5장. 객체지향방법론

숙명여자대학교
SOOKMYUNG WOMEN'S UNIVERSITY

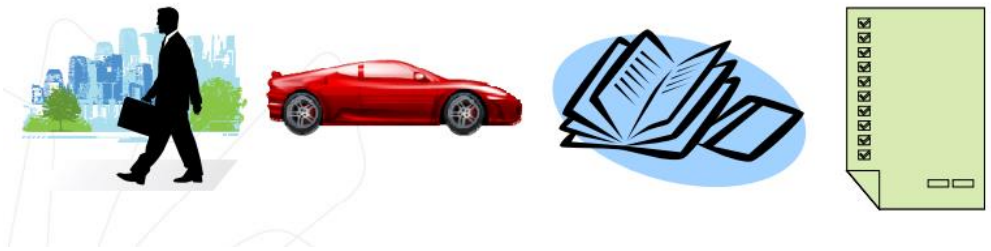
Contents

1. 객체지향 개념

- 객체지향 방법론
  - 현실세계의 개체(Entity)를 속성과 메소드를 결합시킨 객체 형태로 표현하는 개념으로 객체 간의 메시지 통신을 통해 시스템을 구현하는 개발방법
- 객체지향의 핵심개념은 객체와 클래스
- 객체지향의 기본원리
  - 캡슐화(Encapsulation)
  - 정보은닉(Information hiding)
  - 상속(Inheritance)
  - 다형성(Polymorphism)

1.1 객체지향의 핵심개념 (1/4)

- 객체(Object)는 현실세계에 존재하거나 생각할 수 있는 개념을 표현
  - 사람, 자동차, 은행계좌, 주문, 계좌이체 등



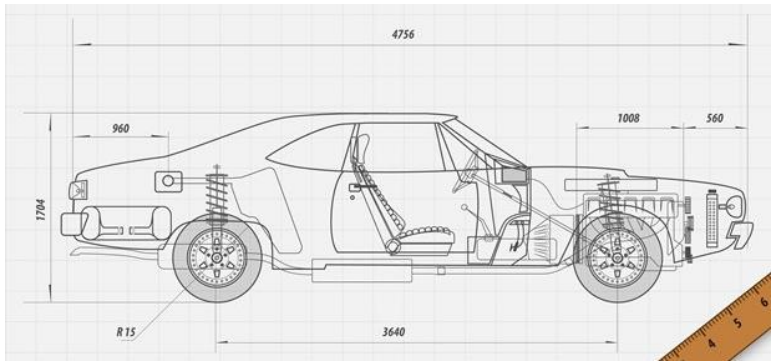
  - 물리적 객체 & 개념적 객체

1.1 객체지향의 핵심개념 (2/4)

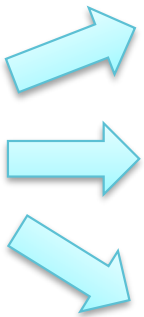
- 객체(object)가 되려면 ?
  - 상태를 가져야 한다
    - 객체가 가지는 자료 값이 상태를 표현함
    - 시간이 흐르면서 변할 수 있음
    - 예: 강좌(course)의 상태 – “open”, “close”
  - 객체는 잘 정의된 오퍼레이션이 있어야 함
    - 객체의 오퍼레이션은 동작을 결정
    - 객체 외부에서 요구할 때 오퍼레이션을 호출
    - 예: 강좌 시간 결정, 수강생 추가, 수강생 삭제
  - 고유한 Identity가 있어야 함 : 객체를 구별할 수 있는 수단

1.1 객체지향의 핵심개념 (3/4)

- 클래스(class) : 객체의 설계도면
  - 공통의 속성과 행위를 가진 객체를 묶어 추상화한 개념
  - 인스턴스(Instance) : 클래스에서 파생된 하나의 실제 객체 (objects=instances of classes)



클래스



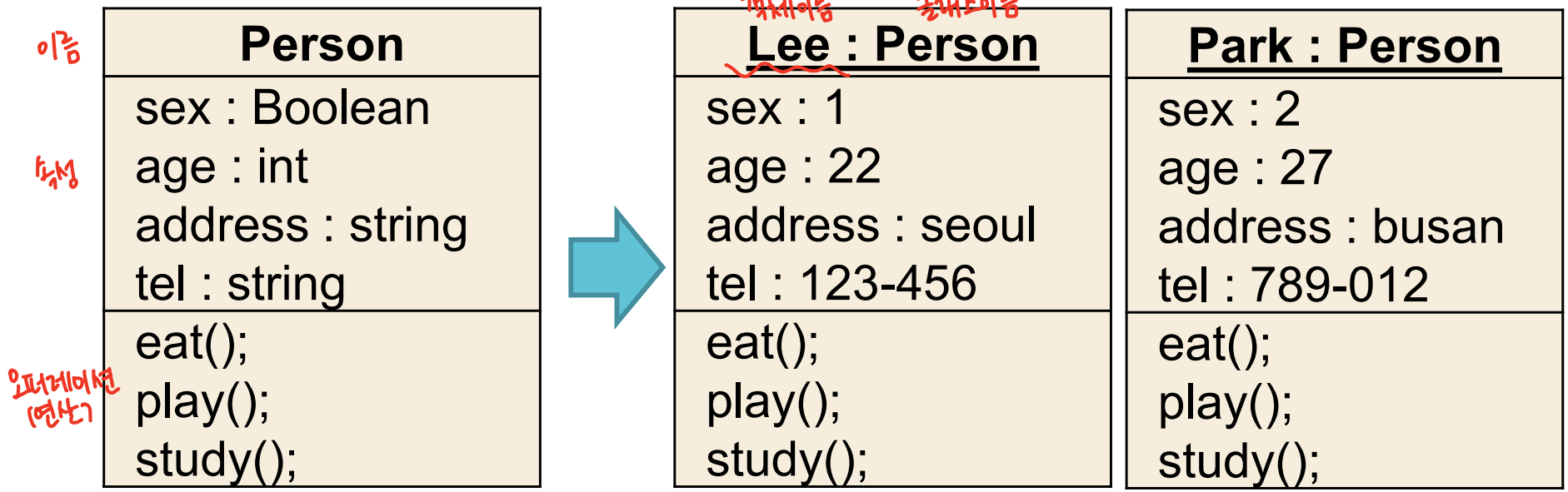
객체

객체

객체

1.1 객체지향의 핵심개념 (4/4)

- 클래스와 객체 → UML에서 다르표현
  - 클래스는 개념적인 의미이며, 객체는 구체적인 의미
  - 한 클래스에서 생성된 객체들은 같은 속성과 같은 연산에 대한 정의를 가짐

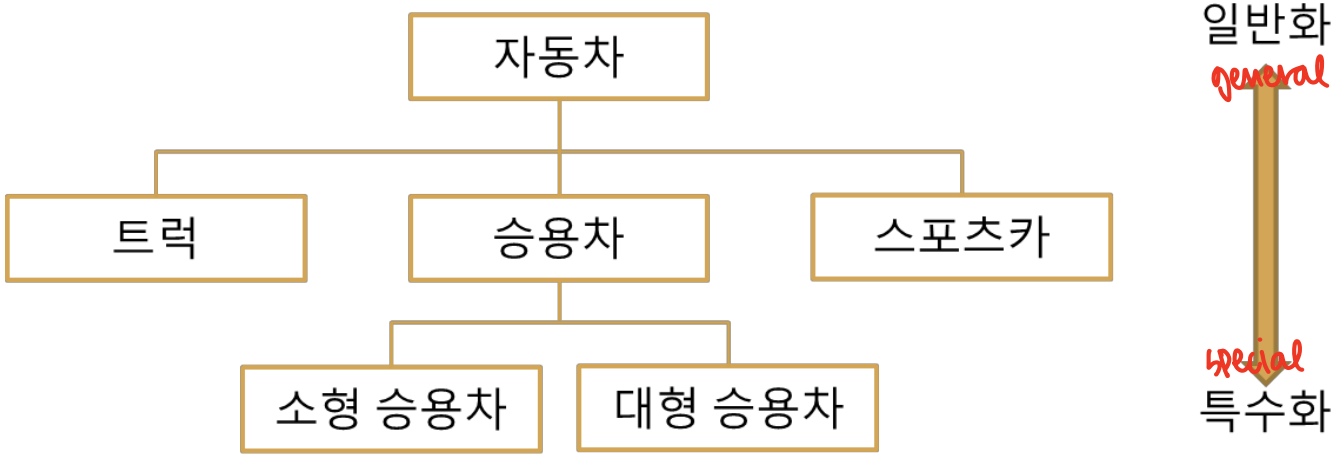


1.2 객체지향의 원리 (1/4)

- 상속(Inheritance)

캡슐화.정보은닉은 객체지향이 아니어도 가능!

