2023학년도 2학기

# 소프트웨어 공학

## 7. 요구사항 분석 (유스케이스 다이어그램)

SW융합대학 컴퓨터공학과
남 재 현 (jaehyun.nam@dankook.ac.kr)

DKU 단국대학교 DANKOOK UNIVERSITY
BoanLab Networked Systems and Security

# 목 차

# UML의 이해

- Unified Modeling Language (UML)
  - 시스템 개발을 위한 시각적인 설계 표기 제공
  - 객체 지향 시스템을 개발할 때 산출물을 명세화, 시각화, 문서화하는 데 사용
  - 개발하는 시스템을 이해하기 쉬운 형태로 표현
  - 분석가, 설계자, 의뢰인 등이 효율적으로 의사소통 할 수 있게 해 줌

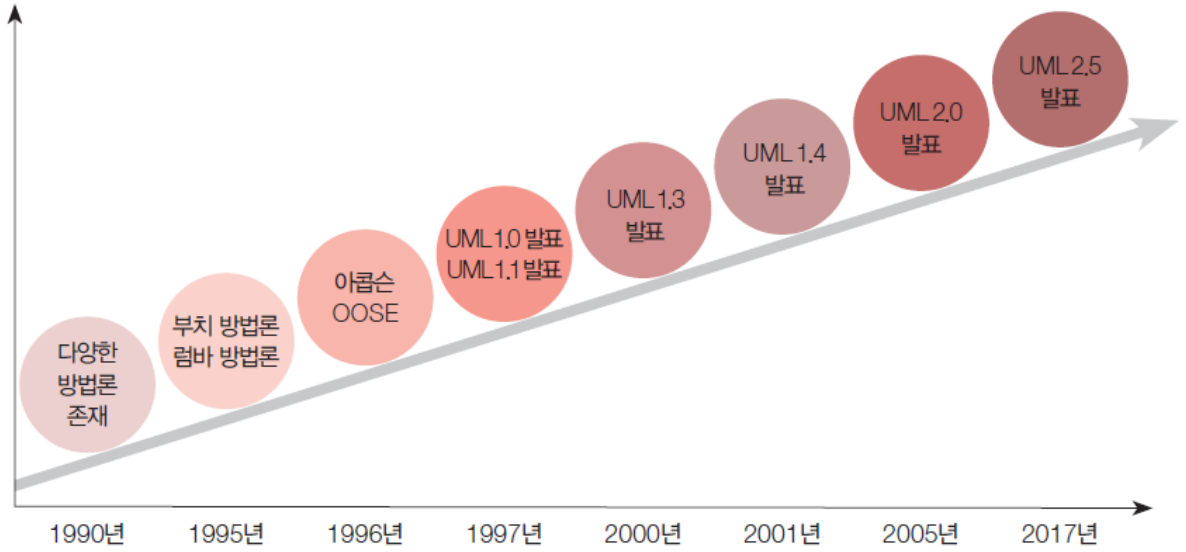
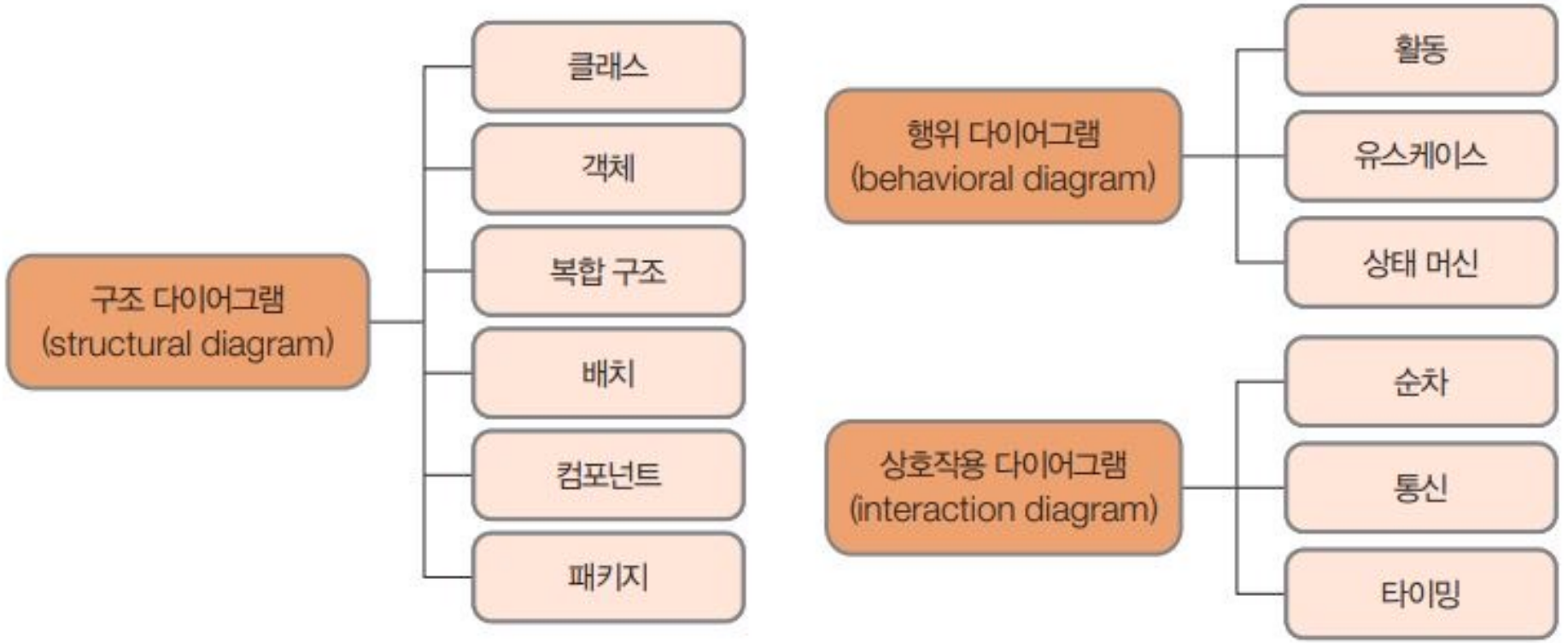



그림 1-2 UML의 발전 과정

# UML의 이해

- Unified Modeling Language (UML)
  - 소프트웨어의 전체를 판단할 수 있도록 12개의 다이어그램을 제시
  - 시스템 내 상호작용, 시스템 전체 구조, 컴포넌트 간 관계 등 시각화



# UML의 이해

- UML의 특징
  - 시각화Visualization 언어
    - 소프트웨어의 개념 모델을 시각적인 형태로 표현하며 명확히 정의된 표준화된 다이어그램을 제공
    - 이를 이용해 오류 없는 원활한 의사소통 가능
  - 명세화Specification 언어
    - 소프트웨어 개발 과정인 분석, 설계 단계의 각 과정에서 필요한 모델을 정확하고 완전하게 명세화
    - 명세화에서 각 다이어그램의 기호는 의미를 담고 있으며 추상적이지만 고유의 특성을 갖고 있음
  - 구축Construction 언어
    - 자바Java, C++, 비주얼 베이직Visual Basic, C# 같은 다양한 프로그래밍 언어로 표현가능
    - UML로 설계된 모델을 프로그램 코드로 자동 변환할 수 있으며,
      이미 구축된 소스 코드를 UML로 역변환하여 분석하는 역공학Reverse Engineering도 가능
  - 문서화Documentation 언어
    - StarUML, 투게더Together 등 케이스 툴CASE Tool을 이용하여 설계한 내용을 자동으로 문서화

# UML의 이해

- UML과 모델링

```c
#include <stdio.h>
int main() {
      int a, b;
      scanf("%d %d", &a, &b);
      printf("a + b = %d", a + b);
 }
```

(a) C로 구현한 덧셈 프로그램

```python
def add():
      num = int(input("num1:"))
      num2 = int(input("num2:"))
      result = num + num2
      print("두수의 합은 " result)
```

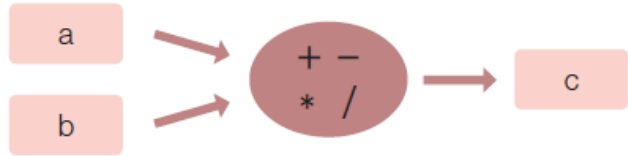
(c) 파이썬으로 구현한 덧셈 프로그램

그림 1-3 다양한 언어로 구현한 덧셈 프로그램

```java
import java.util.Scanner;
public class AddDemo {
      public static void main(String[] args) {
            Scanner in = new Scanner(System.in );
            int a = in.nextInt();
            int b = in.nextInt();
            System.out .printf("%d + %d = %d", a, b, a+b);
   }
}
```

