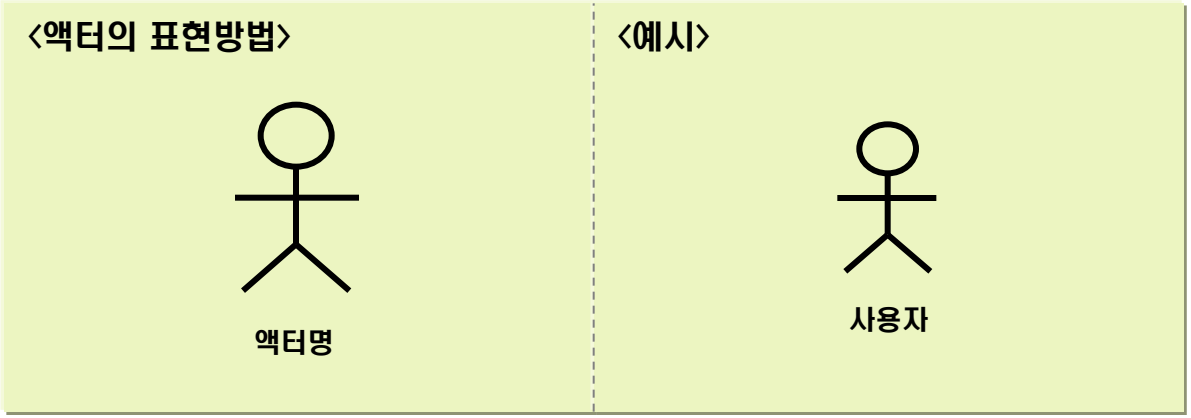
액터(Actor)

❖ 의미

- 시스템의 외부에 있으면서 시스템과 상호 작용을 하는 사람 또는 다른 시스템

❖ 표기법

- 원과 선을 조합하여 사람 모양으로 표현
- 그 위 또는 아래에 액터명 표시
- 액터명은 액터의 역할로 정함



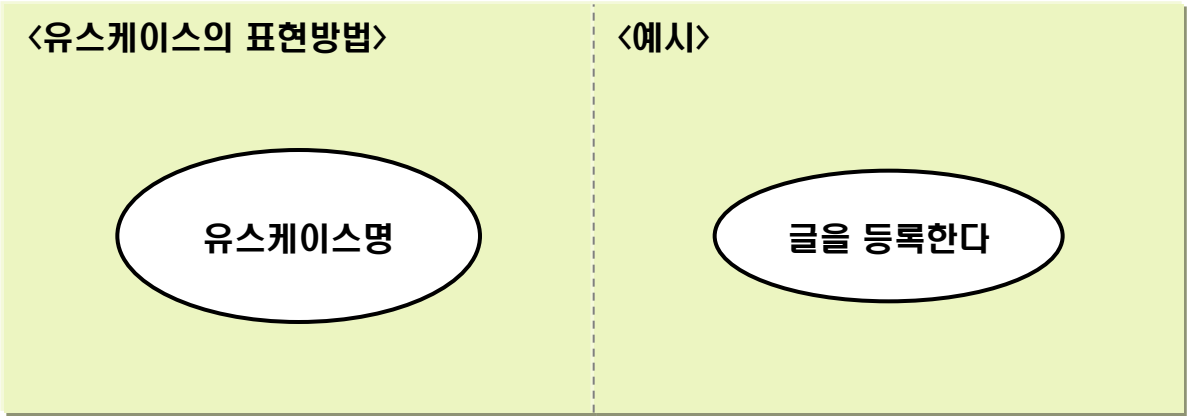
유스케이스(Usecase)

❖ 의미

- 시스템이 액터에게 제공해야 하는 기능의 집합
- 시스템의 요구사항을 보여줌

❖ 표기법

- 타원으로 표시하고 그 안쪽이나 아래쪽에 유스케이스명을 기술
- 유스케이스의 이름은 “~한다” 와 같이 동사로 표현
- 각 유스케이스가 개발될 기능 하나와 연결될 수 있도록 함



관계(Relationship) (1/3)

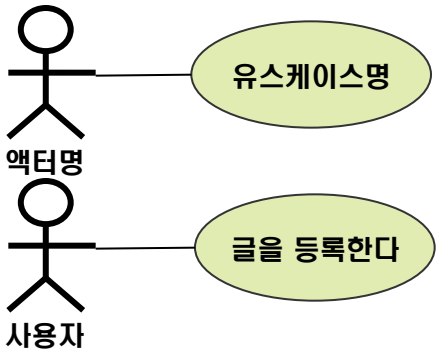
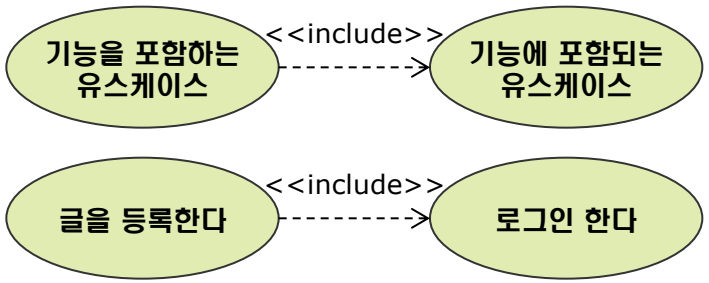
❖ 의미

- 액터와 유스케이스 사이의 의미 있는 관계

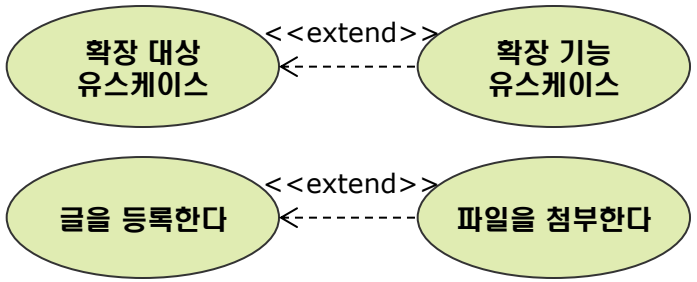
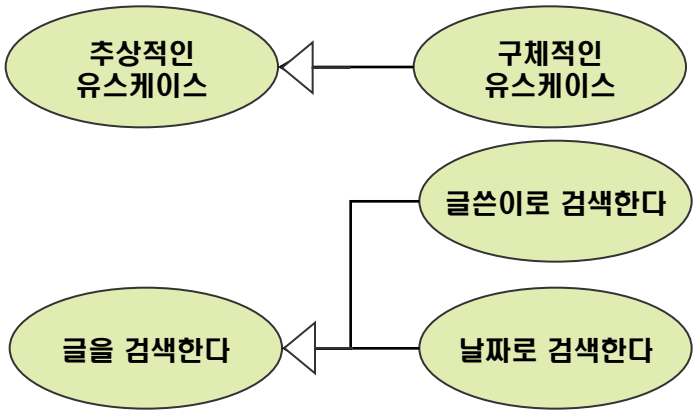
❖ 종류

- 연관 관계(association)
- 의존 관계(dependency)
  - 포함 관계(include)
  - 확장 관계(extend)
- 일반화 관계(generalization)

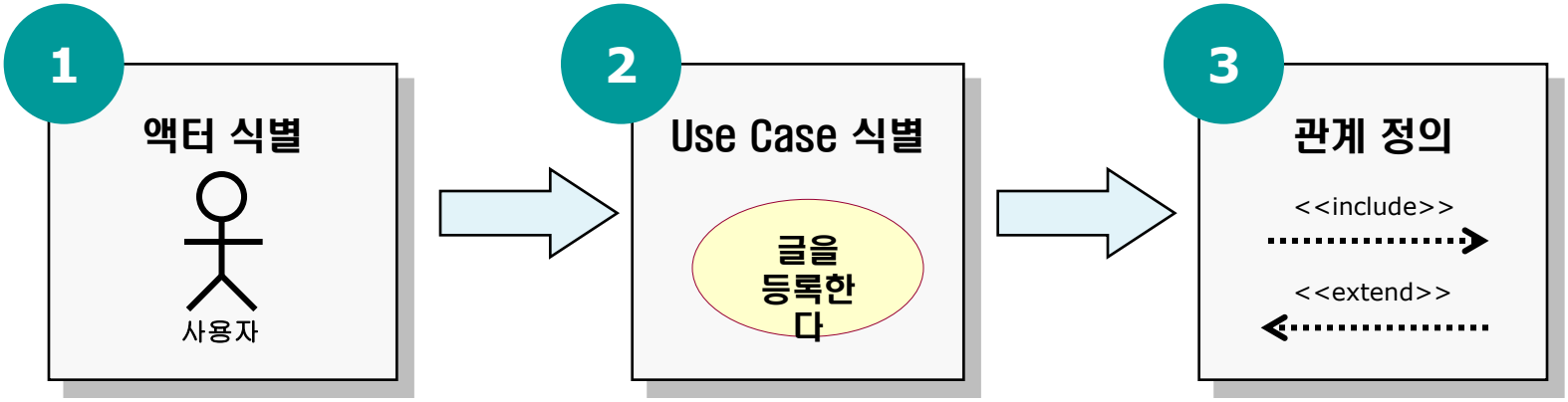
관계(2/3)