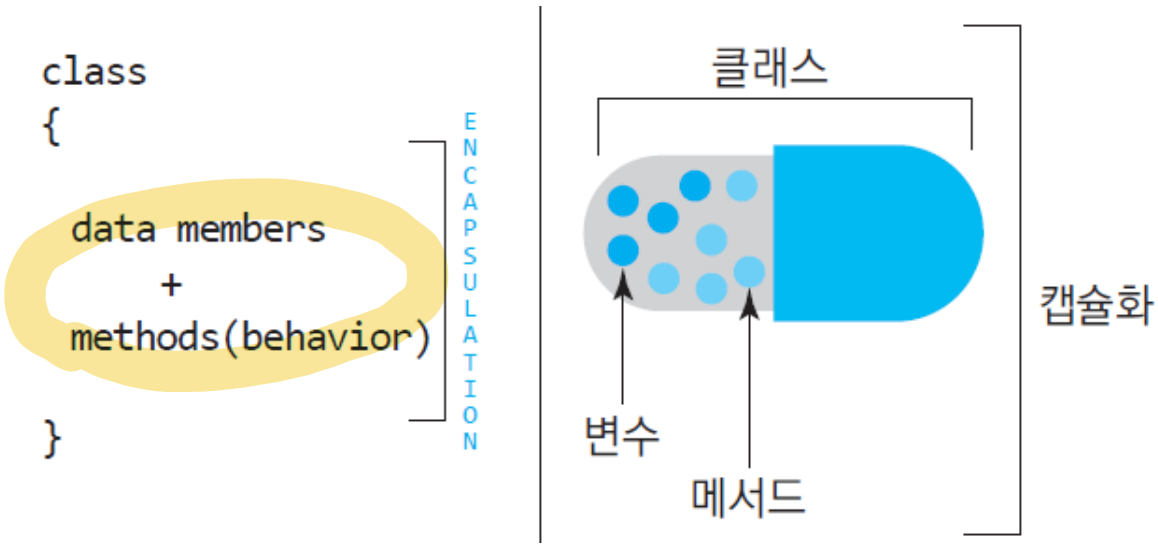
  - 객체지향 패러다임만이 갖는 고유 특성
  - 클래스 간의 계층구조를 통해 상위클래스의 모든 특성을 물려받는 것
  - 이미 정의한 클래스를 재사용하고, 확장 할 수 있도록 지원하는 개념



1.2 객체지향의 원리 (2/4)

구조적방법론으로개발 → 데이터중심으로 기능 별 → 접근제한

- 캡슐화(encapsulation)
  - 객체에 대한 외부로부터 접근을 제한 → private / public → 공개된부분(인터페이스)으로 접근
  - 관련된 항목을 모아서 하나의 단위로 취급하는 것
  - 객체내부의 변경이 시스템 전체에 주는 영향을 최소한으로 억제



1.2 객체지향의 원리 (3/4)

캡슐화와 거의 같은기능

- 데이터 은닉 (information hiding)
  - 캡슐화와 비슷한 개념으로 객체의 상세한 내용을 객체 외부에 철저히 숨김
  - 단순히 메시지만으로 객체와의 상호 작용을 하게 만듦 ↳ 공개된 인터페이스
  - 외부에서 알아야 하는 부분만 공개하고, 나머지는 숨김으로써 대상을 단순화시키는 효과가 있음 (독립성, 유지보수성, 이식성) ⇒ 추상화의 원리와 같음
  - 객체지향언어의 접근제어방식으로 구현
    - public/private/protected 접근제어자

1.2 객체지향의 원리 (4/4)

- 다형성(polymorphism)
  - 프로그램 언어의 각 요소들(상수, 변수, 식, 오브젝트, 함수, 메소드 등)이 다양한 data type에 속하는 것이 허가되는 성질
  - 즉, 서로 다른 객체가 동일한 메시지에 대하여 서로 다른 방법으로 응답할 수 있는 기능
    - Overriding : 같은 이름의 메소드가 여러 클래스에서 다른 기능을 하는 것
    - Overloading : 같은 이름의 메소드가 인자의 개수나 자료형에 따라서 다른 기능을 하는 것

1.3 UML의 이해 (1/5)

- UML(Unified Modeling Language) 은 대표적인 모델링 언어
- 1990년 초기 객체지향 방법론 등장하면서, 표기법의 차이가 문제로 대두 ⇒ OMT + OOSE + OOAD
- 1997년 OMG1)에 의해 표준 채택
- 가장 최근 버전 : 2017년 발표된 UML 2.5.1

1) Object Management Group : 컴퓨터 산업 표준화 컨소시움. http://www.omg.org

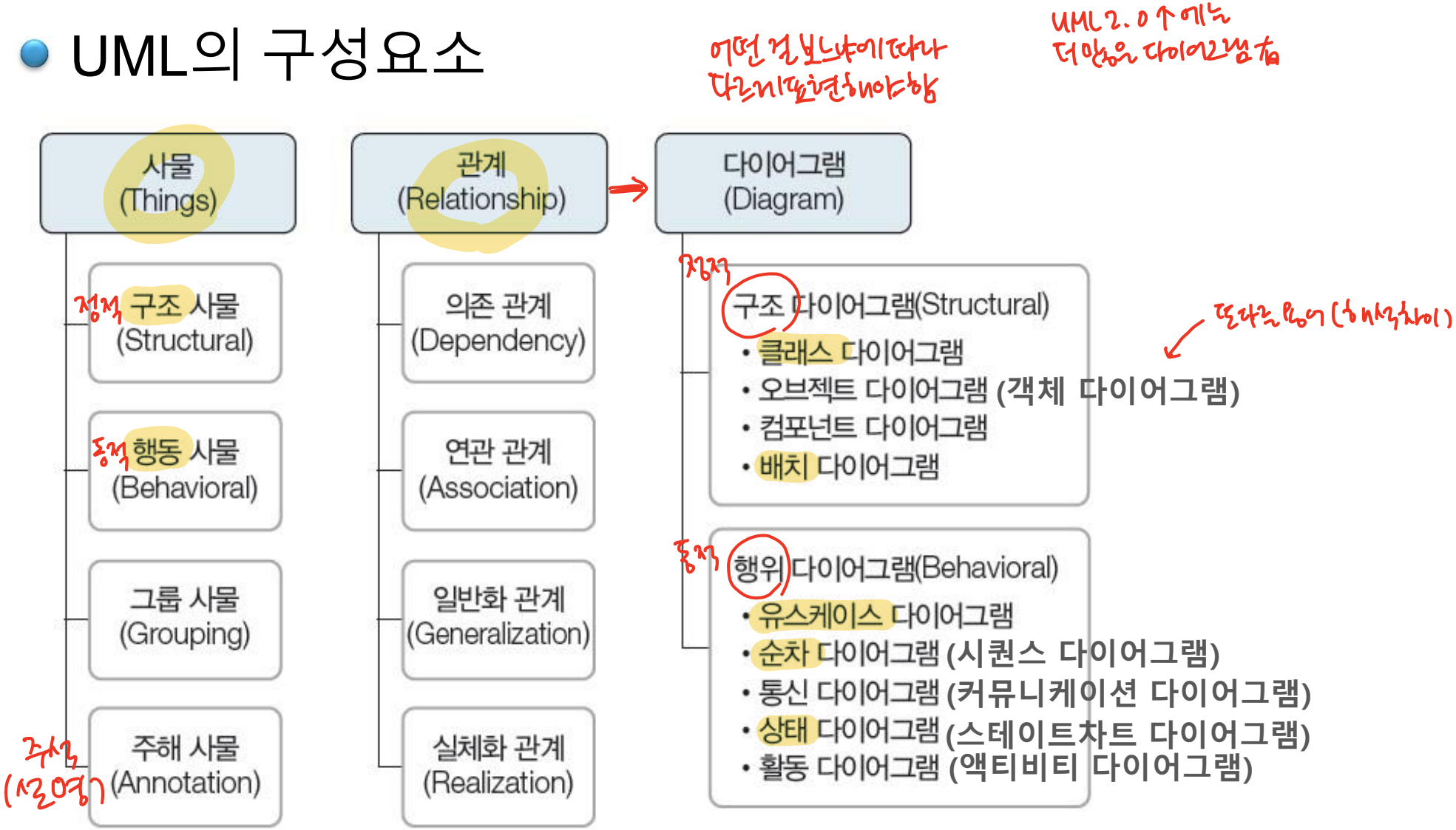
1.3 UML의 이해 (2/5)

- UML(Unified Modeling Language)의 특징
  - 가시화
    - SW 개념 모델을 시각적인 그래픽 형태로 표기
    - 표기법에 사용하는 기호에 명확한 정의를 부여하는 것
  - 명세화 요구사항이 빠짐없이 포함돼야하듯이
    - 정확하고, 명백하며, 완전한 모델을 만드는 것
    - 시스템 아키텍처와 모든 상세 내역에 대한 문서화 그래픽의 한계보완
  - 구축
    - UML로 명세화 된 설계 ⇔ 소스코드