(b) 자바로 구현한 덧셈 프로그램


(a) 덧셈 연산 프로그램의 시각적 표현




(b) 사칙연산 프로그램의 시각적 표현

그림 1-4 프로그램의 시각적 표현

개발하고자 하는 프로그램을 시각적으로 표현하는 것이며,
이때 의뢰자의 요구에 맞게 쉽게 수정해서
결과적으로 유지보수 기간을 줄여 생산성을 높일 수 있음

# UML의 이해

- UML과 모델링


(a) 개인이 제어할 수 있는 작업 : 개집 짓기


(b) 개인이 제어할 수 없는 작업 : 큰 건축물 짓기

그림 1-5 모델링이 필요한 이유

# 요구사항의 표현 – 객체지향 방법

- 유스케이스 다이어그램 (Use Case Diagram)
  - 시스템 기능에 대한 사용자 입장을 표현한 다이어그램
  - 시스템 분석가와 사용자가 시스템의 사용 방법과 목적을 결정하는데 도움을 주는 도구

- 유스케이스 다이어그램의 필요성
  - 요구 사항 정의는 개발과 설계에서 매우 큰 비중을 차지함
    - 누가who 시스템을 사용할 것인가?
    - 시스템은 사용자를 위해 무엇what을 해야 하는가?
    - 사용자와 상호작용하기 위해 시스템이 제공해야 할 인터페이스interface는 무엇인가?

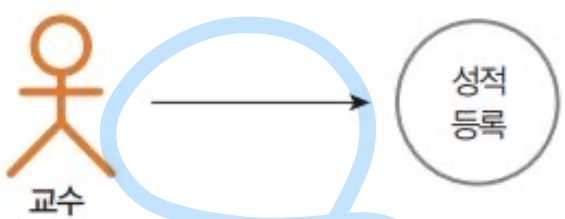
# 유스케이스 다이어그램

- 유스케이스 (Use Case)
  - 사용자가 시스템을 통해 사용하고 싶은 기능
  - 유스케이스가 모여 하나의 서브시스템을 이루고 서브시스템이 모여 개발 시스템이 됨
  - 즉, 전체 시스템은 유스케이스를 모아 놓은 것과 같아야 함

# 유스케이스 다이어그램

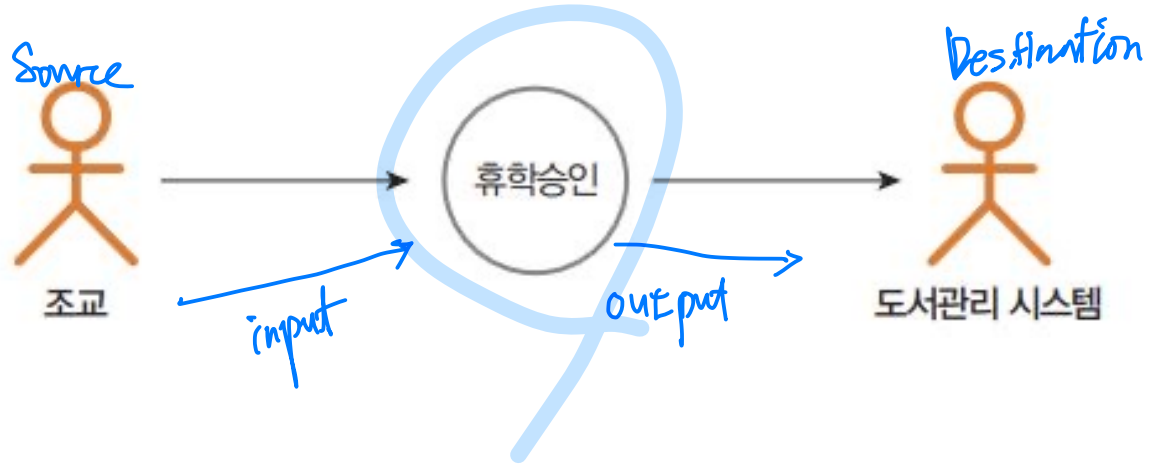
- 액터 (Actor) 종류
  - 사용자 (User)
    - 시스템을 사용하는 사람(역할)을 의미


  - 액터와 유스케이스의 관계는 화살표(→)를 사용해 표현
    - 화살표 방향은 액터에서 유스케이스로 향함



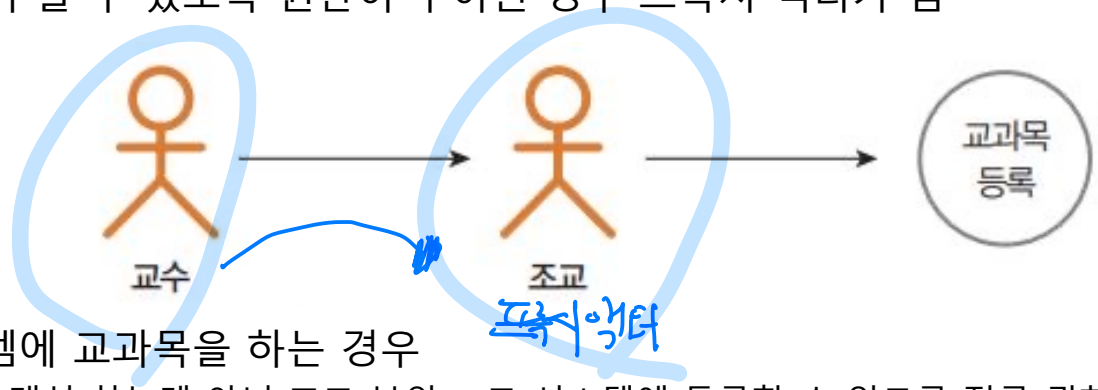
# 유스케이스 다이어그램

- 액터 종류
  - 시스템 (System)
    - 해당 프로젝트의 개발 범위에 속하지 않지만, 데이터를 주고받는 등 서로 연동되는 또 다른 시스템
  - 유스케이스가 시스템 액터를 사용(참조)
    - 화살표 방향은 유스케이스에서 시스템 액터로 향함



# 유스케이스 다이어그램

- 액터 종류
  - 주요 액터 (Primary Actor)
    - 시스템에게 작업의 실행을 요구하는 능동적 입장의 액터 (대부분 여기에 해당)
  - 보조 액터 (Secondary Actor)
    - 유스케이스로부터 요청을 받거나 메시지를 전달받아 수동적으로 작업하는 액터
    - 예시) 학사관리시스템에서 보조 액터는 시스템을 유지관리하는 유지보수 담당자
  - 프록시 액터 (Proxy Actor)
    - 액터와 시스템의 중간 위치에서 무엇인가 대신해 주는 액터
    - 시스템에 어떠한 행위 할 수 있도록 권한이 부여된 경우 프록시 액터가 됨




    - 예시) 학사관리시스템에 교과목을 하는 경우
      - 교수 ID로 교수를 대신 하는게 아닌 조교 본인 ID로 시스템에 등록할 수 있도록 접근 권한을 부여한 경우 조교는 프록시 액터가 됨

# 유스케이스 다이어그램

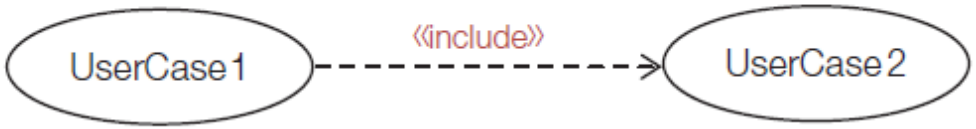
- 액터 식별 → 누가 쓰지?
  - 액터 식별 방법 예시 - 학사관리시스템
    - 대상) 시스템을 사용하는 사람을 찾음
      - 학사관리시스템을 사용하는 사람은 교수, 학생, 조교, 학사담당직원 등
    - 행위의 대상) 시스템에 자료를 등록/수정/삭제/조회하는 사람을 찾음 (구체적)
      - 학사관리시스템에서 액터는 성적을 등록/수정/삭제/조회할 수 있는 교수, 성적을 조회만 할 수 있는 학생, 시스템을 관리하는 학사담당직원
    - 연계 시스템) 시스템에 연동된 또 다른 시스템을 찾음 (시스템 액터) → 카카오 로그인, 구글 맵 등
      - 휴학신청 유스케이스와 연동되는 도서관리시스템
    - 그 외) 시스템을 유지 보수하는 사람을 찾음 (보조 액터) → 운용하는 사람도 생각.
      - 시스템의 기능을 직접 사용하지는 않지만 시스템이 잘 구동되도록 유지 및 관리해 주는 유지보수담당자
    - 그 외) 권한은 없지만 역할을 대신하는 사람을 찾음 (프록시 액터)