관계 종류	설명	표기법
연관 관계 (association)	• 유스케이스와 액터간의 상호작용이 있음을 표현 • 유스케이스와 액터를 실선으로 연결함	
포함 관계 (include)	• 하나의 유스케이스가 다른 유스케이스의 실행을 전제로 할 때 형성되는 관계 • 포함되는 유스케이스는 포함하는 유스케이스를 실행하기 위해 반드시 실행되어야 하는 경우에 적용 • '포함하는 유스케이스'에서 '포함되는 유스케이스' 방향으로 화살표를 점선으로 연결하여 표현하고 <<include>>라고 표기	

관계(3/3)

관계 종류	설명	표기법
확장 관계 **(extend)**	• 확장 기능(Extending) 유스케이스와 확장 대상(Extended) 유스케이스 사이에 형성되는 관계 • 확장 대상 유스케이스를 수행 할 때에 특정 조건에 따라 확장 기능 유스케이스를 수행하기도 하는 경우에 적용 • '확장 기능 유스케이스'에서 '확장 대상 유스케이스' 방향으로 화살표를 점선으로 연결하여 표현하고 <<extend>>라고 표기	
일반화 관계 **(generalization)**	• 유사한 유스케이스들 또는 액터들을 모아 그들을 추상화한 유스케이스 또는 액터와 연결시켜 그룹핑(Grouping)함으로써 이해도를 높이기 위한 관계 • '구체적인 유스케이스'에서 '추상적인 유스케이스' 방향으로 끝부분이 삼각형의 테두리로 표현된 화살표를 실선으로 연결하여 표현	

유스케이스 다이어그램 작성 순서



- 사용자의 역할을 식별
- 외부 시스템 식별

- 액터가 요구하는 서비스 식별
- 액터가 요구하는 정보 식별
- 액터가 시스템과 상호작용하는 행위 식별

- 액터간 유스케이스간의 일반화 정의
- 액터간 유스케이스간의 연관관계 정의
- 유스케이스간 포함, 확장관계 정의

액터 식별

- 액터를 찾기 위한 질문들
  - 누가 정보를 제공하고, 사용하고, 삭제하는가?
  - 누가 또는 어떤 조직에서 개발될 시스템을 사용할 것인가?
  - 누가 요구사항에 대해 관심을 가지고, 시스템이 만들어낸 결과에 관심이 있는가?
  - 누가 시스템이 잘 운영될 수 있도록 유지보수 및 관리를 하는가?
  - 개발될 시스템과 상호작용하는 하드웨어나 소프트웨어 시스템은 무엇인가?

유스케이스 식별

- **유스케이스를 찾기 위한 질문들**
  - 액터가 원하는 시스템 제공 기능은 무엇인가?
  - 액터는 시스템에 어떤 정보를 생성, 수정, 조회, 삭제하고 싶어 하는가?
  - 액터는 시스템의 갑작스러운 외부 변화에 대해 어떤 정보를 필요로 하는가?
  - 시스템이 어떤 기능을 제공하면 액터의 일상 작업이 효율적이고 편리해지는가?
  - 모든 기능 요구사항들을 만족할 수 있도록 유스케이스가 모두 식별되었는가?

관계를 식별하기 위한 질문

- 연관 관계(Association)
  - 액터와 유스케이스 간에 상호 작용이 존재하는가?
- 포함 관계(Include)
  - 이 유스케이스를 실행하기 위하여 반드시 실행되어야 하는 유스케이스가 존재하는가?
- 확장관계(Extend)
  - 이 유스케이스를 실행함으로써 선택적으로 실행되는 유스케이스가 있는가?
- 일반화 관계(Generalization)
  - 액터 또는 유스케이스가 구체화 된 다른 여러 액터나 유스케이스를 가지고 있는가?

[예] 게시판

❖ 예제 요구사항

- SE사는 A고객으로부터 다음의 요구사항을 전달받았다.

· 사용자 요구사항

: 글을 등록, 수정, 삭제할 수 있는 게시판을 개발한다.
(단, 관리자 모드는 개발하지 않는다)

- **글을 등록할 때에는 파일을 첨부할 수 있다.**
- **글을 조회하여 읽을 수 있다.**
- **등록된 글은 글쓴이 혹은 날짜 별로 검색할 수 있다.**
- **게시판의 모든 기능은 사용자 로그인 후에 사용할 수 있다.**

1) 시스템 식별

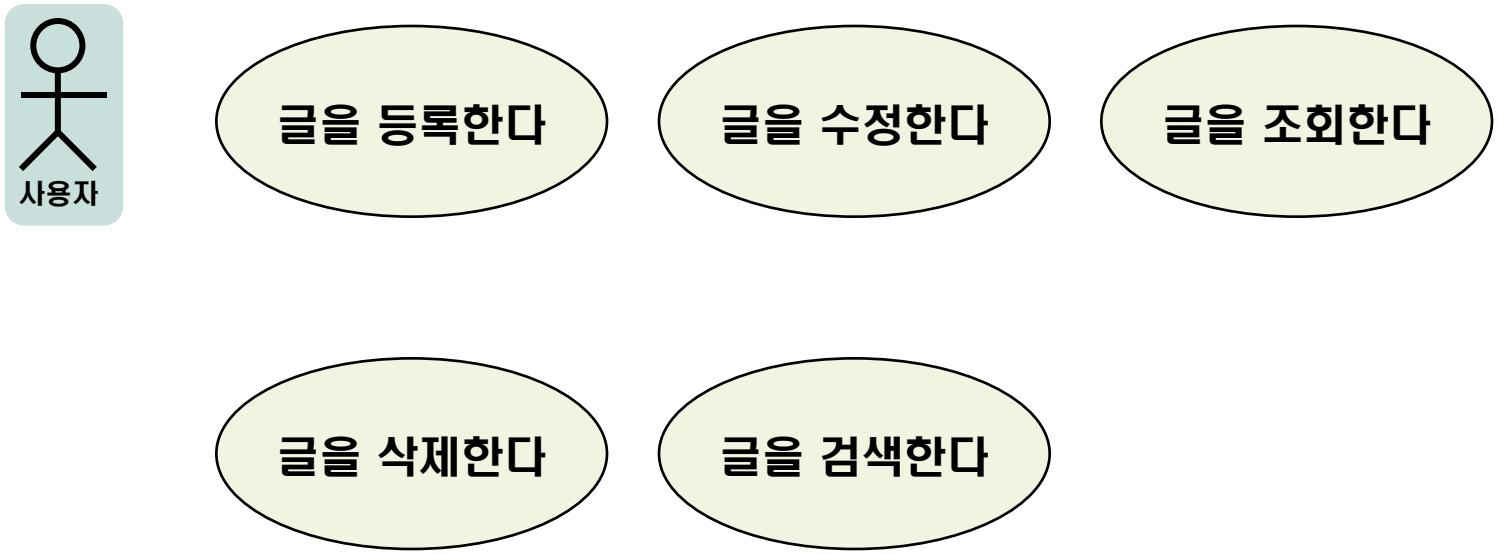
❖ 요구사항을 통해 만들고자 하는 시스템은 ‘게시판’ 임

2) 액터 식별

- 개발할 '게시판' 외부에서 상호작용하는 액터로 글을 등록하고 삭제하는 등의 역할을 하는 '사용자'가 식별됨

3) 유스케이스 식별

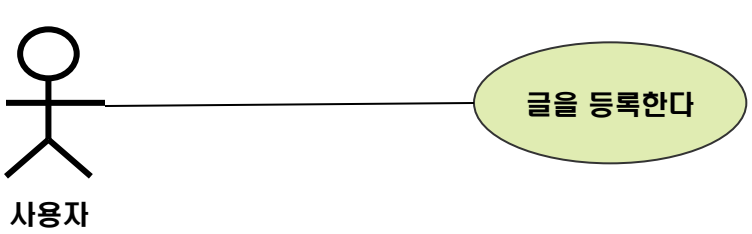
- '사용자' 는 게시판을 통해 글을 등록, 수정, 조회하는 등의 작업을 함