자동전환 !! ⇒ 객체지향방법론은 분석·설계가 따로 구분되지 않음
클래스다이어그램 → 코드
↖
설계문서

1.3 UML의 이해 (3/5)

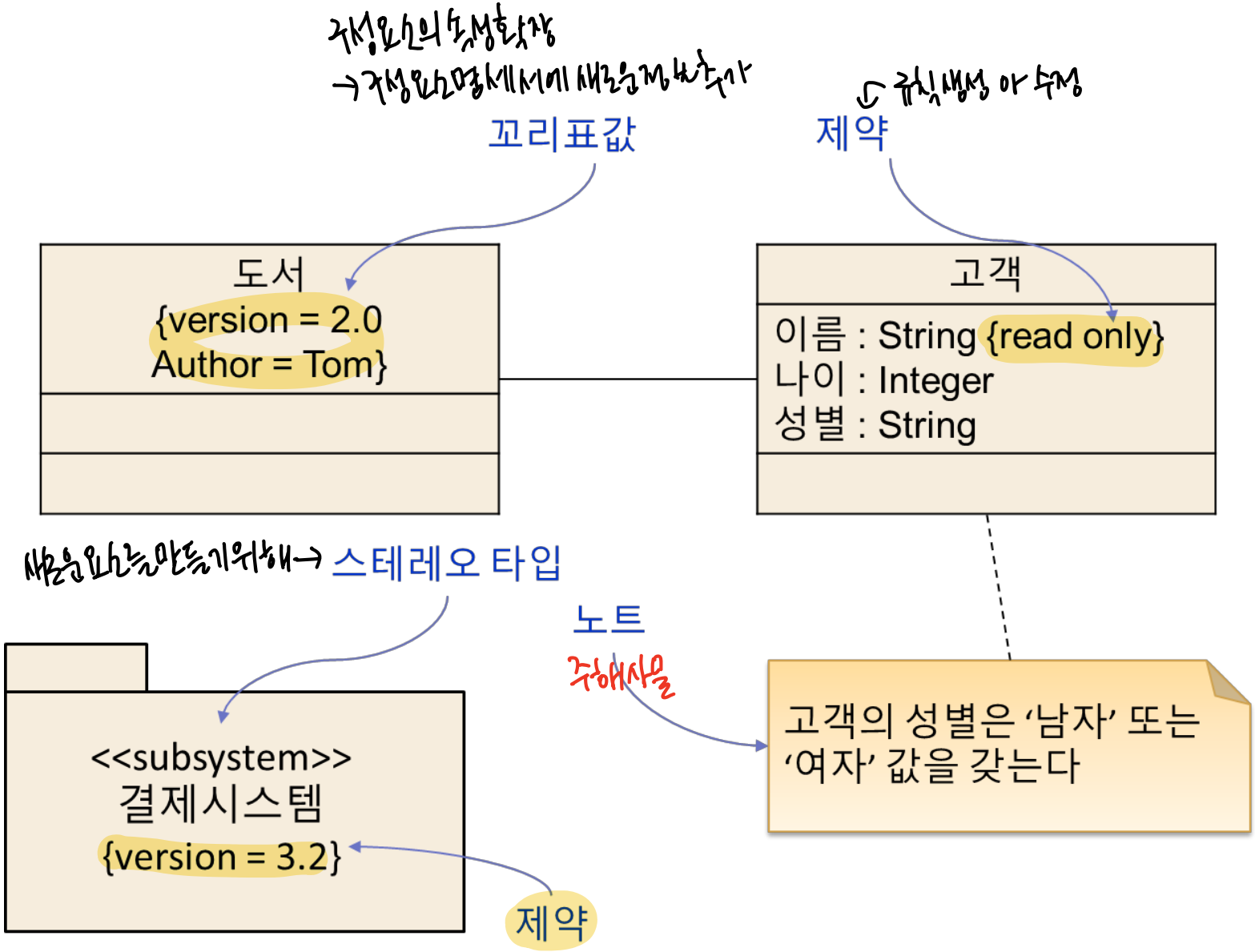
- UML의 구성요소



1.3 UML의 이해 (4/5)

- UML 확장 메커니즘
  - 스테레오타입(stereotype) : 기본적인 요소 외에 새로운 요소를 만들기 위함. (<< >> 기호를 사용)
  - 꼬리표값(tagged values) : 구성 요소가 갖는 속성을 확장하여 구성요소의 명세서에 새로운 정보를 추가 (tag = value 기호를 사용)
  - 제약(constraints) : 구성 요소에 있는 이전의 규칙을 수정하거나 새롭게 생성하기 위함 ({ } 기호를 사용)

1.3 UML의 이해 (5/5)


재정의의 도입확장
→ 재정의명세서에 새로운 내용추가
꼬리표값
제약
↳ 규칙생성 or 수정
도서
{version = 2.0
Author = Tom}
고객
이름 : String {read only}
나이 : Integer
성별 : String
새로운 요소를 만들기 위해 → 스테레오 타입
노트
주의사항
<<subsystem>>
결제시스템
{version = 3.2}
제약
고객의 성별은 '남자' 또는
'여자' 값을 갖는다

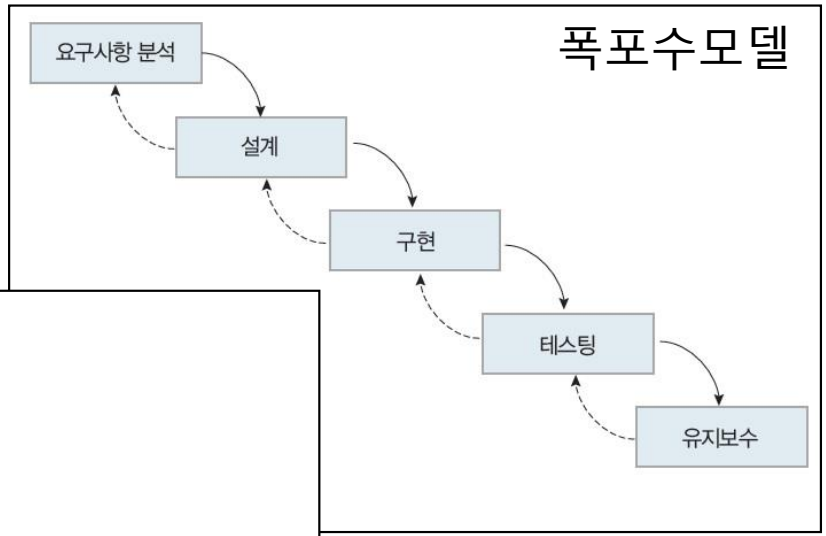
2. 객체지향방법론의 특징 (1/4)

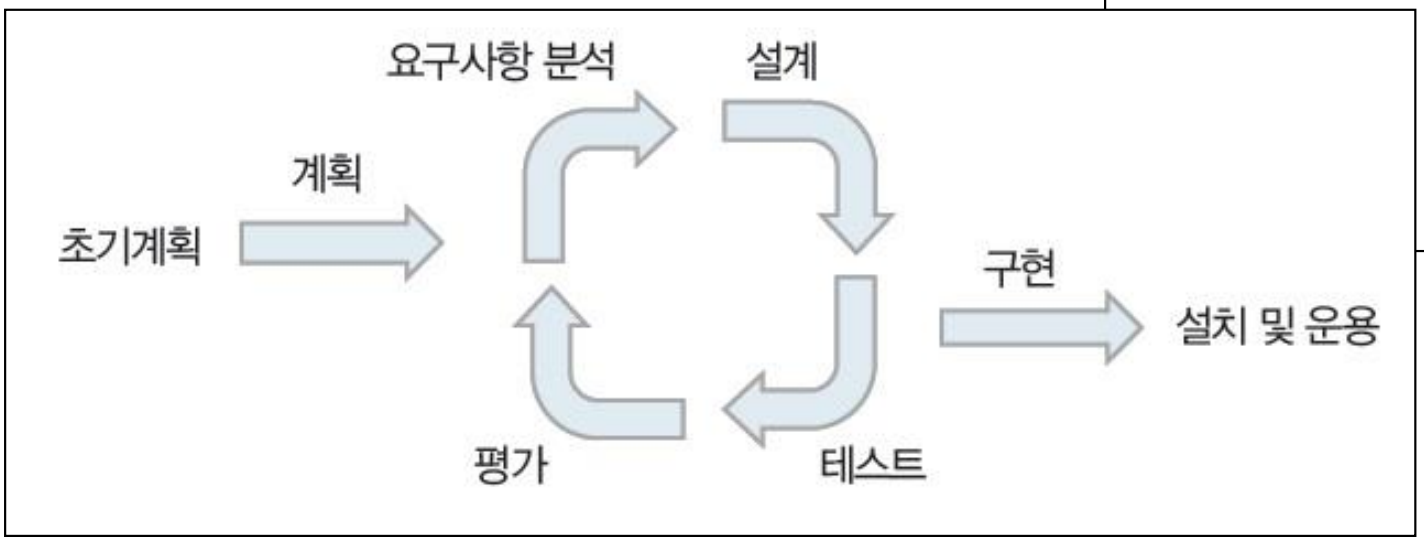
전에 배운내용

- 반복적인 프로세스 → 폭포수방식!
- 솔기 없는 프로세스 심리스. 경계 블랙홀
- 상향식 접근 방식
- 재사용을 고려

2. 객체지향방법론의 특징 (2/4)