# 유스케이스 다이어그램

- 액터 이름 설정 규칙
  - 이름만 봐도 액터의 의미를 알 수 있게 정함
  - 역할을 나타내는 단어로 사용자 액터의 이름을 정함
    - 부장, 과장 같은 직책이 아닌 역할로 정함
    - 구체적인 역할을 나타내는 단어를 사용
  - 시스템 액터는 시스템 이름을 사용해 이름을 정함

# 유스케이스 다이어그램

- 엑터와 유스케이스
  - 유스케이스 사이의 포함Include 관계
    - 다른 유스케이스에서 기존 유스케이스를 재사용할 수 있음을 나타냄
  - 유스케이스 사이의 확장Extend 관계
    - 기존 유스케이스에 진행 단계를 추가하여 새로운 유스케이스를 만들어내는 관계




(a) 포함 관계


(b) 확장 관계

그림 3-3 유스케이스 다이어그램에서 사용되는 관계들

# 유스케이스 다이어그램

- 엑터와 유스케이스 사이의 관계
  - 연관Association 관계
    - 해당 액터와 정보를 주고받는 유스케이스와 설정함
    - —로 표시


그림 3-4 액터가 외부 시스템인 경우

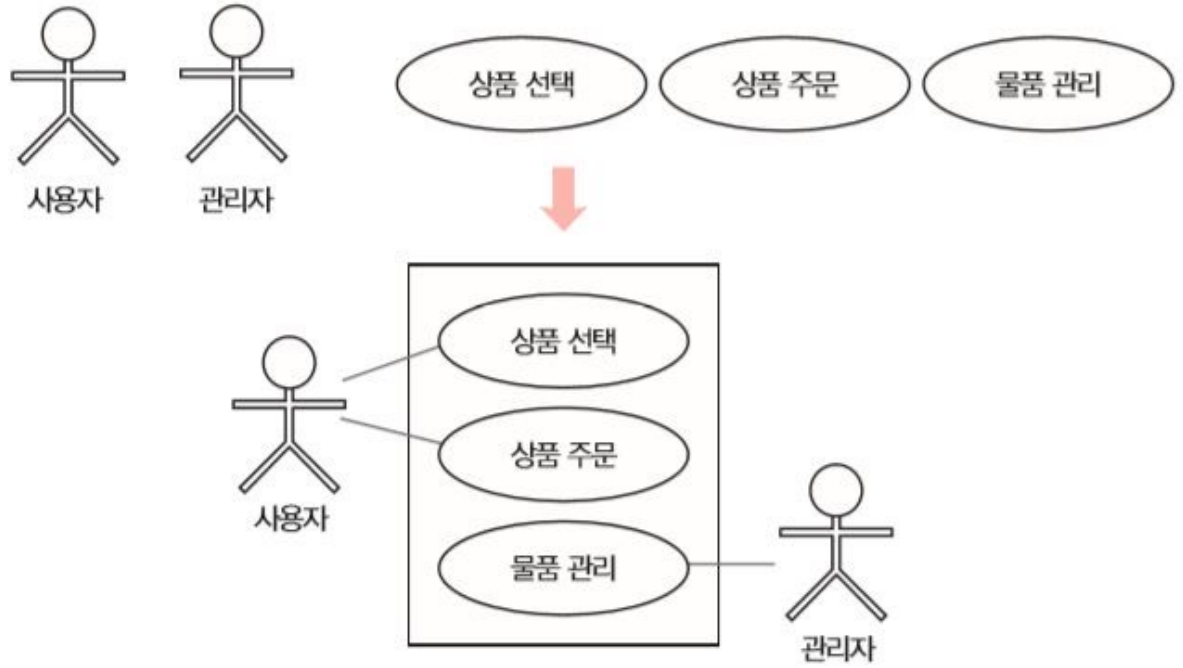



그림 3-5 액터와 유스케이스의 연관 관계

# 유스케이스 다이어그램

- 유스케이스 사이의 관계
  - 포함 관계
    - 하나의 유스케이스를 수행할 때, 같은 기능이 있는 다른 유스케이스가 반드시 수행되는 관계

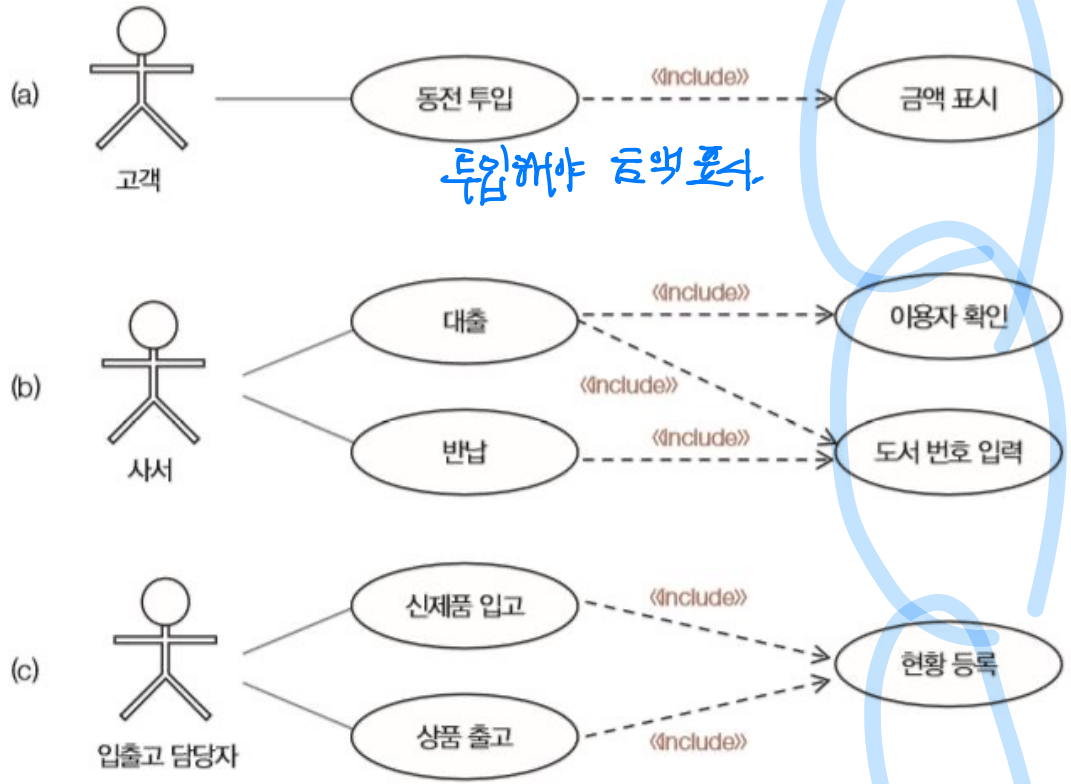



그림 3-6 유스케이스 간 포함 관계 예

(a)에서 고객이 자판기에 동전을 투입하면 금액이 자동으로 표시

(b)에서는 사서가 이용자 확인과 도서 번호 입력을 거쳐 대출하고, 반납 시 도서 번호만 입력

(c)에서는 입출고 담당자가 신제품 입고나 상품 출고를 하면 자동으로 현황 등록이 이루어짐

표 3-1 대출과 반납 유스케이스의 이벤트 흐름

| 대출 유스케이스 이벤트 흐름 | 반납 유스케이스 이벤트 흐름 |
|---|---|
| 1. 사서가 대출을 선택한다. | 1. 사서가 반납을 선택한다. |
| 2. 이용자 확인 유스케이스를 포함한다. | 2. 도서 번호 입력 유스케이스를 포함한다. |
| 3. 이용자가 대출이 가능한지 확인한다. | 3. 도서 번호 입력을 처리한다. |
| 4. 이용자에게 대출 가능 여부를 표시한다. | 4. 도서 반납을 처리한다. |
| 5. 도서 번호 입력 유스케이스를 포함한다. | |
| 6. 도서 관리 시스템이 도서 대출을 처리한다. | |