4) 관계 정의

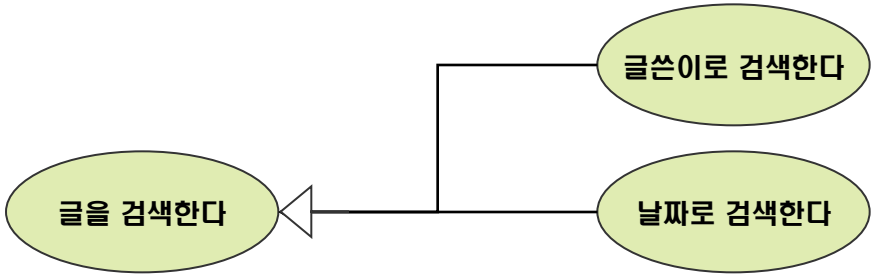
- **연관관계**

(액터-유스케이스)



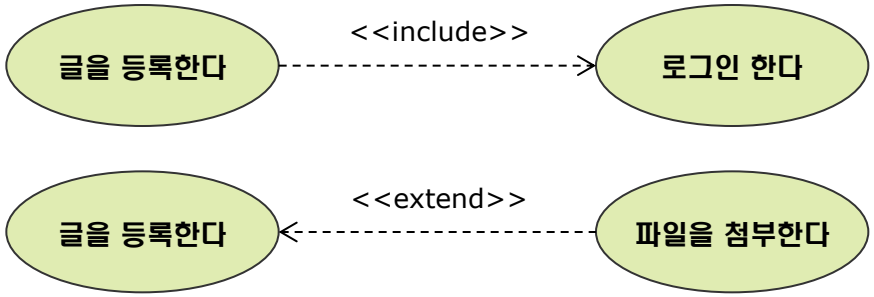
- **일반화관계**

(유스케이스(액터)-유스케이스)

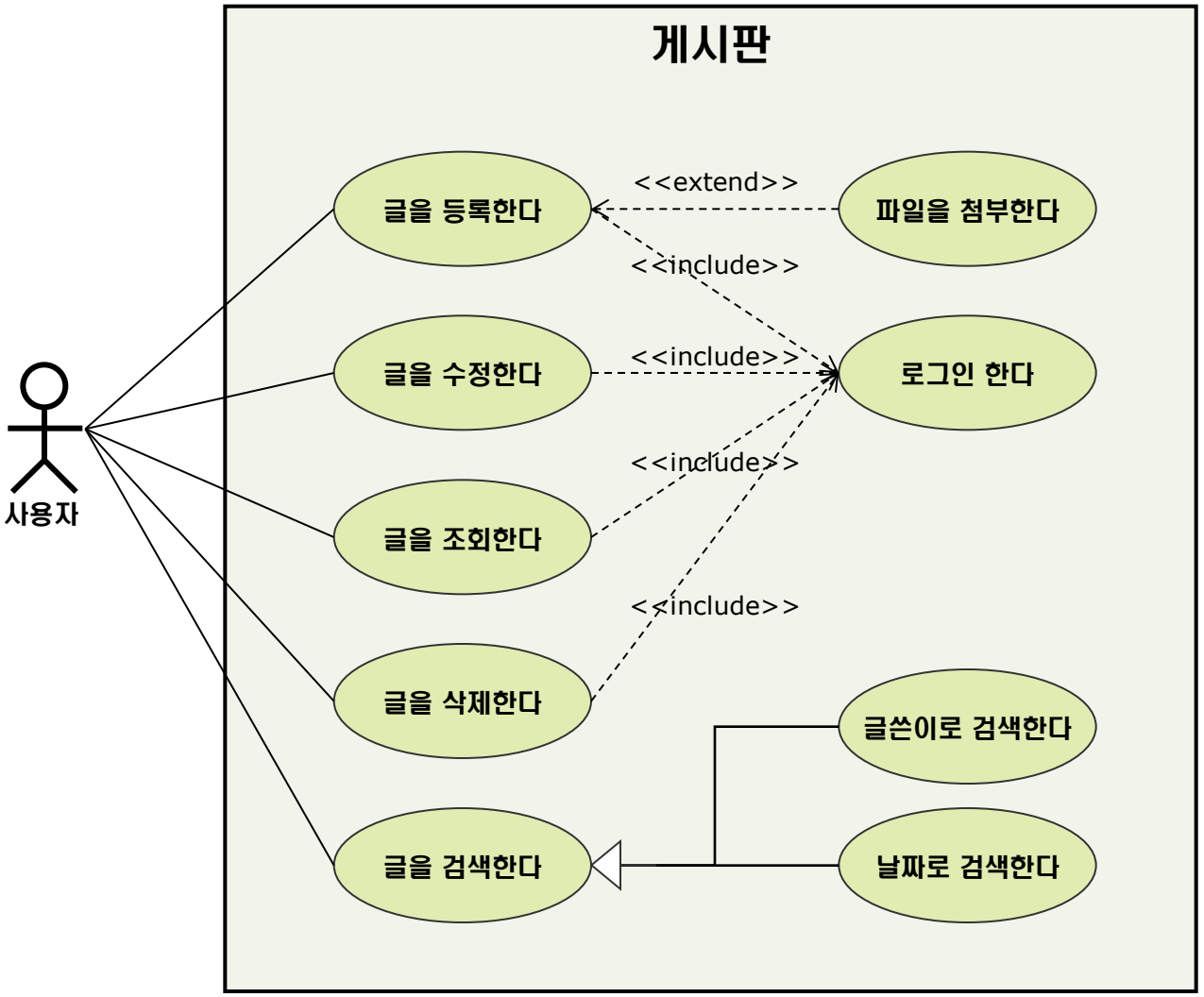


- **포함관계, 확장관계**

(유스케이스-유스케이스)