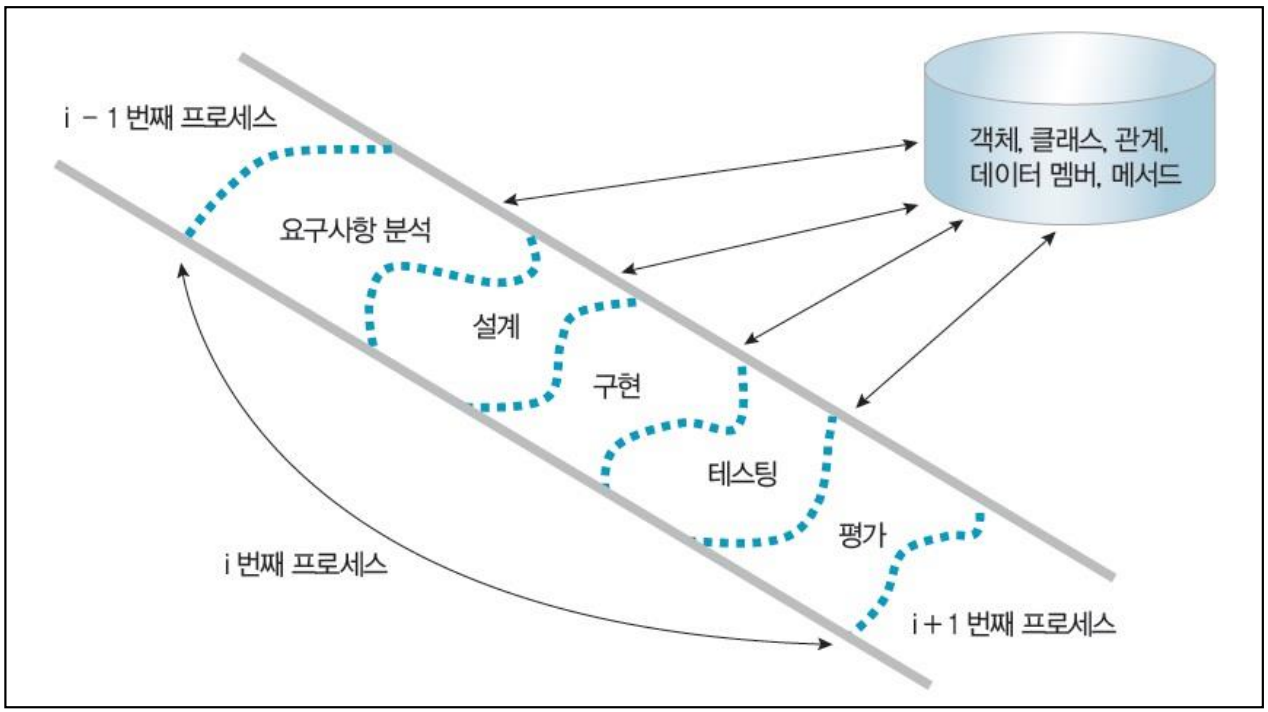
- 반복적인 프로세스
  - 객체지향개발은 소프트웨어 생명주기를 반복하여 적용하도록 제안





2. 객체지향방법론의 특징 (3/4)

- seamless 프로세스
  - 프로세스를 구성하는 각 단계 간의 경계선이 불분명하다는 것을 뜻함



2. 객체지향방법론의 특징 (4/4)

- 상향식 프로세스
  - 구조적 방법론 : 하향식(Top Down) 프로세스를 사용
  - 객체지향 방법론 : 상향식 프로세스를 사용

- 재사용에 대한 고려
  - 구조적 방법론 : 성공적인 소프트웨어 산물을 만들고자 하는 개발 공정에 치중
  - 객체지향 방법론 : 재사용이 고려되어 프로세스가 진행

객체지향방법론
① 기능모델링 → 유스케이스 다이어그램
② 구조모델링
③ 행위모델링

3. 기능 모델링 개요 (1/2)

- 기능모델링 : 소프트웨어 분석을 위한 객체지향 방법론의 첫번째 활동
  - 사용자로부터 도출한 요구사항으로부터 소프트웨어 시스템이 해야 할 기능이 무엇인지를 식별해가는 과정
  - 유스케이스 다이어그램 : 시스템과 관련된 외부 요소들을 표현 (기능들)
  - 유스케이스 작성 : 시스템이 사용자에게 눈에 보이는 결과를 생성하기 위해 수행하는 작업을 순서대로 정리 해 놓은 것

유스케이스 다이어그램 + 유스케이스 + 외

3. 기능 모델링 개요 (2/2)

- 유스케이스 작성의 목적
  - 외부 액터에 의해 인식되는 시스템의 기능과 요구사항을 보여주기 위함
  - 시스템의 범위를 정하는데 도움이 됨
  - 개발 과정을 계획하는데 사용
  - 요구를 개발하고 검증하는데 사용
  - 테스트케이스를 정의하는데 기초가 됨
  - 사용자 매뉴얼 구성하는데 사용될 수 있음

3.1 유스케이스 다이어그램 (1/3)

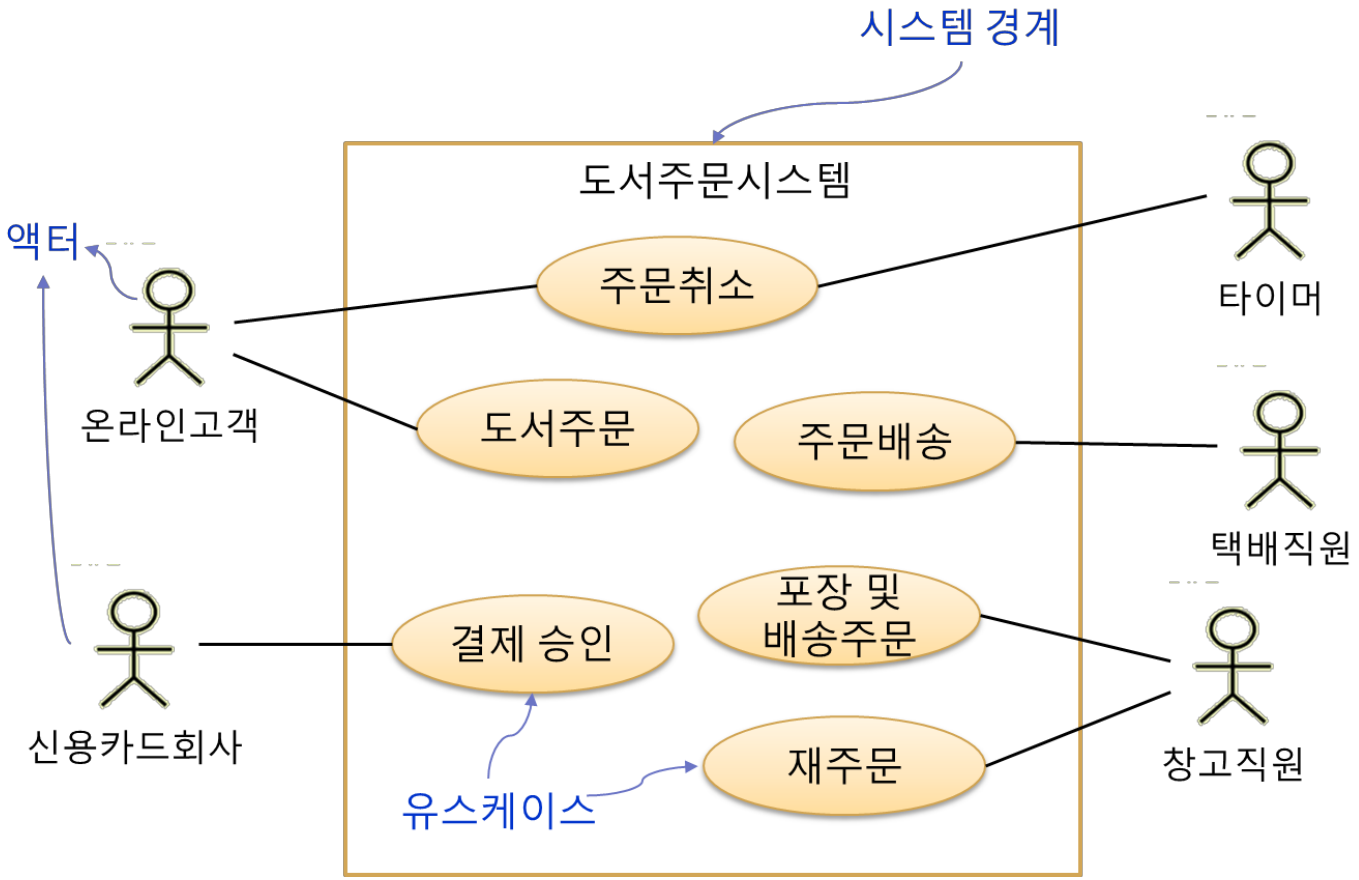
비기능적 요구사항 나타나지 X

- 시스템 최상위 수준에 해당하는 기능을 사용자 관점으로 나타냄
  - 사용자가 시스템을 통해 제공받는 주요 기능과 시스템과 사용자 간의 상호작용을 표현
- 유스케이스(UseCase, 사용사례, 쓰임새)
  - 외부에서 본 시스템의 뷰
  - 사용자의 관점에서 시스템이 제공하는 서비스를 나타낸 것
  - 시스템과 외부 사용자(액터, actor) 사이의 목표지향적인 인터랙션의 집합