# 유스케이스 다이어그램

- 유스케이스 사이의 관계
  - 확장 관계
    - 확장하는 유스케이스는 상위 유스케이스로부터 어떠한 특정 조건에 의해 수행
    - 기본 유스케이스를 수정하지 않고 새로운 요구 사항을 추가로 표현하고자 할 때 사용

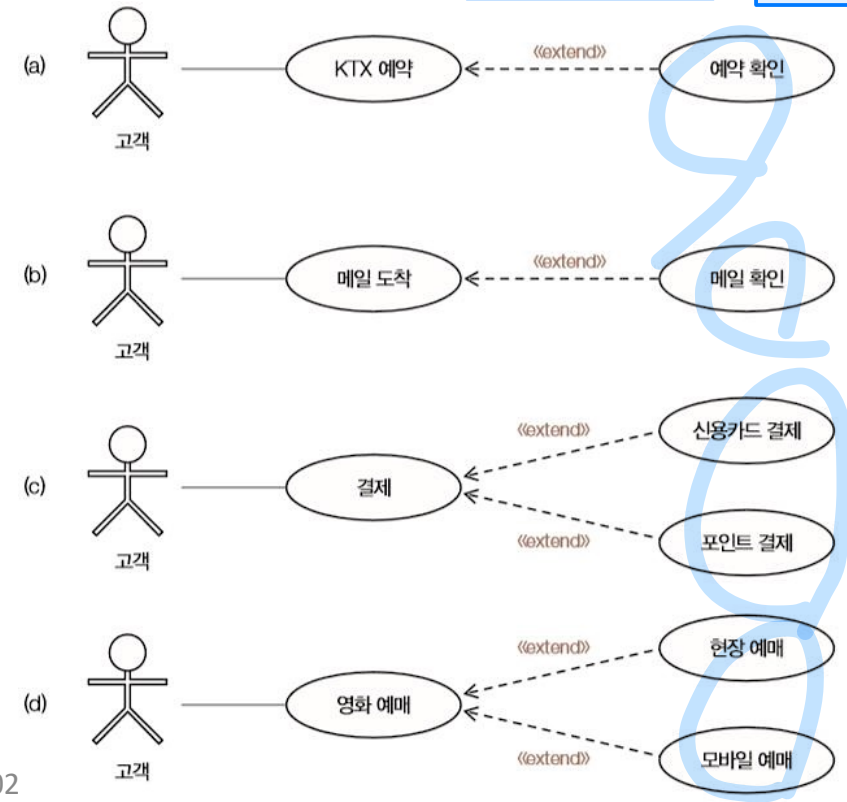



그림 3-7 확장 유스케이스

(a)는 KTX를 예약한 후 결과를 확인하거나 확인하지 않을 수 있는 예다

(b)는 메일이 도착했으나 확인은 선택이다

(c)는 결제할 때 신용카드 또는 포인트 로 결제하는 경우이다

(d)는 영화를 현장에서 예매하거나 모바일로 예매하는 경우이다


그림 3-8 도서 예약 유스케이스의 확장

표 3-2 도서 예약 유스케이스에 대한 이벤트 흐름

| 도서 예약 유스케이스 이벤트 흐름(확장) | 도서 예약 유스케이스 이벤트 흐름 |
|---|---|
| 1. 사용자가 도서 예약을 선택한다.<br>2. 예약 가능 확인을 확장한다.<br>3. 도서가 예약 가능한지 알려준다.<br>4. 도서 예약을 시행한다. | 1. 사용자가 도서 예약을 선택한다.<br>2. 시스템은 도서 예약을 시행한다. |

# 유스케이스 다이어그램

- 유스케이스 사이의 관계
  - 확장 관계와 포함 관계의 차이 (확장 관계)
    - 기준 유스케이스 이후의 이벤트 흐름은 확장 유스케이스의 수행 결과에 의존

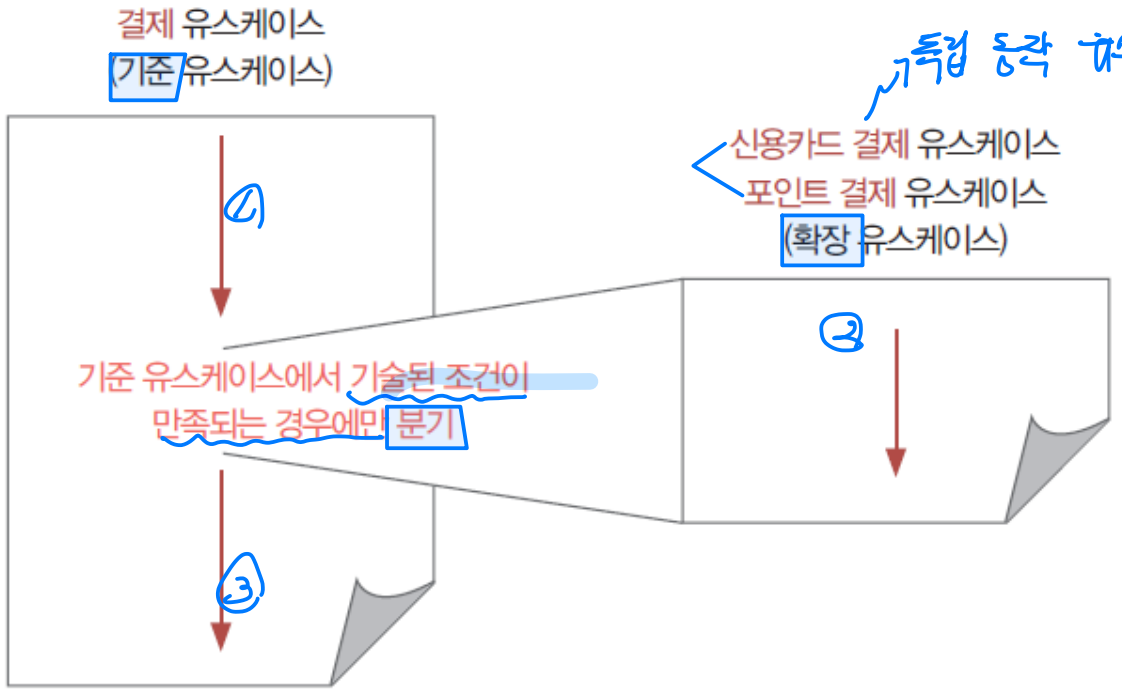



그림 3-9 확장 관계의 이벤트 흐름

1. 기준 유스케이스인 결제에 기술된 이벤트 흐름이 차례로 수행
2. 확장 부분에서 확장 유스케이스인 신용카드 결제나 포인트 결제로 분기
3. 확장 유스케이스에 기술된 이벤트 흐름의 수행이 완료
4. 다시 기준 유스케이스로 되돌아와서 이후의 이벤트 흐름을 수행

# 유스케이스 다이어그램

- 유스케이스 사이의 관계
  - 확장 관계와 포함 관계의 차이 (포함 관계)
    - 포함 유스케이스의 수행 결과에 따라서 기준 유스케이스의 이벤트 흐름이 영향을 받음