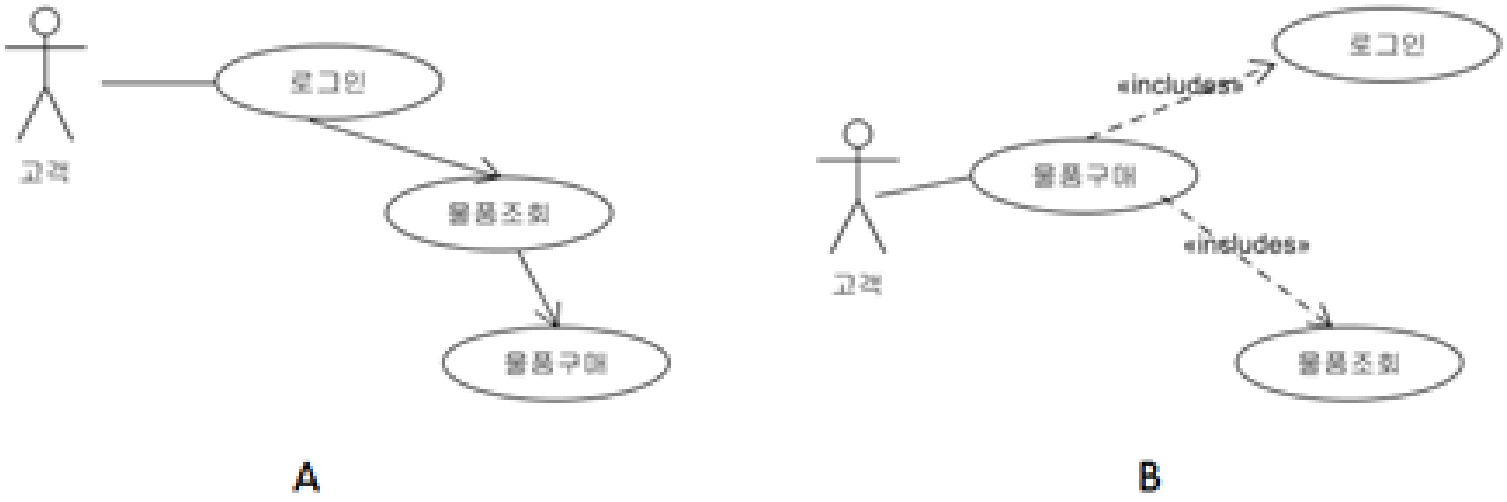


완성된 게시판 유스케이스 다이어그램



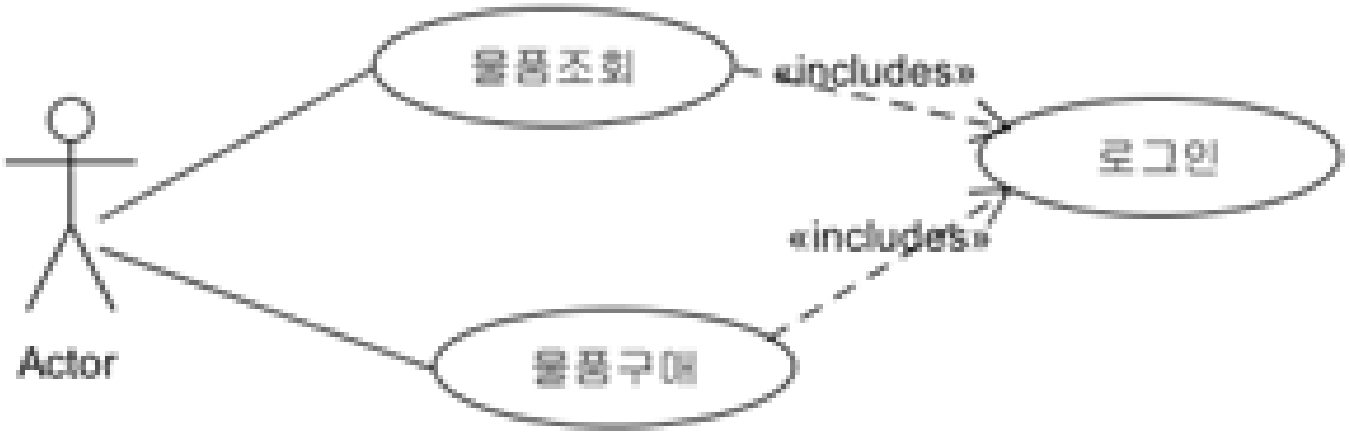
주의!!!

관계를 남용하지 마라.

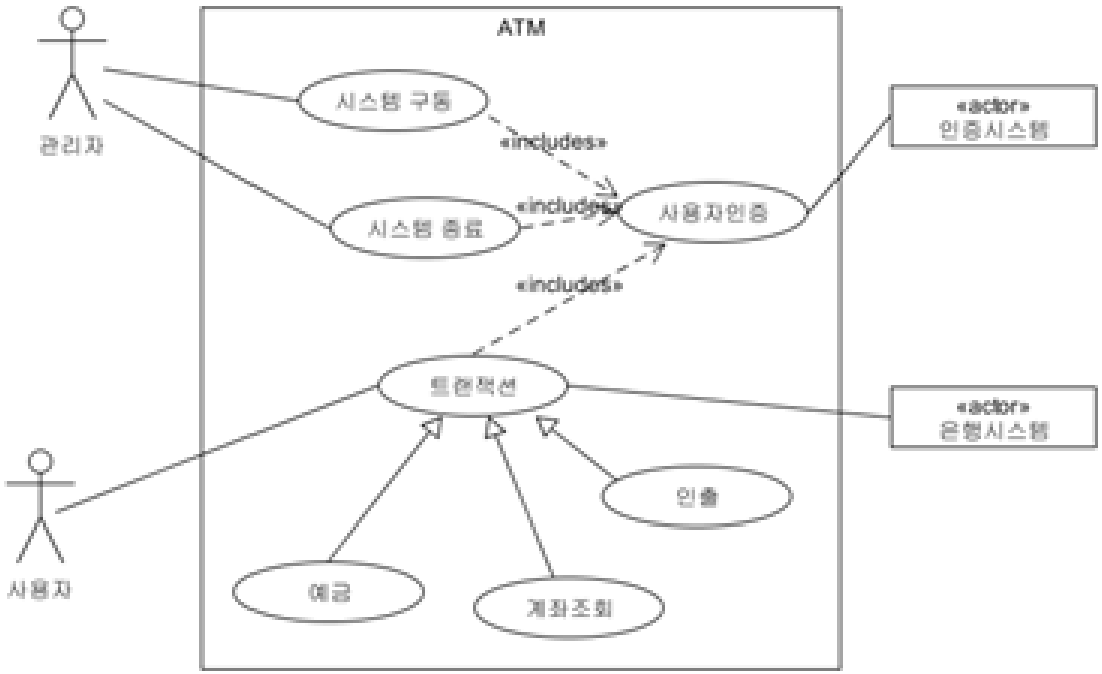


유스케이스 다이어그램은 흐름도가 아니다.

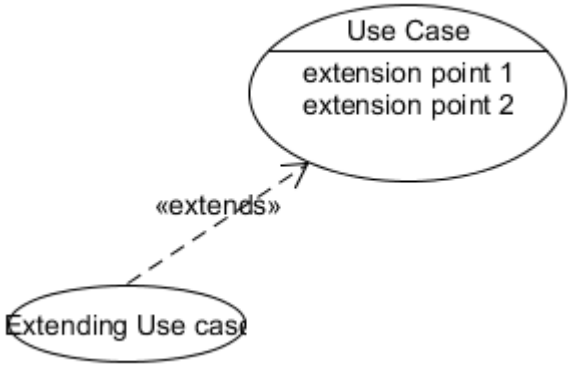
올바른 예



예제

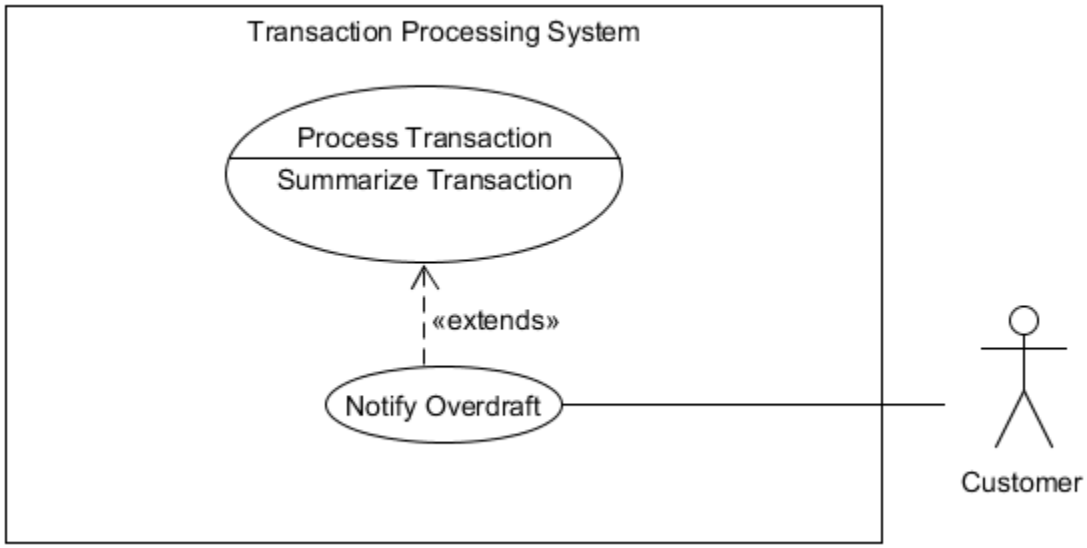

ATM
관리자
시스템 구동
«includes»
«includes»
시스템 종료
사용자인증
«includes»
«actor»
인증시스템
트랜잭션
«actor»
은행시스템
인출
사용자
예금
계좌조회

<<extend>> 관계



- 확장점 1, 2에서 extending use case가 Use Case의 행위를 확장한다
- 확장 관계는 행위를 추가하고자 할 때 사용한다

예


Transaction Processing System
Process Transaction
Summarize Transaction
«extends»
Notify Overdraft
Customer

Example <<extend>>

Use Case: ProcessTransactions

Brief Description: Process bank transactions collected for accounts.