기능에는 목표가있으므로

3.1 유스케이스 다이어그램 (2/3)

- 유스케이스 다이어그램 구성요소
  - 액터(actor), 유스케이스(usecase), 관계



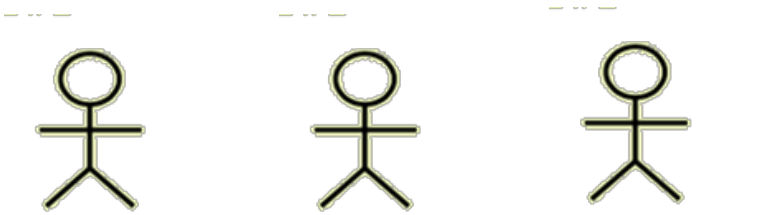
3.1 유스케이스 다이어그램 (3/3)

- 작성 절차
  - 요구사항 정의 내용 검토
  - 요구사항을 토대로 시스템의 경계를 결정
  - 경계가 설정된 시스템에 대한 명칭(시스템 명)을 정의
  - 시스템 외부에 존재하는 액터를 식별하고, 각 액터의 역할 정의
  - 각 액터가 시스템에서 사용하는 기능을 식별
  - 식별된 기능(유스케이스)과 액터 간의 관계를 정의
  - 유스케이스 명세서(Use Case Description)를 작성

=유스케이스작성

3.1.1 액터

- 시스템으로부터 서비스를 받을 필요가 있는 외부 요소
  - 사람(역할), 시스템, 조직을 표현
  - 외부에 존재하는 상호작용 대상
  - 역할을 중심으로 식별
  - 예) James라는 사람이 시스템 사용자이면서 시스템 관리자 역할을 할 때, 액터는 사용자와 관리자의 두 가지로 식별



3.1.2 유스케이스

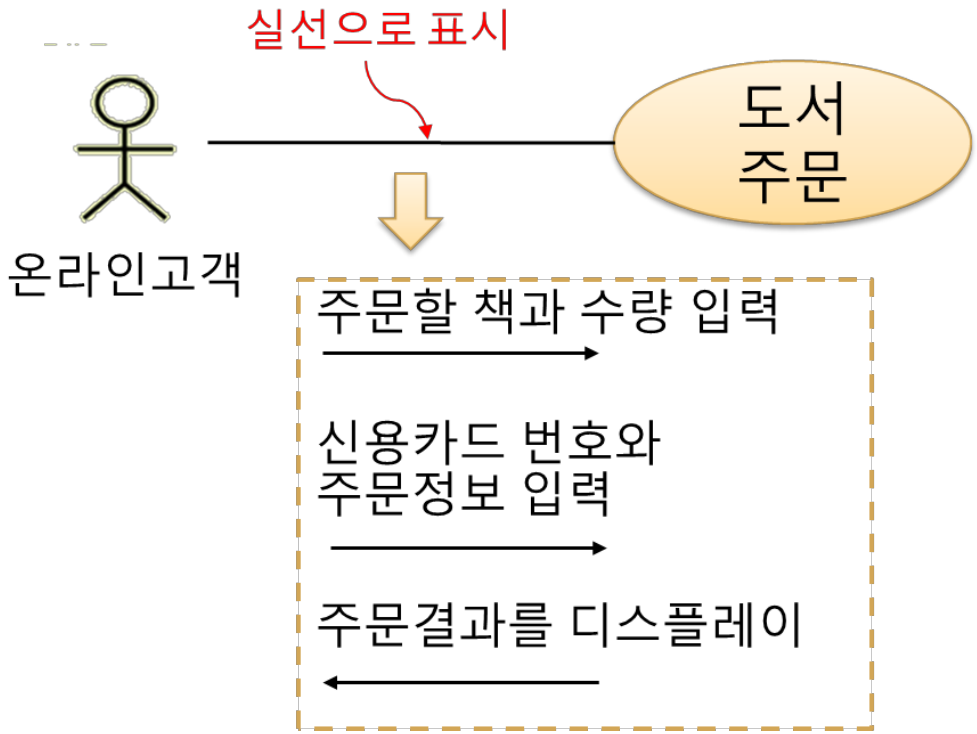
- 액터에게 서비스를 제공하기 위하여 시스템이 수행하는 중요 프로세스
- 유스케이스 이름은 작업을 의미하는 동사구 형태
  - 예: 도서주문, 회원가입, 처방전발급
- 유스케이스는 액터에 의해 시작되어야 함
- 단일 유스케이스가 여러 액터에 의하여 구동 될 때
  - 작업이 동일하면 같은 유스케이스
  - 이벤트 흐름이 다르면 다른 유스케이스
- 하나의 유스케이스는 동작 수행에 대한 종료를 포함하도록 정의

3.1.3 관계 (1/4)

- 유스케이스 다이어그램의 관계
  - 연관관계(association) : 커뮤니케이션 관계로 유스케이스와 액터 사이의 관계
  - 의존관계(dependency) : 유스케이스들 사이의 관계
    - 포함관계
    - 확장관계
  - 일반화 관계(generalization) : 유스케이스와 유스케이스, 또는 액터와 액터 사이의 관계

3.1.3 관계 (2/4)

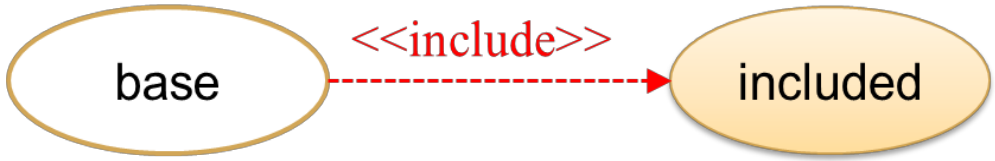
- 연관 관계
  - 액터와 유스케이스 사이에 상호작용이 있음을 표현
  - 유스케이스들 사이에는 연관관계가 나타나면 안됨



3.1.3 관계 (3/4)

의존 中

- 포함(include) 관계 유스케이스끼리
  - 공통의 유스케이스를 별도로 정의
  - base 유스케이스를 실행하기 위해 반드시 included 유스케이스가 실행되어야 함

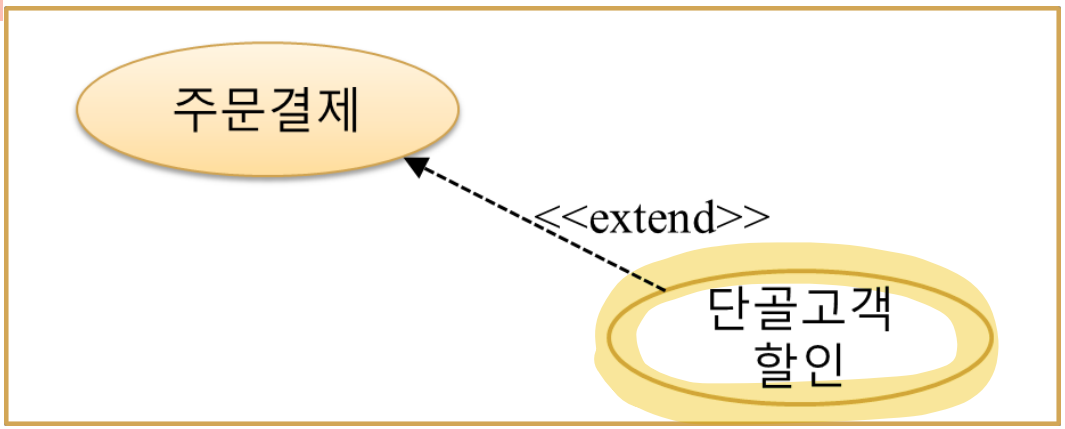


3.1.3 관계 (4/4)