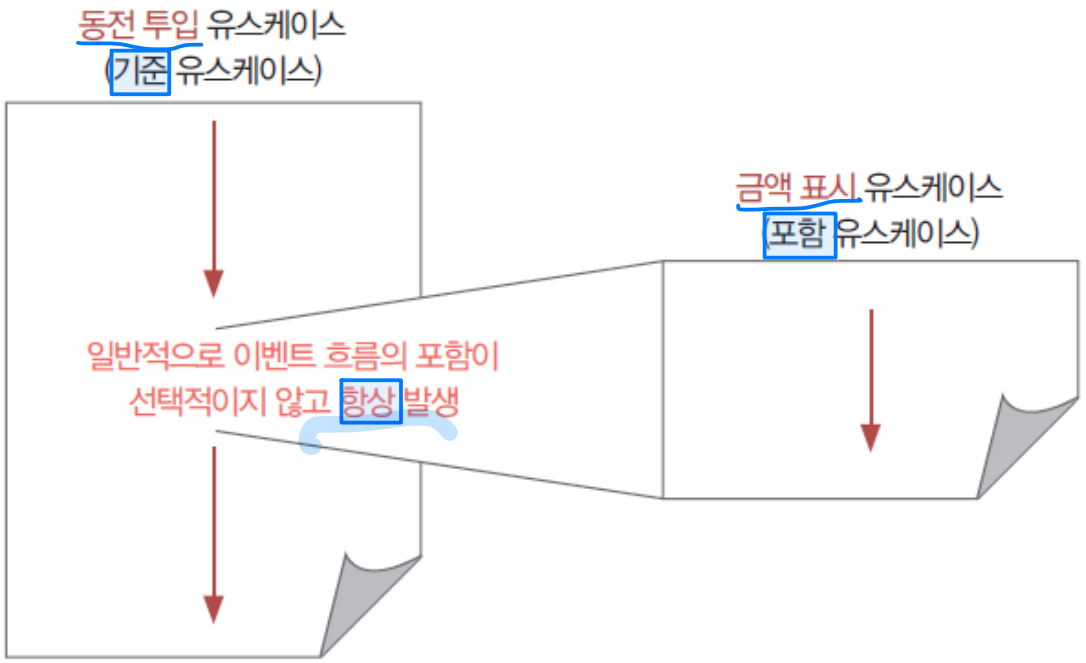



그림 3-10 포함 관계의 이벤트 흐름

1. 기준 유스케이스인 동전 투입에 기술된 이벤트 흐름이 차례로 수행
2. 특정 지점에서 포함된 유스케이스(금액 표시)로 바로 분기
3. 금액 표시 유스케이스의 이벤트 흐름이 모두 수행되면 다시 동전 투입
4. 유스케이스의 이벤트 흐름으로 돌아와 이후의 이벤트를 수행

# 유스케이스 다이어그램

- 유스케이스 사이의 관계
  - 확장 관계와 포함 관계의 차이

| | 포함 관계 | 확장 관계 |
|---|---|---|
| 목적 | • 여러 유스케이스에 공통적인 기능을 표현하기 위해 사용된다. | • 기준 유스케이스에 부가적으로 추가된 기능을 표현하기 위해 사용된다. |
| 이벤트 흐름 | • 포함 유스케이스로 분기되는 이벤트 흐름이 필수적이다.<br>• 기준 유스케이스 이후의 이벤트 흐름이 포함 유스케이스의 수행 결과에 의존한다. | • 기준 유스케이스에 기술된 조건에 따라 분기가 선택적으로 수행된다.<br>• 기준 유스케이스 이후의 이벤트 흐름이 확장 유스케이스의 결과에 의존하지 않는다. |

# 유스케이스 다이어그램

- 액터 사이의 관계
  - 일반화 관계
    - 액터들이 유스케이스와 중복하여 관계가 나타나면 액터들을 통합하여 일반화 관계로 표현
    - 추상적인 액터와 좀 더 구체적인 액터 사이에 관계를 맺어줌

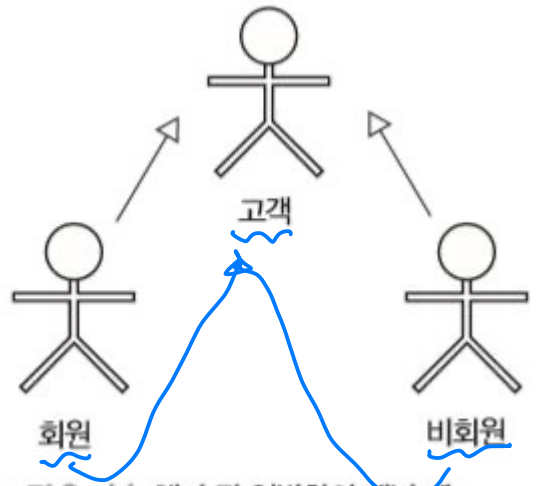



그림 3-11 액터 간 일반화의 개념 예

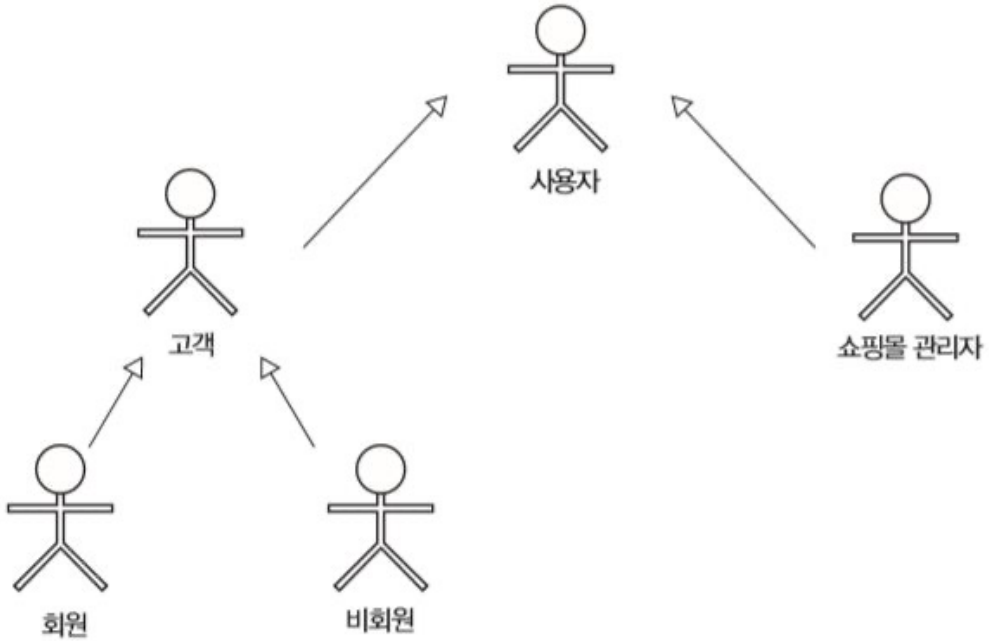
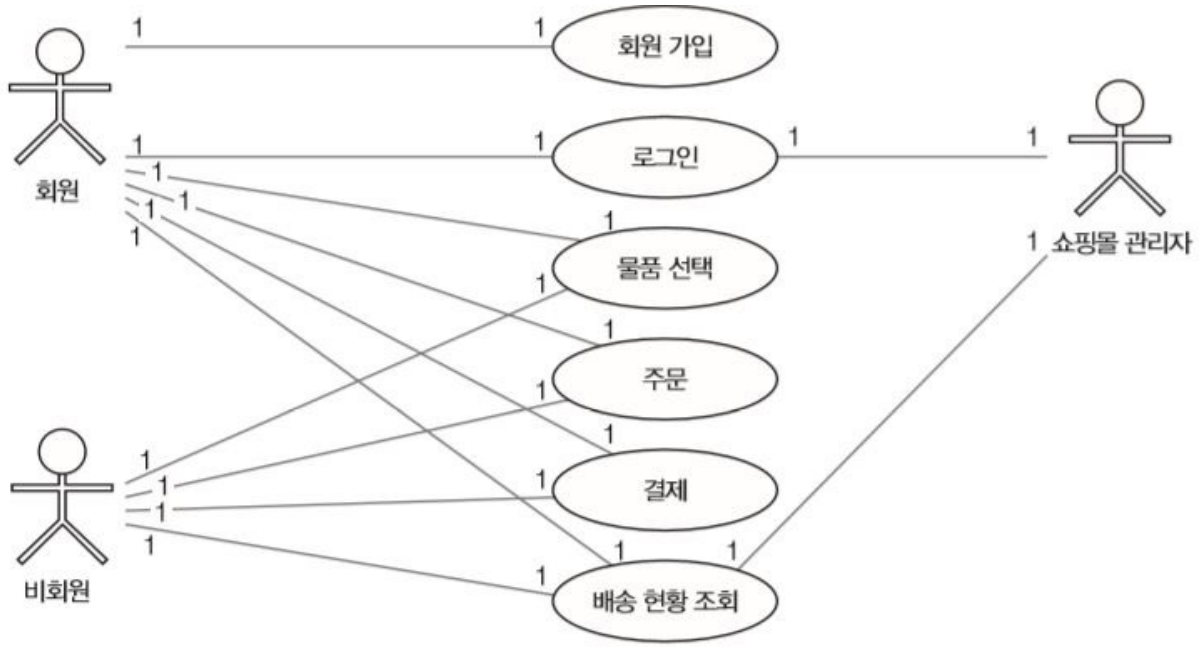