Main Success Scenario:

1. At the end of the day (23:58), the use case starts.

Steps 2.-5. are done for each unprocessed transaction:

2. The System determines the account to which the transaction is applied.
3. The System applies the transaction to the account. The balance of the account is increased/decreased by the amount of the transaction.
4. The System records a log for the transaction information.
5. The System marks the transaction is processed.
6. {Summarize Transaction}

   A transaction summary is created for each account.

7. Use case ends.

Label for Extension Point

Example <<extend>>

Use Case: NotifyOverdraft

Brief Description: Notifies the Customer that his account has become overdrawn.

Pre-Condition: This service is only available if the Customer has purchased the overdraft notification service.

Main Success Scenario:

1. The use case starts as extension of ProcessTransactions at {Summarize Transactions} if the Customer has purchased the overdraft notification service and the set of completed transactions has caused the account to become overdrawn.
2. The System determines the Customer's preferred notification mechanism.
3. The System composes the overdraft notification.
4. The System transmits the overdraft notification to the Customer using the Customer's preferred notification mechanism.
5. The use case ends.

유스케이스 기술서 작성

❖ 개요

- **유스케이스 다이어그램을 보완하기 위한 산출물**
- **유스케이스 다이어그램과의 차이**
  - 유스케이스 다이어그램: 유스케이스는 시스템의 기능을 표현하는 것
  - 유스케이스 기술서: 각각의 유스케이스에 대해서 해당 유스케이스가 어떻게 수행되는지를 표현하는 수단

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.

번호 매기기

- 기본흐름을 작성할 때에는 기본 흐름을 구성하는 각 단계의 수행 주체가 액터인지 시스템인지가 명확하게 구분되는 것이 좋으며 각 수행단계의 순서를 1, 2, 3,... 등으로 구분한다.
- 만약 각 단계가 보다 구체화된 단계를 가진다면 (해당단계).1, (해당단계).2, 등으로 명시한다. 예를 들어 단계 3이 보다 구체화된 3개의 단계를 갖는다면 3.1, 3.2, 3.3로 기술한다.
- 대체흐름은 오류나 예외적인 사건이 발생하는 단계를 문자와 함께 기술한다. 예를 들면 기본흐름 2번 단계에서 사용자가 타당한 은행카드를 입력하지 않고 버스카드를 입력하였다면 다음과 같이 기술한다: 2a. 시스템이 인식하지 못하는 카드가 입력되는 경우

유스케이스 기술서 작성 (3/3)

❖ 유스케이스 기술서 예제

- **유스케이스명: 글을 등록한다**
- **액터명: 사용자**
- **유스케이스 개요 및 설명**
  - **사용자는 원하는 글을 게시판에 등록한다.**
- **작업 흐름**
  - **정상흐름**
    1. 로그인 유스케이스를 수행 한다.
    2. 사용자는 글쓰기 버튼을 선택한다.
    3. 시스템은 글쓰기 박스를 보여한다.
    4. 사용자는 글쓰기 박스에 원하는 글을 작성한다.
    5. 사용자는 등록 버튼을 선택한다.
    6. 시스템은 글을 데이터베이스에 저장한다.
  - **대체 흐름1**

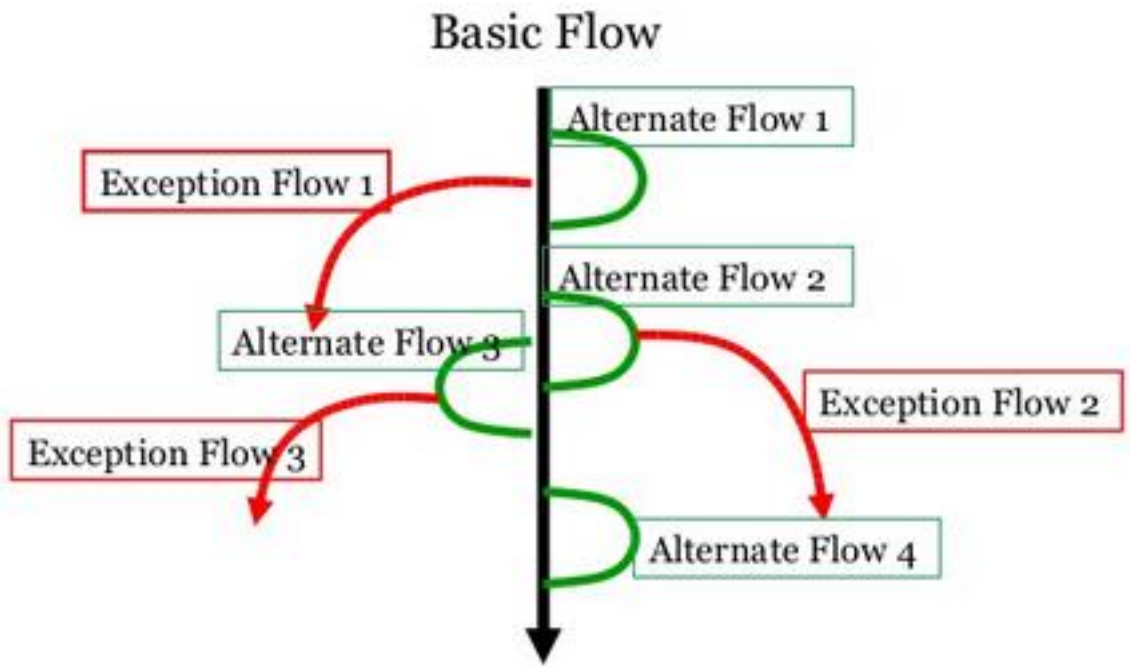
    4a. 등록 취소를 수행하는 경우, 게시판 목록 조회 화면을 표시한다.

    4b. 정상 흐름 4에서 글쓰기 박스에 글을 쓰지 않고 등록 원하는 경우, “내용을 넣으세요” 라는 메시지를 표시한다.

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 경고를 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.
대체흐름 3	...