의존中

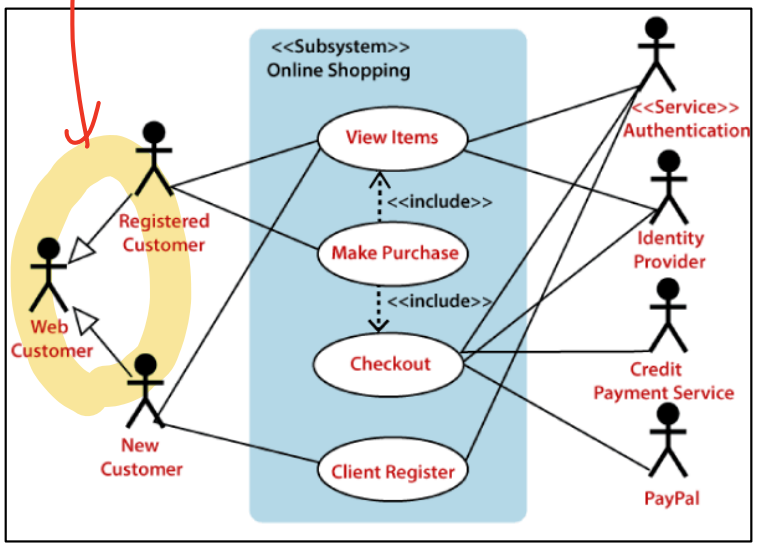
- 확장(extend) 관계 유스케이스끼리
  - 존재하는 유스케이스의 동작을 조건적으로 확장
  - 이벤트의 추가나 예외적인 케이스

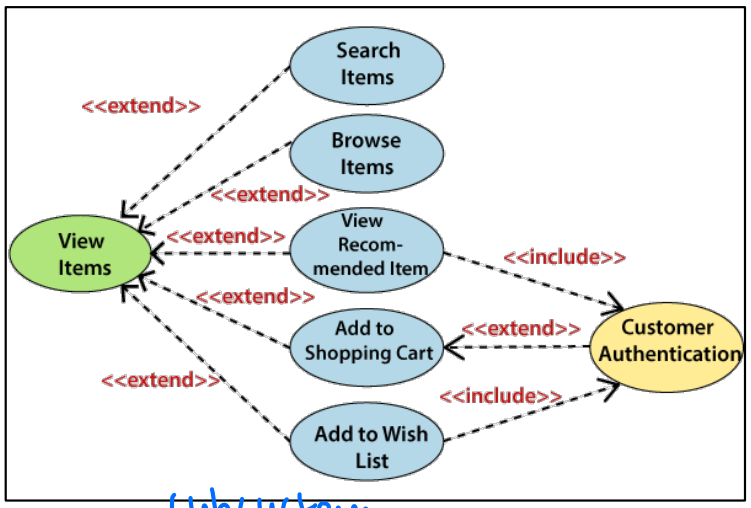
① 확장관계는 유스케이스 사이의 관계
→ 액터와 유스케이스 사이는 불가능



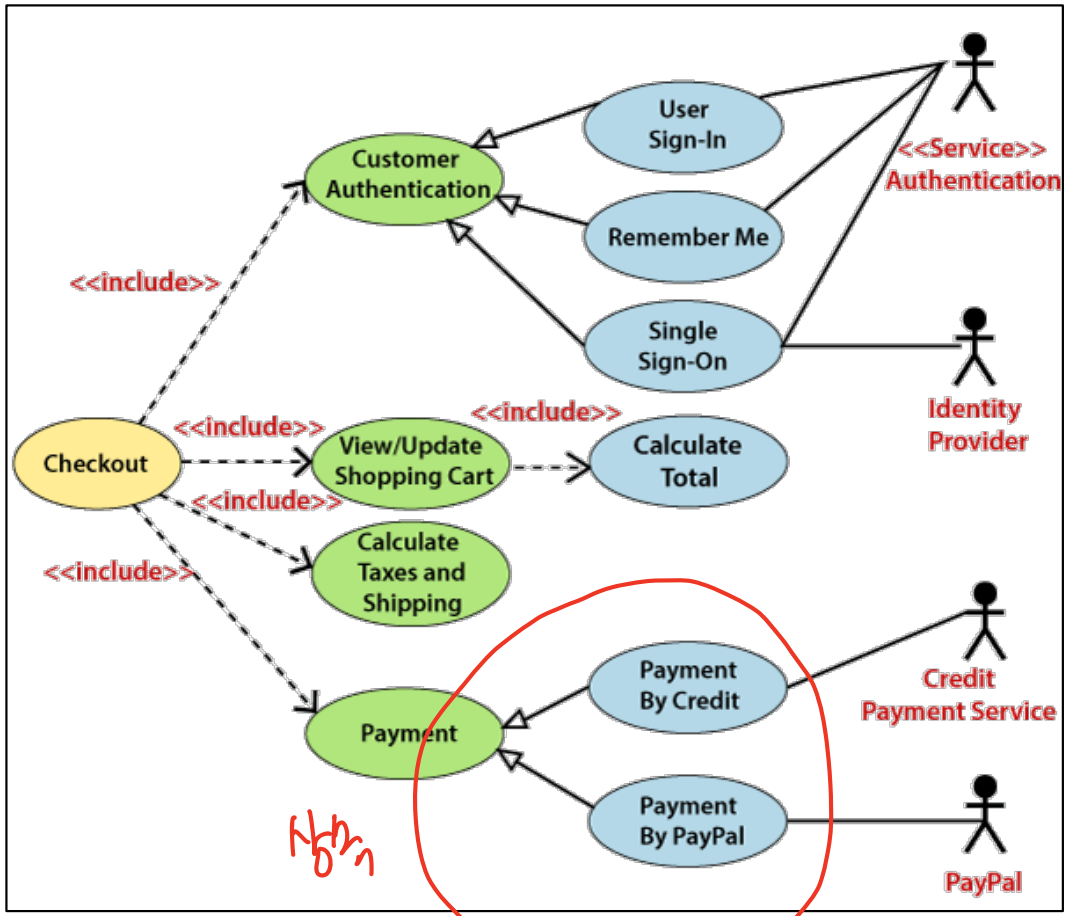
3.1.4 예제

일반화관계 (상속)





subsystem
→스테레오타입으로 표시



상속

인증API 가 있는 서비스개발
→스테레오타입으로 표시

3.2 유스케이스 작성

- 유스케이스에 대한 간단한 식별 정보와 유스케이스 내부에서 처리해야 하는 기능의 상세한 흐름을 나타냄

유스케이스 이름:
관련 액터: UC와 상호작용하는 액터 (primary actor, secondary actor)
유스케이스 개요 : UC 기능을 간결하고 명확하게 기술
시작조건 : 수행이 시작되기 위하여 필요한 조건이나 시스템의 상태
정상적 사건의 흐름:
1.
2.
3.
…
대안흐름 : 정상흐름 이외의 선택흐름과 예외흐름
A1.
A2.
종료조건 : 수행이 완료된 후에 만족되어야 하는 조건
확장지점 : 확장관계 유스케이스가 호출되는 사건흐름 지점

유스케이스 이름: 책 주문
액터: 고객
개요: 선택한 책을 배달 정보를 입력하고 주문
시작 조건: 책이 선택되어 수량이 지정되어 있고 신용 불량 고객이 아니어야 함.
정상적 사건의 흐름:
1. 고객이 '주문' 메뉴를 눌러 유스케이스가 시작된다.
2. 고객이 이름과 주소를 입력한다.
3. 책의 ISBN 번호를 입력하면
3.1 시스템이 책의 자세한 정보와 가격을 화면에 제공한다.
3.2 시스템이 책의 가격을 총액에 누적한다.
3을 계속 반복
4. 고객이 확인을 누르면
5. 시스템이 정보를 검토하여 이상이 없으면 주문 정보를 저장하고 화면에 주문 확인 정보를 출력한다.
대안 흐름:
A1: 불량 데이터 (예외흐름)
1. 기본흐름 5 단계에 불량 데이터가 발견되면 시작한다.
2. 시스템이 고객에게 불량 데이터를 지적하고 정보를 다시 출력하여 고칠 수 있게 한다.
3. 기본흐름 5 단계를 계속한다.
A2: 취소 (선택흐름)
1. '책 주문' 유스케이스의 어느 단계에서든지 '취소' 버튼을 누르면 시작한다.
2. 시스템이 고객에게 취소 의사를 다시 확인한다.
3. '확인'을 누르면 유스케이스를 종료한다.
종료조건: 주문이 취소되지 않는다면 시스템에 저장하고 확인된 주문으로 표시한다.