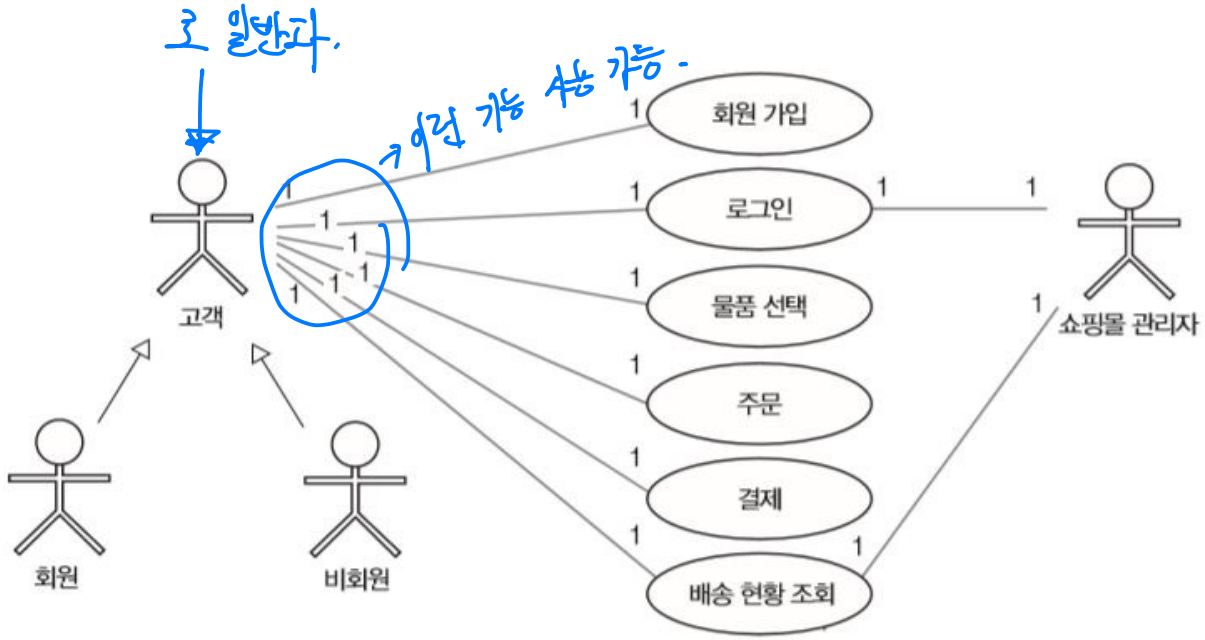

그림 3-12 일반화 관계 예

# 유스케이스 다이어그램

- 액터 사이의 관계
  - 일반화 관계




(a) 일반화 관계 적용 전




(b) 일반화 관계 적용 후

그림 3-13 일반화 관계 적용 전과 후 비교

# 유스케이스 다이어그램

- 액터 사이의 관계
  - 중복 관계
    - 유스케이스 모델링을 할 때 유스케이스 이벤트 흐름에서 중복된 부분이 있을 때 설정

표 3-4 대출 및 반납 유스케이스의 이벤트 흐름

| 대출 유스케이스 이벤트 흐름 | 반납 유스케이스 이벤트 흐름 |
|---|---|
| 1. 사서가 이용자의 아이디와 비밀번호를 도서 관리 시스템에 입력한다. | 1. 사서가 이용자의 아이디와 비밀번호를 도서 관리 시스템에 입력한다. |
| 2. 도서 관리 시스템이 아이디와 비밀번호로 고객 정보를 확인하고 원하는 작업을 선택할 수 있는 화면을 보여준다. | 2. 도서 관리 시스템이 아이디와 비밀번호로 고객 정보를 확인하고 원하는 작업을 선택할 수 있는 화면을 보여준다. |
| 3. 사서가 대출을 선택한다. | 3. 사서가 반납을 선택한다. |
| 4. 도서 관리 시스템이 사서에게 도서 번호를 입력하기 위한 화면을 보여준다. | 4. 도서 관리 시스템이 사서에게 도서 번호를 입력하기 위한 화면을 보여준다 |
| 5. 사서가 도서 번호를 입력한다. | 5. 사서가 도서 번호를 입력한다. |
| 6. 도서 관리 시스템은 도서 목록에서 도서 번호가 유효한지 확인한다. | 6. 도서 관리 시스템은 도서 목록에서 도서 번호가 유효한지 확인한다. |
| 7. 도서 관리 시스템은 이용자에게 대출이 가능한지 확인한다. | 7. 도서 관리 시스템이 도서 반납을 처리한다. |
| 8. 이용자에게 대출 가능 여부를 표시한다. | |
| 9. 도서 관리 시스템이 도서 대출을 처리한다. | |

표 3-5 고객 확인 유스케이스와 도서 번호 확인 유스케이스

| | |
|---|---|
| 고객 확인 | 1. 사서가 이용자의 아이디와 비밀번호를 도서 관리 시스템에 입력한다.<br>2. 도서 관리 시스템이 아이디와 비밀번호로 이용자 정보를 확인하고 원하는 작업을 선택할 수 있는 화면을 보여준다. |
| 도서 번호 확인 | 4. 도서 관리 시스템이 사서에게 도서 번호를 입력하기 위한 화면을 보여준다.<br>5. 사서가 도서 번호를 입력한다.<br>6. 도서 관리 시스템이 도서 목록에서 도서 번호가 유효한지 확인한다. |

포함 요구사항 만드는 이유? 기능, 절차 작성 → 절차는 뺄 하고.

# 유스케이스 다이어그램

- 액터 사이의 관계
  - 중복 관계

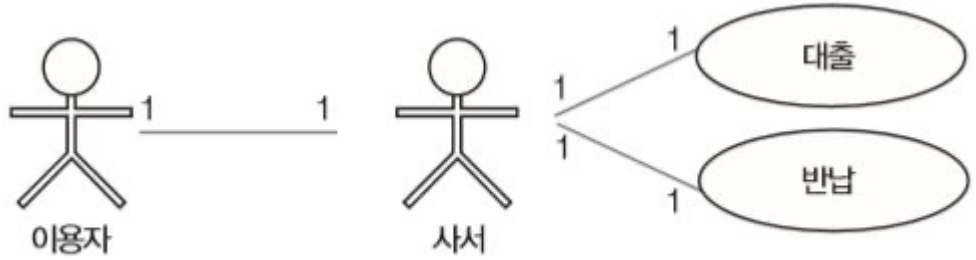



그림 3-14 도서 관리 시스템의 유스케이스 다이어그램

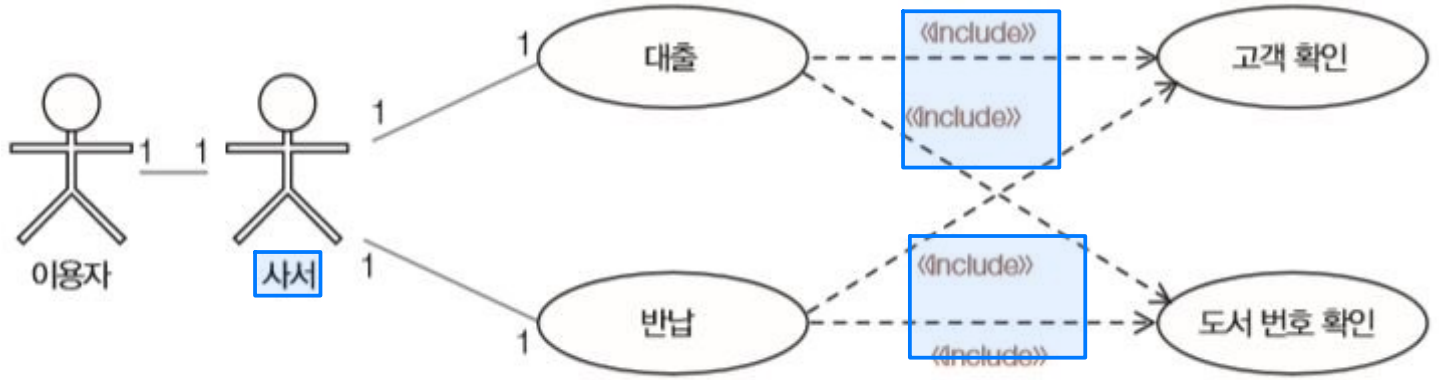



그림 3-15 중복이 제거된 도서 관리 시스템의 유스케이스 다이어그램

# 유스케이스 다이어그램의 단계별 모델링

- 유스케이스 모델링 단계
  - 1단계 : 시스템 상황 분석 → 인터뷰
    - 시스템 상황을 분석하여 문제 기술서를 작성
  - 2단계 : 액터 식별
    - 행위자와 그들의 책임을 확인 → 결국 action 주체.
    - 다음 질문으로 찾을 수 있음
      - 시스템의 주요 기능을 사용하는 사람이 누구인가?
      - 시스템을 지원하기 위해 필요한 사람은 누구인가?
      - 시스템을 유지하고 관리하는 사람은 누구인가?
      - 시스템에 필요한 하드웨어 장치는 무엇인가?
      - 시스템과 상호작용하는 다른 시스템은 무엇인가?
      - 시스템의 처리 결과에 연결되는 사람 또는 사물은 무엇인가?