대안 흐름과 예외흐름

- 대안 흐름은 유스케이스를 바로 종결지시키지 않고 기본 흐름과 다른 경로를 실행한다
- 예외 흐름은 유스케이스를 비정상적인 상태로 종결한다

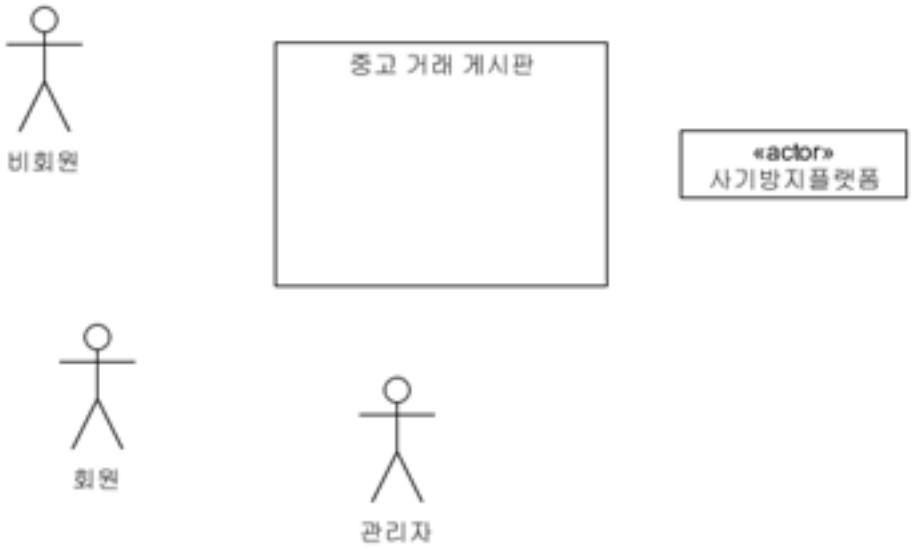


예제

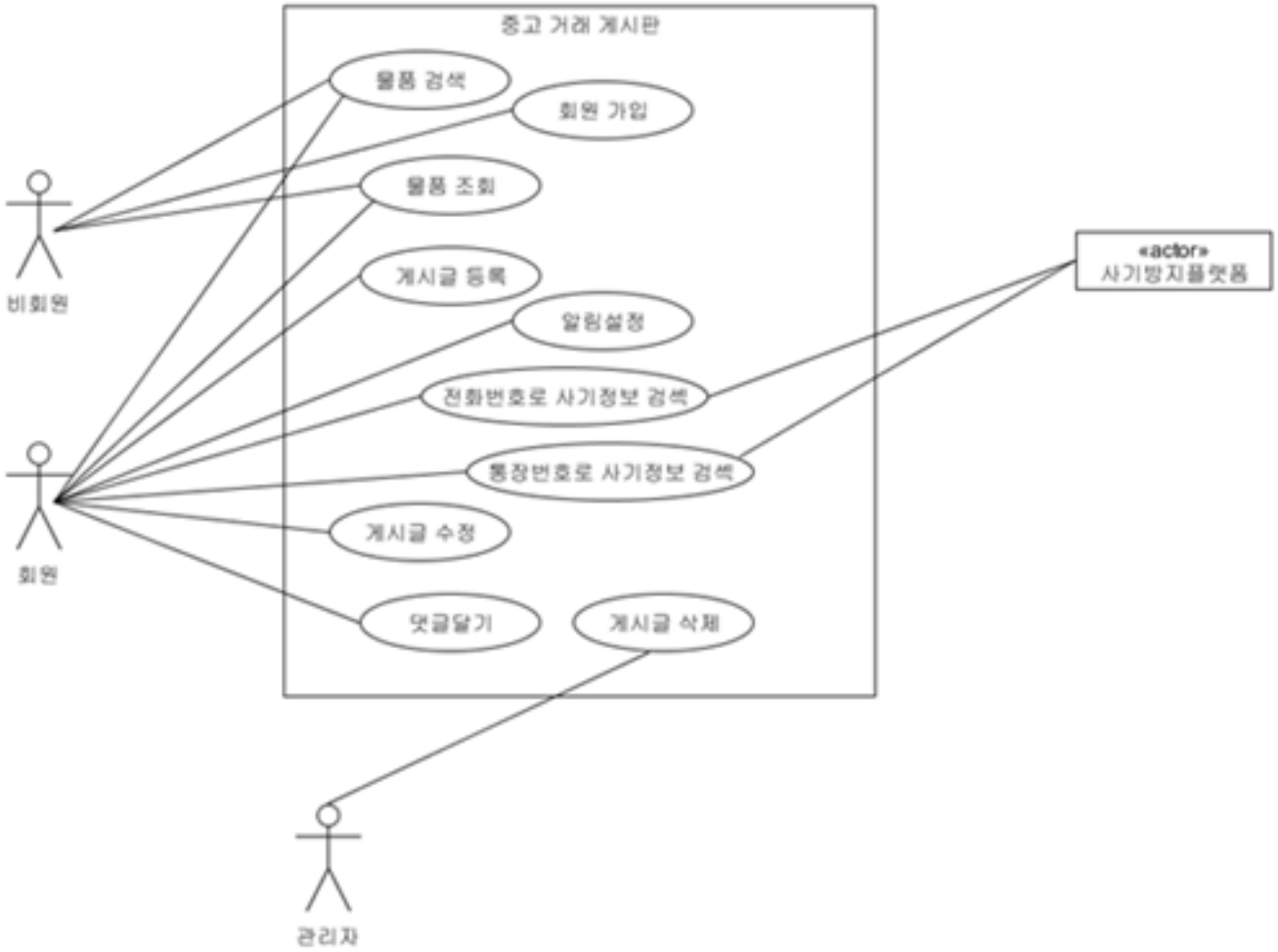
중고거래 게시판 명세

회원이 아닌 사용자는 물품에 대한 정보를 조회하거나 검색할 수 있으며 또한 회원 가입을 할 수 있다. 회원인 사용자는 물품에 대한 게시글에 댓글을 달 수 있으며 원하는 상품의 키워드를 등록하여 알림 설정을 할 수 있다. 또한 회원의 경우 판매나 구매 게시글을 작성하고 수정할 수 있다. 게시글을 작성할 때 경우에 따라 그림 파일을 첨부 할 수 있다. 그러나 일단 게시글이 등록되면 삭제할 수 없다. 삭제는 관리자만이 가능하다. 그리고 중고 물품의 안전한 거래를 위하여 판매자의 전화번호나 통장 번호를 사용하여 사기 이력 정보를 제공하는 시스템(사기방지플랫폼)의 서비스를 이용하여 판매자의 사기 이력정보를 조회할 수 있다. 회원과 관리자의 역할을 하려면 사용자 인증을 받아야 한다.