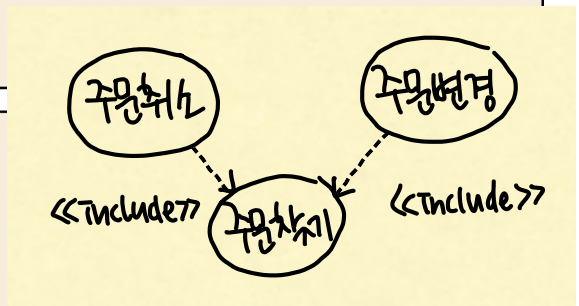
3.2.1 유스케이스 사건 흐름

- 기본 흐름
  - 유스케이스의 여러 시나리오 중에서 가장 일반적이고 정상적인 상황을 나타내는 하나의 이벤트 흐름
- 대안 흐름(Alternative Flow)
  - 한 유스케이스에서 기본 흐름이 표현하는 상황을 제외한 모든 다른 상황
  - 수행이 끝나면 기본 흐름으로 돌아오는 경우도 있고 대안 흐름에서 종료되는 경우도 있음
  - 선택 흐름과 예외흐름

3.2.2 유스케이스의 링크 (1/2)

- 포함 관계

유스케이스: 주문 찾기
사건의 흐름:
1. 시스템은 고객의 ID를 가지고 최근 3년 동안 주문한 모든 주문을 찾는다.
2. 찾은 모든 주문을 시간이 빠른 순서로 나열한다.
3. 고객이 특정 주문을 선택한다.



유스케이스: 주문 취소
사건의 흐름:
1. 고객이 '주문 취소' 메뉴를 선택한다.
2. <<포함: '주문 찾기'>>
3. 시스템은 고객이 선택한 주문의 상태가 '상품 준비 중'이라면,
   3.1 주문의 상태를 취소로 만든다.
   3.2 회계 시스템에 고객에게 환불 또는 카드 승인 취소를 요청하고 마친다.
4. 고객이 선택한 주문의 상태가 '배송 중'이라면
   4.1 고객에게 반품에 관한 정책을 알리고 종료한다.

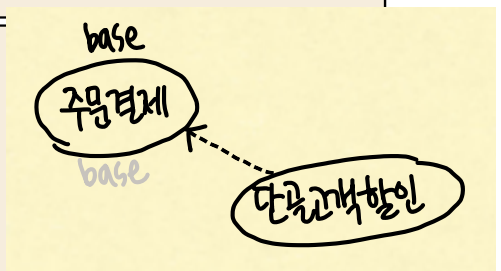
3.2.2 유스케이스의 링크 (2/2)

- 확장 관계

유스케이스: 책 주문
사건의 흐름:
1. 고객이 '책 주문' 메뉴를 선택한다.
2. 고객이 이름과 주소를 입력한다.
3. 책의 ISBN 번호를 입력하면
   a) 시스템이 책의 자세한 정보와 가격을 화면에 제공한다.
   b) 시스템이 책의 가격을 총액에 누적한다.
   3을 계속 반복
4. 고객이 확인을 누르면
5. 시스템이 정보를 검토하여 이상이 없으면 주문 정보를 저장하고 화면에 주문 확인 정보를 디스플레이 한다.

?!

유스케이스: 멤버십 할인
사건의 흐름:
<<확장: '책 주문'에서 주문할 책이 다 입력 된 후 확장>>
1. 고객이 단골 고객의 할인 프로그램인 멤버십이 있다면 시작한다.
2. 시스템은 화면에 멤버십 할인 해당자임을 디스플레이 한다.
3. 주문 총액에 할인율을 적용하여 할인 총액을 계산한다.
4. 주문 총액에서 할인 총액을 차감하고 디스플레이 한 후 종료한다.



3.3 예제 : 게시판 (1/7)

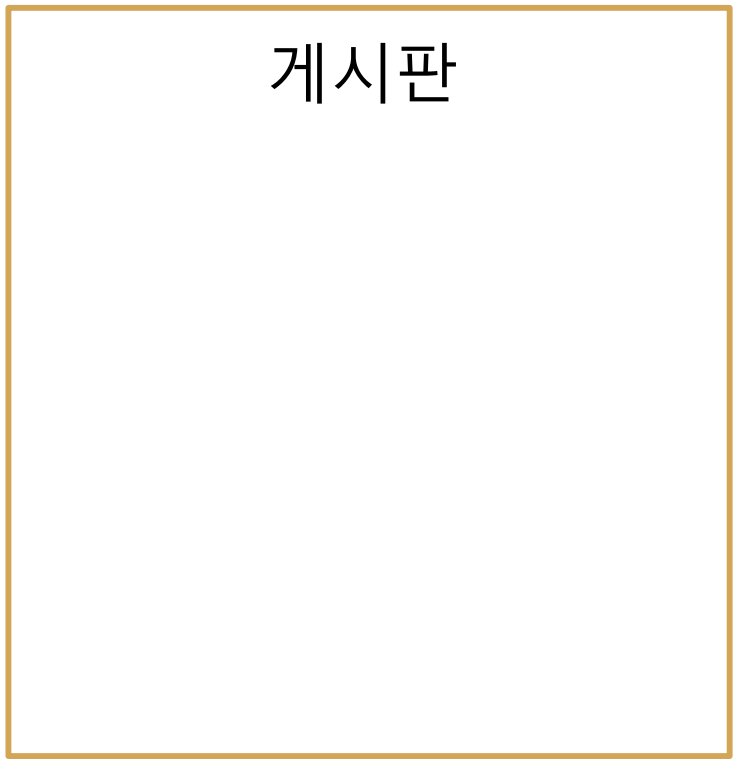
- 사용자 요구사항
  - 글을 등록, 수정, 삭제 할 수 있는 게시판을 개발한다
  - 단, 관리자 모드는 개발하지 않는다

    - 글을 등록할 때 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

3.3 예제 : 게시판 (2/7)

- ① 액터 식별 → 액터가 필요로하는 서비스가 유즈케이스네?
  - 개발 할 게시판의 외부에서 상호작용하는 액터로는 글을 등록하고 삭제하는 등의 역할을 하는 사용자

사용자



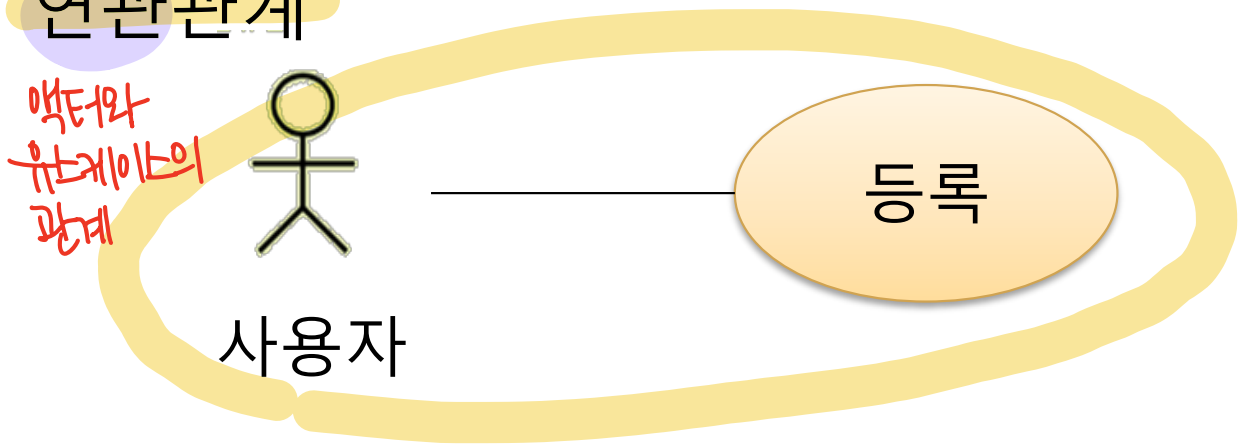
3.3 예제 : 게시판 (3/7)

- 유스케이스 식별 → (구현기능목록)
  - 사용자는 게시판을 통해 글을 등록, 수정, 조회하는 등의 작업을 함

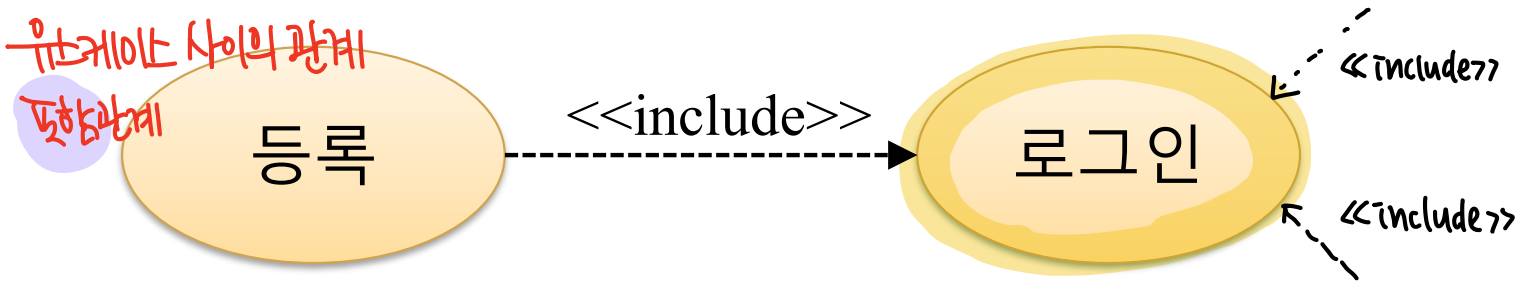
구현될만한것을모두찾아검토
요구사항에 드러나지 않는것찾기

3.3 예제 : 게시판 (4/7)