      → 액터 설정.
  - 3단계 : 유스케이스 식별
    - 액터 관점에서 시스템의 기능을 확인 → 쭉 뽑기.

# 유스케이스 다이어그램의 단계별 모델링

- 유스케이스 모델링 단계
  - 4단계 : 유스케이스 다이어그램 작성
    - 액터와 유스케이스 관계를 설정
    - 유스케이스에서 <<include>> 의존성이 있는지 평가
    - 유스케이스에서 <<extend>> 의존성이 있는지 평가
    - 액터의 일반화 관계를 찾음
  - 5단계 : 유스케이스 명세서 작성
    - 유스케이스명, 액터명 및 개요를 기술
    - 사전 및 사후 조건과 제약사항들을 식별 → 게임했을 때 → 로그인해야 함.(로그인한 상태)
    - 작업(정상, 대치, 예외) 흐름과 시나리오를 도출 ~> 순차 다이어그램
    - 유스케이스 흐름에서 포함이나 확장 유스케이스로 구조화
  - 6단계 : 유스케이스 실체화
    - 구현 시스템의 논리적 구성 요소인 클래스를 식별하고 통신 관계를 파악하는 데 중점

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 1단계: 시스템 상황 분석
    - 도서 관리 시스템
      - 고객은 반드시 회원으로 가입해야 도서를 대여가능 / 대여, 반납, 연체 관리 기능이 있음
    - 관리자
      - 이름과 전화번호로 회원을 확인
      - 연체 관리 기능을 통해 현재 연체 중인 회원과 연체된 도서를 확인
      - 연체금 표시 기능을 사용해 오늘 날짜에 해당하는 연체금을 표시
      - 반납 기능을 통해 반납한 도서 코드를 입력하여 대여 목록에서 삭제
      - 새로운 도서의 등록 및 삭제를 관리할 수 있음
      - 대여할 때는 고객이 도서를 선택하면 도서 코드를 확인하여 시스템에 입력
    - 대여
      - 해당 고객이 현재 대여 중인 도서가 있으면 표시하고, 대여 기간이 지났으면 연체료를 계산하여 보여줌
      - 연체 고객은 연체료를 납부하면 도서를 대여할 수 있음
      - 대여료와 연체료는 현금이나 신용카드 결제를 통해 이루어짐
      - 대여된 도서는 대여 목록에 도서 코드와 고객명으로 등록

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 2단계: 액터 식별
    - 액터의 명칭으로 특정 사람의 이름보다는 역할을 의미하는 이름을 사용
    - 도서 관리 시스템에서는 고객, 관리자, 카드 승인 시스템 등의 액터 추출 가능
      - 이때 카드 승인 시스템은 외부 시스템으로 정의

외부 API.


그림 3-16 액터 식별

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 3단계: 유스케이스 식별
    - 유스케이스는 시스템을 수행하는 일련의 행위
    - 따라서 행위 자체만 표현할 뿐이며, 이때 행위 과정은 기술할 필요성 없음 → 뒤에서 얘기할 것.
    - 시스템에 원하는 기능들이 무엇인지를 찾아서 추출


그림 3-17 유스케이스 식별

# 유스케이스 다이어그램의 단계별 모델링

- 유스케이스 식별
  - 사용자인 액터가 시스템에 요구하는 기능을 후보 유스케이스로 선정
  - 사용자의 업무에서 유스케이스를 도출할 수 있음
  - 데이터베이스에서 데이터를 등록/수정/삭제/조회하는 기능은 하나의 유스케이스로 선정할 수 있음

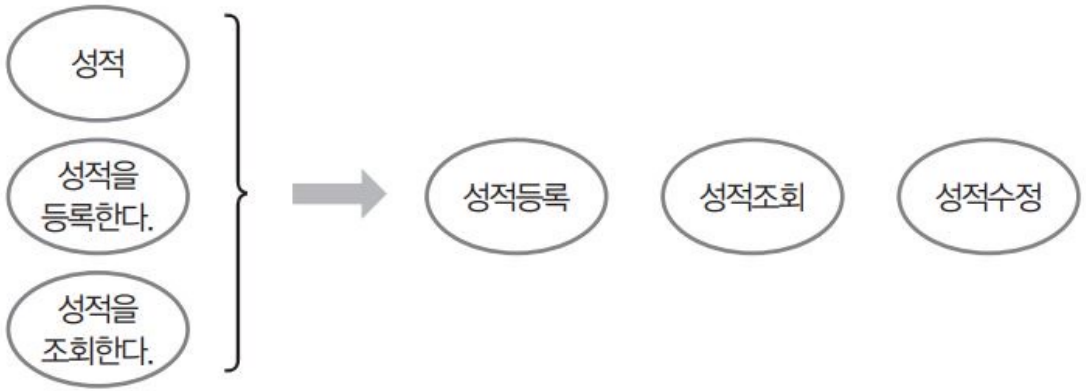
  - 처음에는 가능하다고 생각되는 것을 모두 도출해 봄
  - 하나씩 살펴보는 과정을 거쳐 불필요한 것은 삭제함
  - 필요시 합치거나 분리 해 최종 유스케이스를 선정함