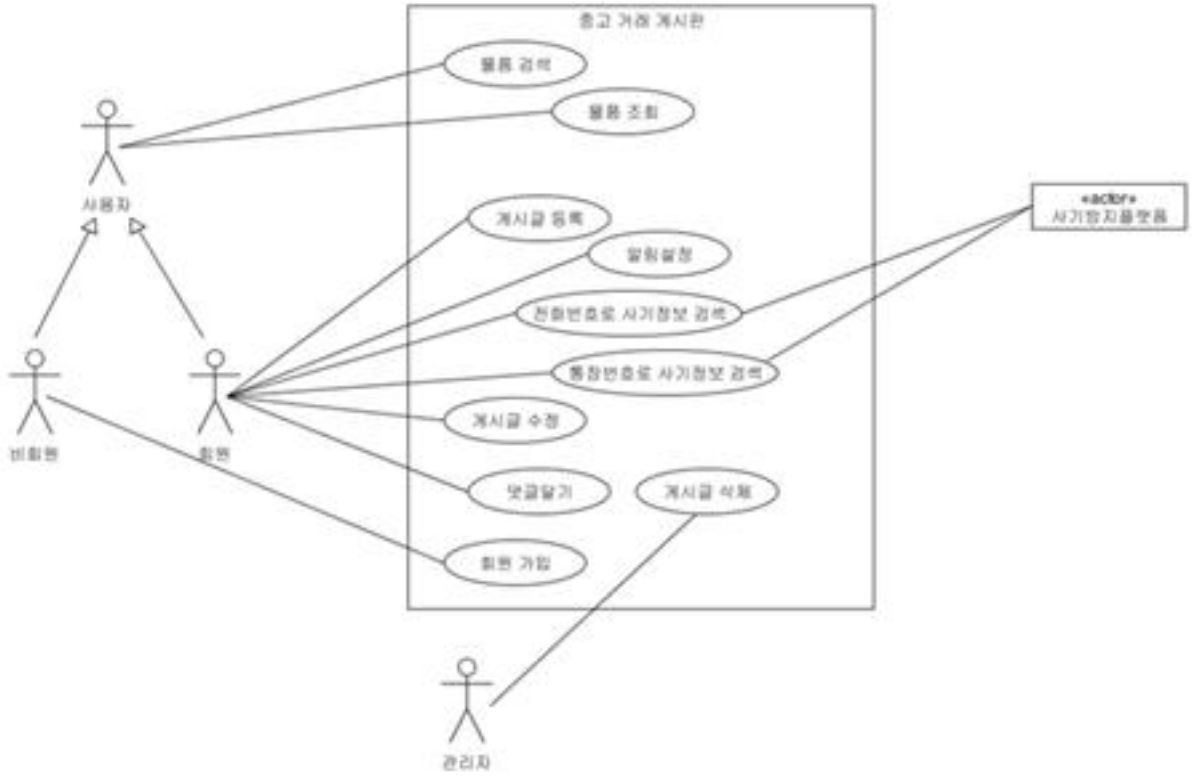
액터



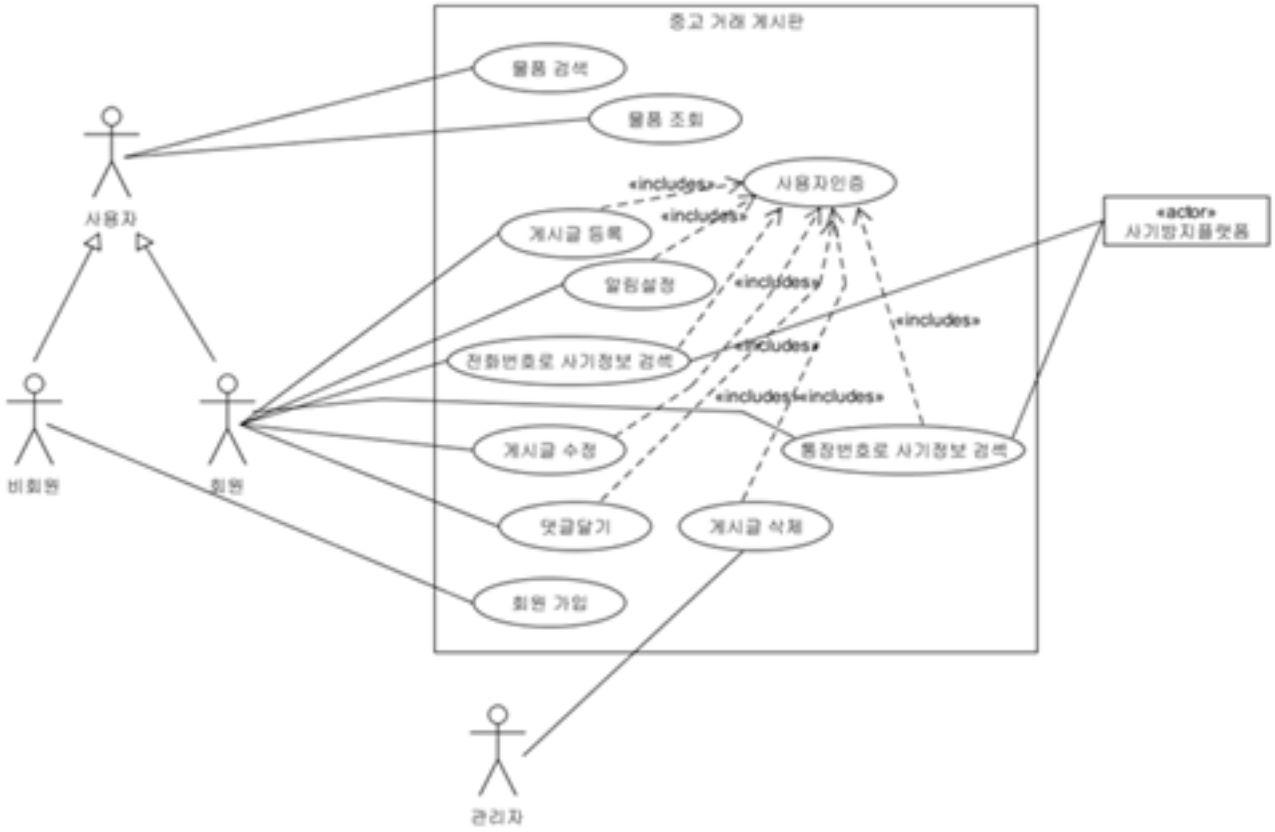
유스케이스


중고 거래 게시판
물품 검색
회원 가입
물품 조회
게시글 등록
알림설정
전화번호로 사기정보 검색
통장번호로 사기정보 검색
게시글 수정
댓글달기
게시글 삭제
비회원
회원
관리자
«actor»
사기방지플랫폼