- 관계 정의
  - 연관관계

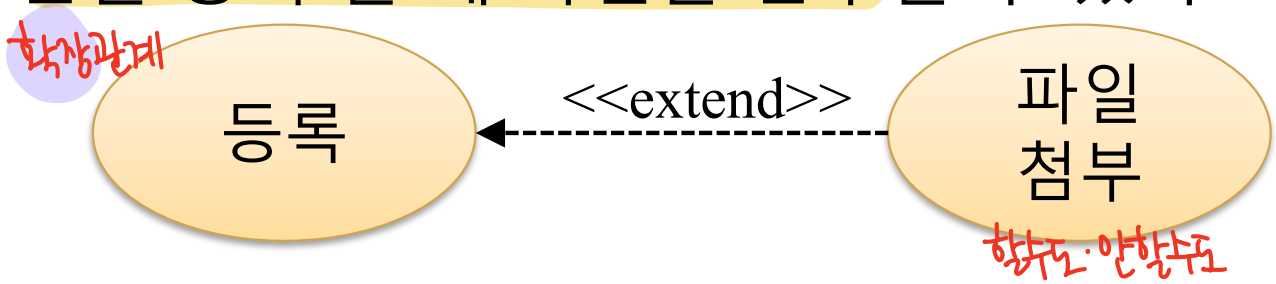


  - 로그인을 해야 모든 기능을 사용할 수 있다

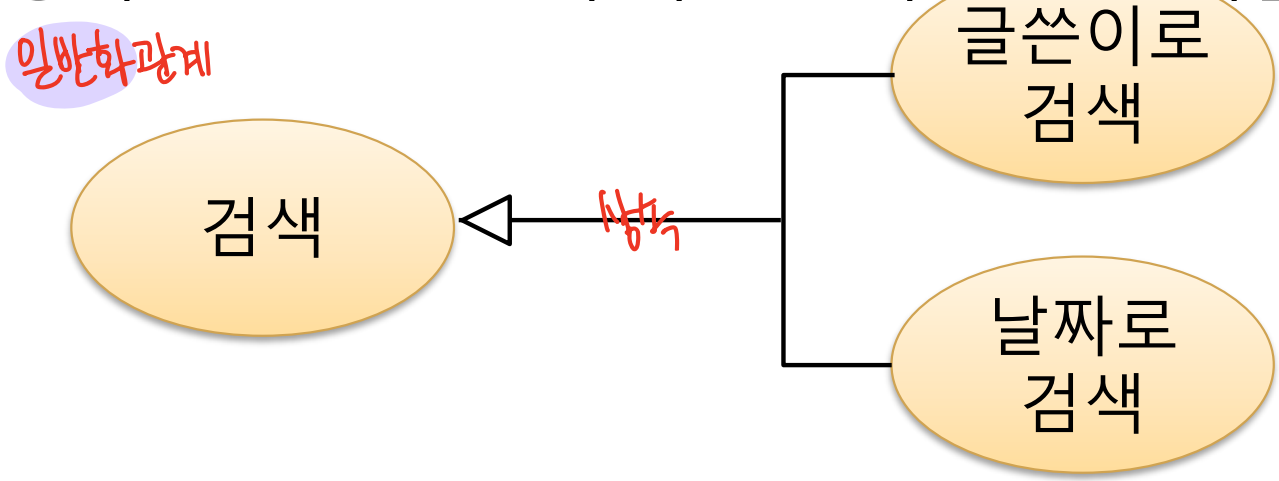


3.3 예제 : 게시판 (5/7)

- 관계 정의
  - 글을 등록 할 때 파일을 첨부할 수 있다

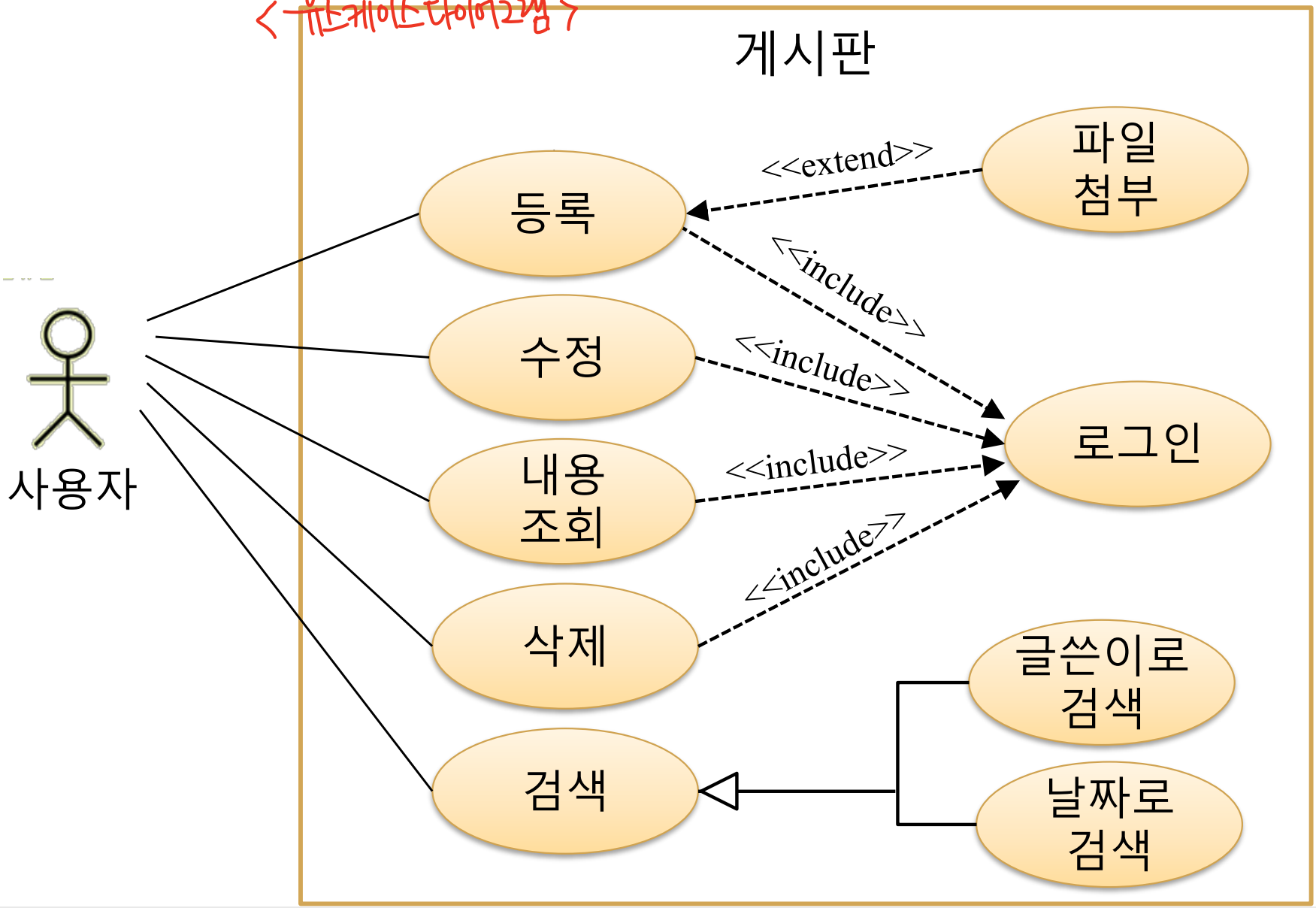


  - 등록된 글은 글쓴이 혹은 날짜 별로 검색할 수 있다



3.3 예제 : 게시판 (6/7)

<유스케이스 다이어그램>



3.3 예제 : 게시판 (7/7)

<유스케이스명세서>

유스케이스 이름: 등록
액터: 사용자
목표: 사용자가 원하는 글을 게시판에 등록한다
시작 조건: 사용자는 로그인 되어있다.
정상적 사건의 흐름:

1. <<포함: '로그인'>>
2. 사용자는 글쓰기 버튼을 클릭한다.
3. 시스템이 글쓰기 창을 보여준다.
4. 사용자는 글쓰기 박스에 원하는 글을 작성한다.
5. 사용자가 등록 버튼을 선택하면, 시스템은 작성된 글과 정보를 저장한다.

대안 흐름:
A1: 등록취소

4a. 어느 단계 에서든지 '취소' 버튼을 누르면, 게시판 목록조회 화면을 표시한다.
4b. 정상흐름 4에서 글쓰기 박스에 글을 쓰지 않고 등록을 원하는 경우, '내용을 넣으세요'라는 메시지를 표시한다.

종료 조건: 사용자가 작성한 글이 시스템에 저장되고, 등록 완료 화면을 표시한다.

~ 중간고사 시험범위 ~