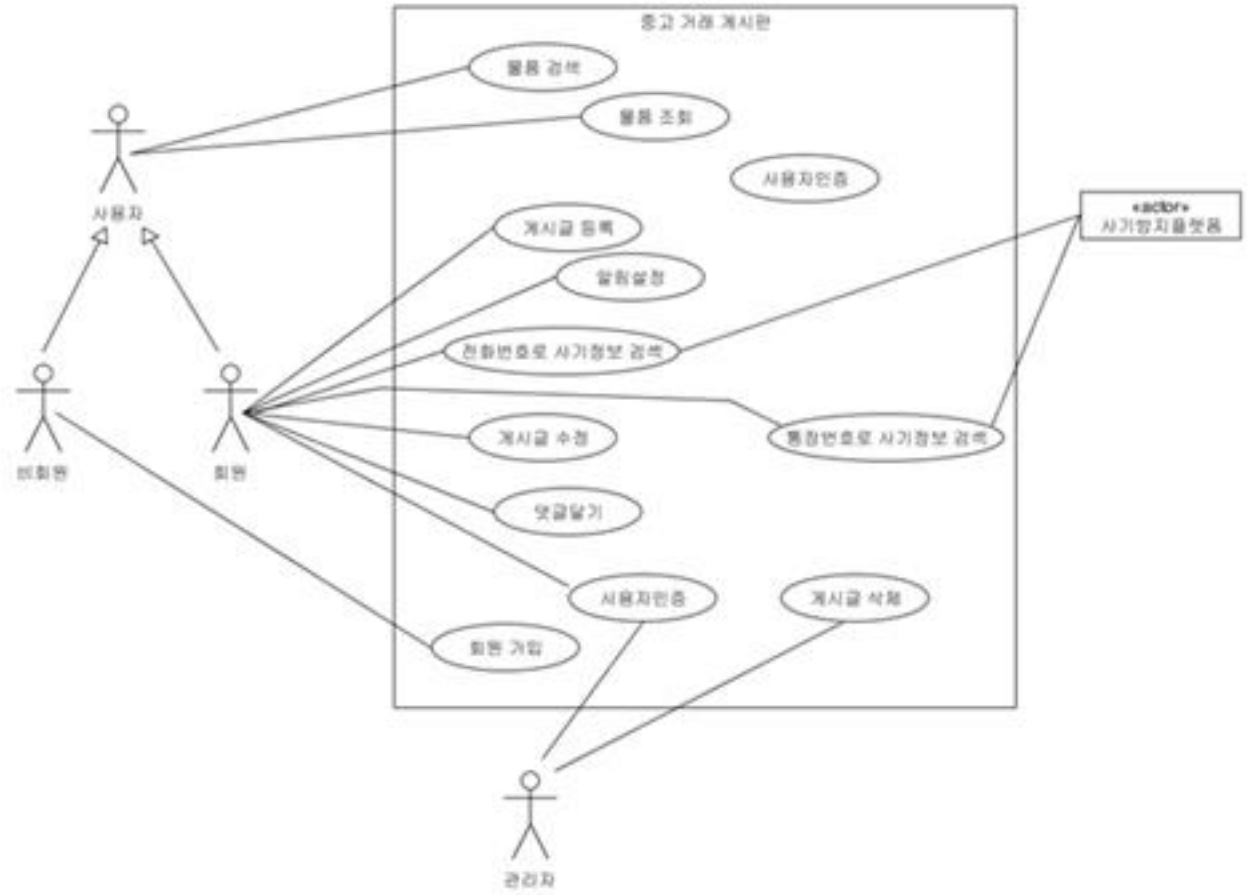
일반화 관계

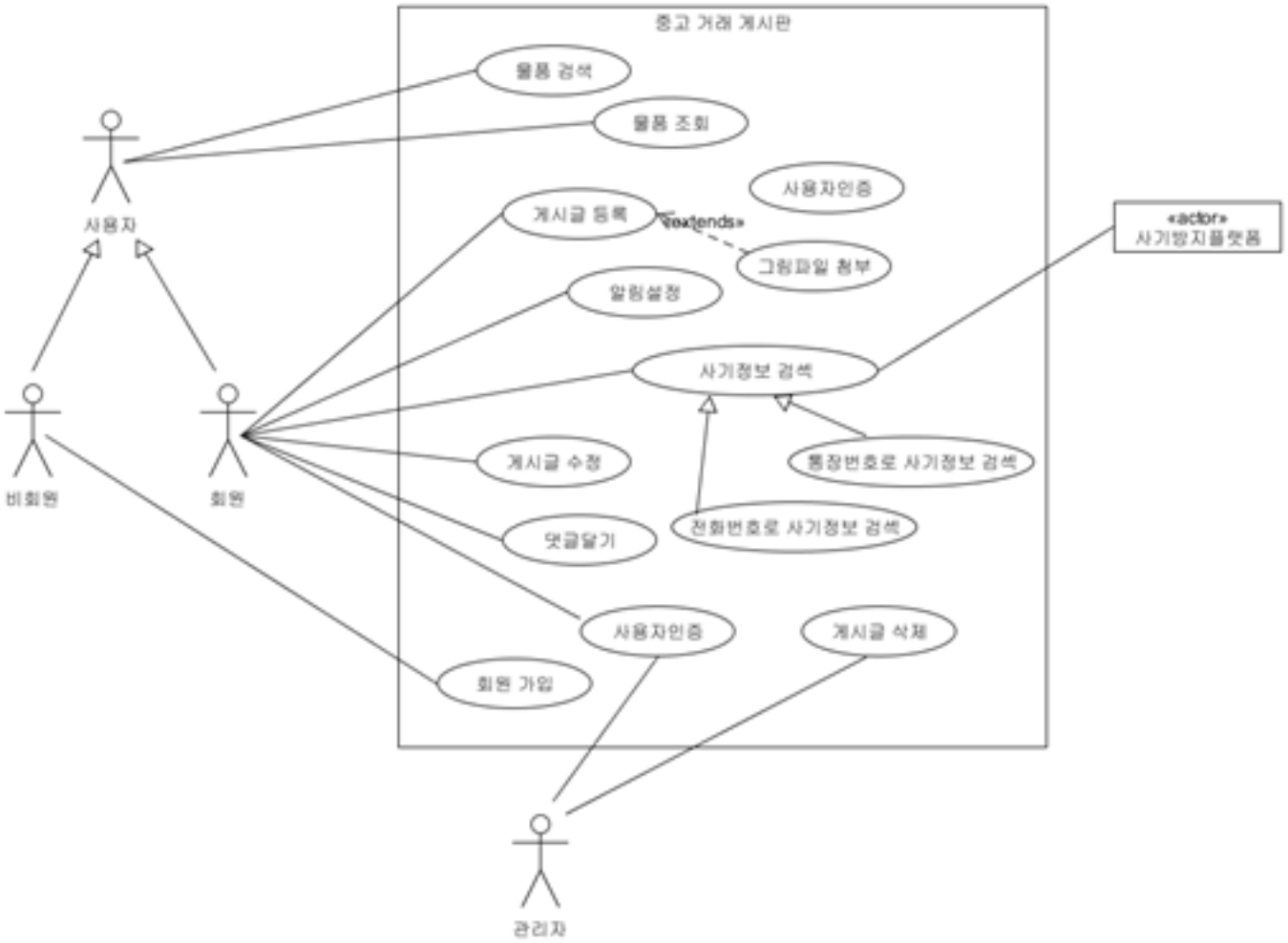

중고 거래 게시판
물품 검색
물품 조회
사용자
게시글 등록
알림설정
전화번호로 사기정보 검색
통장번호로 사기정보 검색
«actor»
사기방지용앱용
게시글 수정
비회원
회원
댓글달기
게시글 삭제
회원 가입
관리자

포함 관계

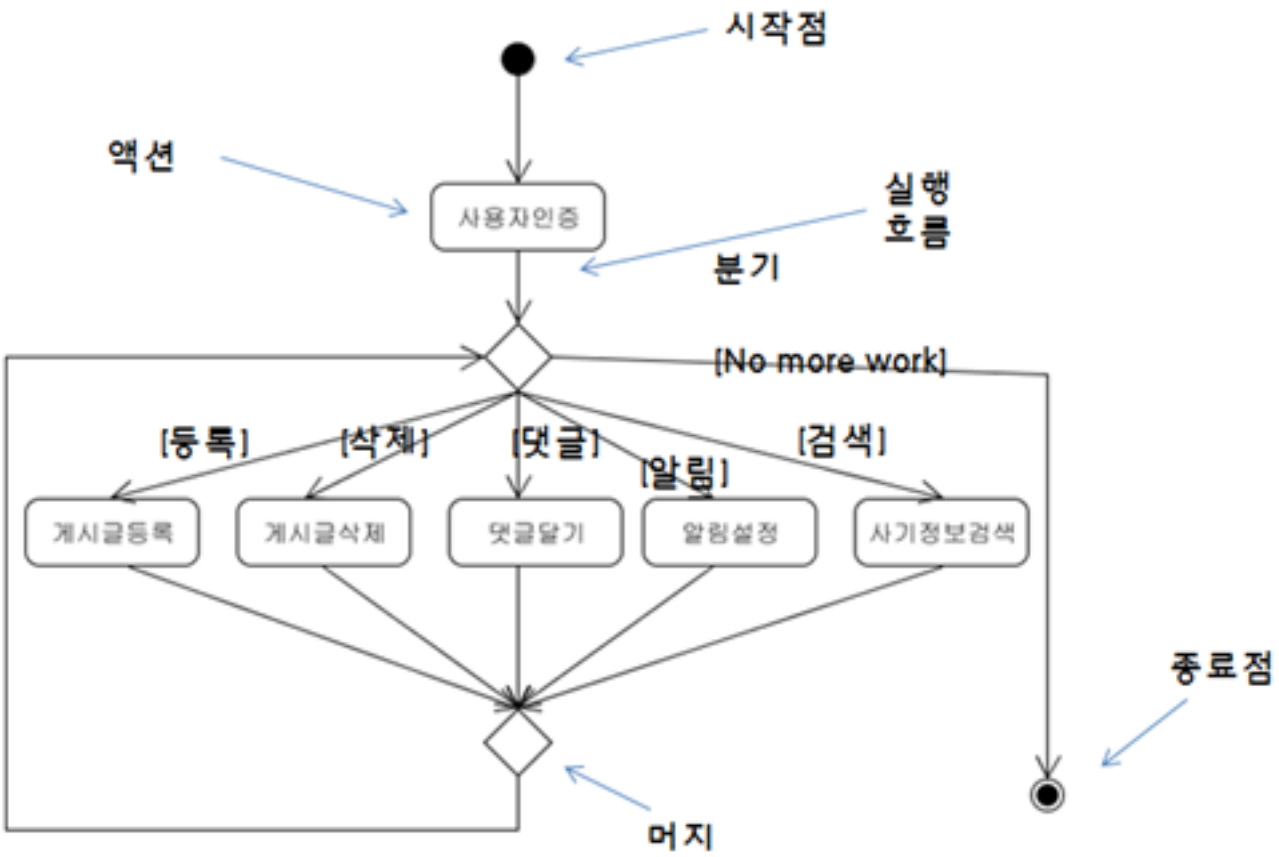

중고 거래 게시판
물품 검색
물품 조회
사용자인증
«includes»
«includes»
게시글 등록
알림설정
전화번호로 사기정보 검색
«includes»
«includes»
«includes»
«includes»
계시글 수정
통장번호로 사기정보 검색
댓글달기
게시글 삭제
회원 가입
«actor»
사기방지플랫폼
사용자
비회원
회원
관리자

수정


중고 거래 게시판
물품 검색
물품 조회
사용자인증
계시글 등록
알림설정
전화번호로 사기정보 검색
계시글 수정
통장번호로 사기정보 검색
댓글달기
사용자인증
계시글 삭제
회원 가입
사용자
비회원
회원
관리자
«actor»
사기방지플랫폼


중고 거래 게시판
물품 검색
물품 조회
사용자인증
게시글 등록
«extends»
그림파일 첨부
알림설정
사기정보 검색
«actor»
사기방지플랫폼
게시글 수정
통장번호로 사기정보 검색
댓글달기
전화번호로 사기정보 검색
사용자인증
게시글 삭제
회원 가입
사용자
비회원
회원
관리자

액티비티 다이어그램


시작점
액션
사용자인증
실행
흐름
분기
[No more work]
[등록]
[삭제]
[댓글]
[알림]
[검색]
게시글등록
게시글삭제
댓글달기
알림설정
사기정보검색
종료점
머지