  - 일반적으로 유스케이스를 찾는 작업은 개발자가 하지만 유스케이스 정의는 사용자 관점 수행해야 함

# 유스케이스 다이어그램의 단계별 모델링

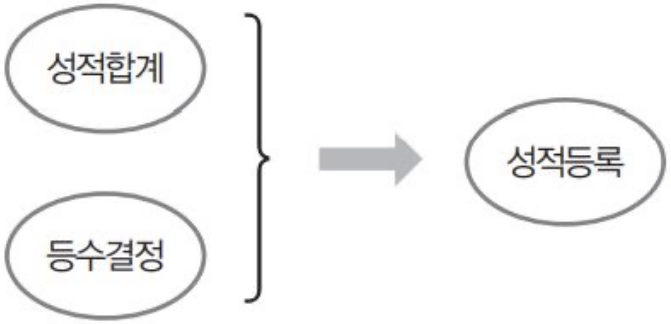
- 유스케이스 이름
  - 사용자 관점에서 이해할 수 있는 이름을 사용


  - 명사보다는 어떤 기능을 하는지 알 수 있는 동사형 명사를 사용



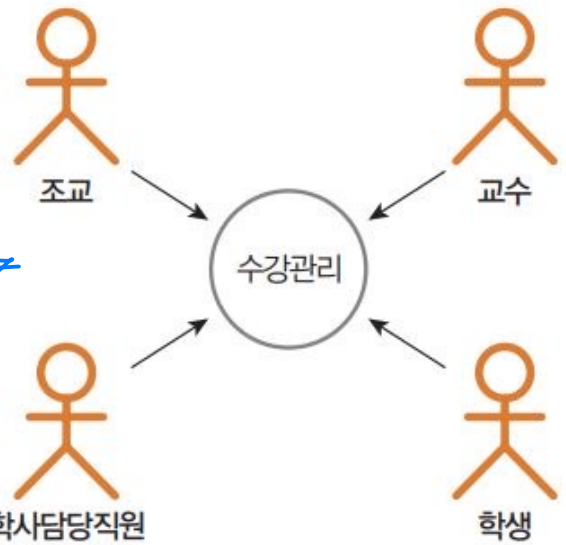

  - 사용자가 시스템을 통해 얻으려는 최종 목적이 나타나는 이름을 사용

과정이아닌

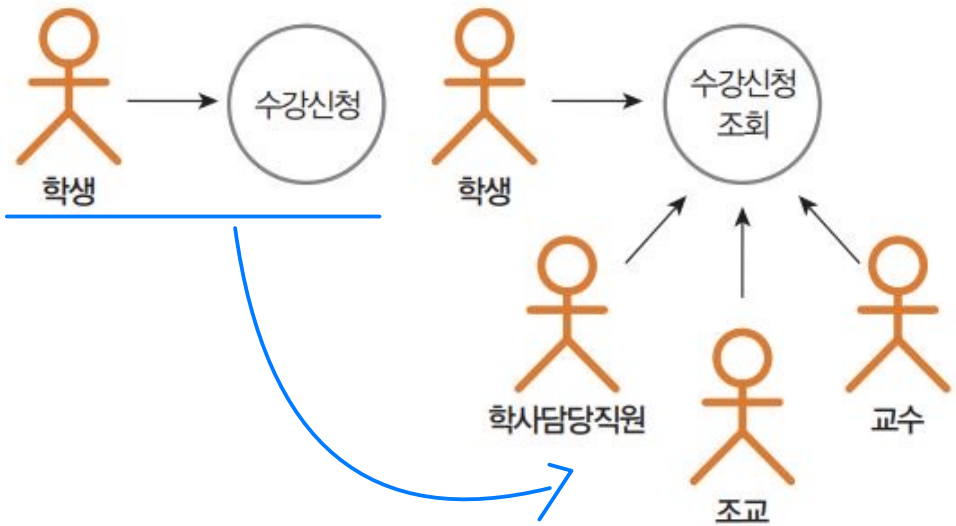


# 유스케이스 다이어그램의 단계별 모델링

- 유스케이스 검증
  - 시스템이 아니라 수작업으로 이루어지는 것은 유스케이스에 포함하면 안 됨
  - 액터가 원하는 최종 결과만 나타내는 유스케이스인지 확인
  - 액터가 수행하는 유스케이스인지 확인
  - 유스케이스 내의 이벤트 흐름 (event flow) 전체를 액터가 사용하는지 확인




(a) 부적절한 유스케이스




(b) 적절한 유스케이스

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 4단계 : 유스케이스 다이어그램 작성
    - 액터와 유스케이스, 유스케이스와 유스케이스의 관계를 설정해 표현
      - 고객 : 대여,반납,결제
      - 관리자 : 연체관리,등록,삭제

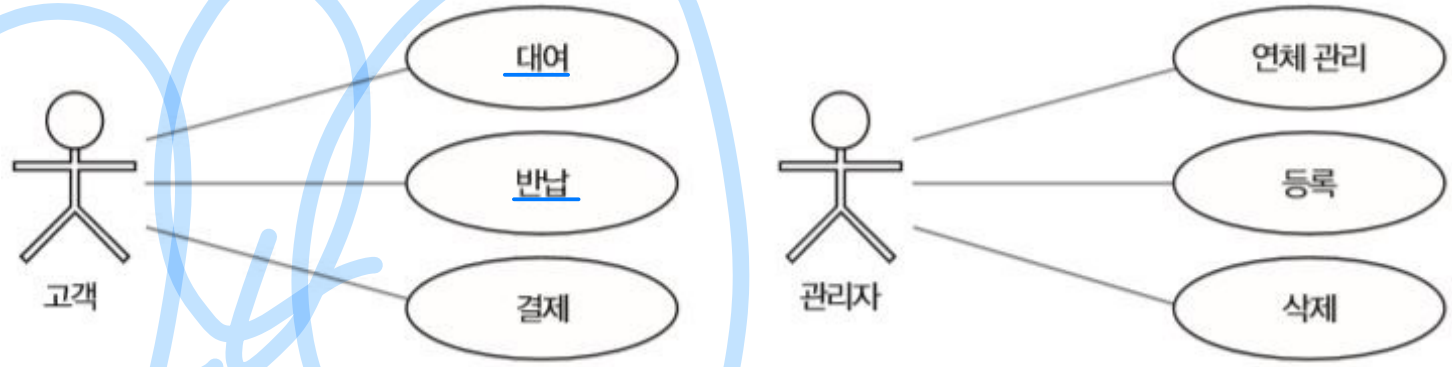



그림 3-18 액터와 유스케이스 사이의 관계 설정

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 4단계 : 유스케이스 다이어그램 작성
    - <<include>>
      - 하나의 유스케이스를 수행 할 때 같은 기능을 가진 또 하나의 유스케이스를 반복적으로 반드시 수행
      - 대여와 반납 유스케이스가 수행될 때는 반드시 도서 번호 입력 유스케이스가 선행(포함)필요
      - 대여할 때는 회원 확인 유스케이스도 포함되어야 한다

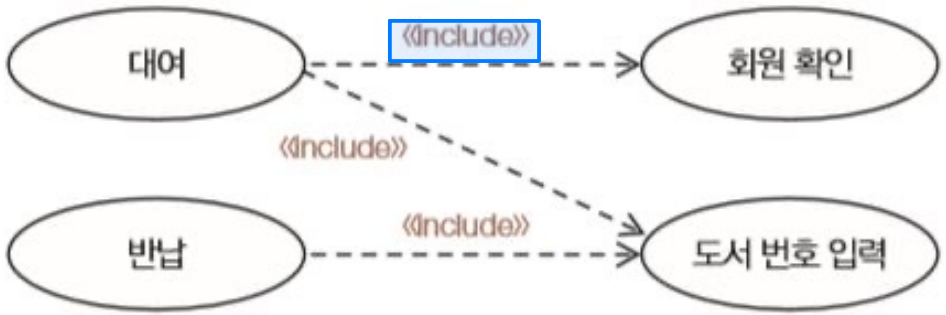



그림 3-19 include 관계

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 4단계 : 유스케이스 다이어그램 작성
    - <<extend>>
      - 여러 유스케이스에 걸쳐 중복 사용되지 않고, 특정 조건에서 한 유스케이스로만 확장되는 것을 의미
      - 상위 유스케이스로부터 어떠한 특정 조건에 의해 수행됨
      - 결제를 수행할 때는 결제 유스케이스로부터 신용카드 지불 유스케이스가 확장되는 형태로 이루어져야 함
      - 신용카드 지불 시 카드 승인사에 카드 승인을 요청해야 하므로, 카드 승인 시스템 액터와도 관계를 설정

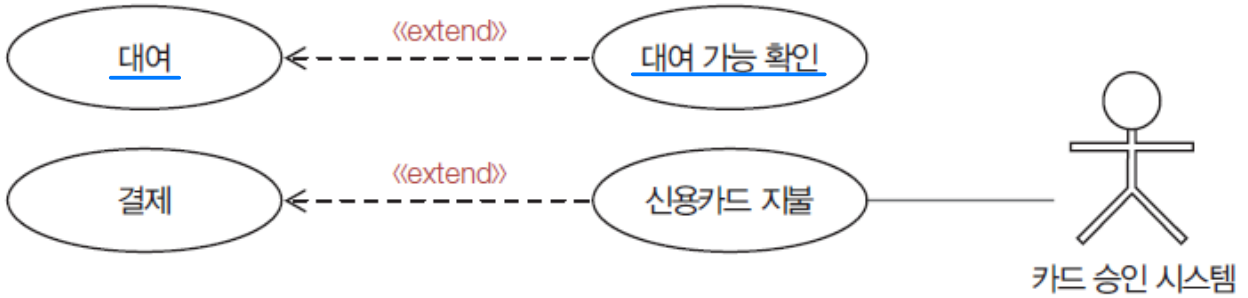



그림 3-20 extend 관계

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 4단계 : 유스케이스 다이어그램 작성
    - 액터와 유스케이스의 관계 및 유스케이스 사이의 관계
      - 고객은 회원 가입을 통해 시스템에 접근할 수 있음
      - 대여와 반납을 수행할 때는 반드시 도서 번호를 입력해야 함
      - 결제 시 신용카드를 사용하면 신용카드 지불 유스케이스로 확장할 수 있으며 신용카드 지불이 요청되면 카드 승인 시스템을 통해 카드 승인을 수행
      - 고객은 회원 가입, 대여, 반납, 결제 업무를 수행
      - 관리자는 등록, 삭제, 연체 관리 업무를 수행

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 4단계 : 유스케이스 다이어그램 작성

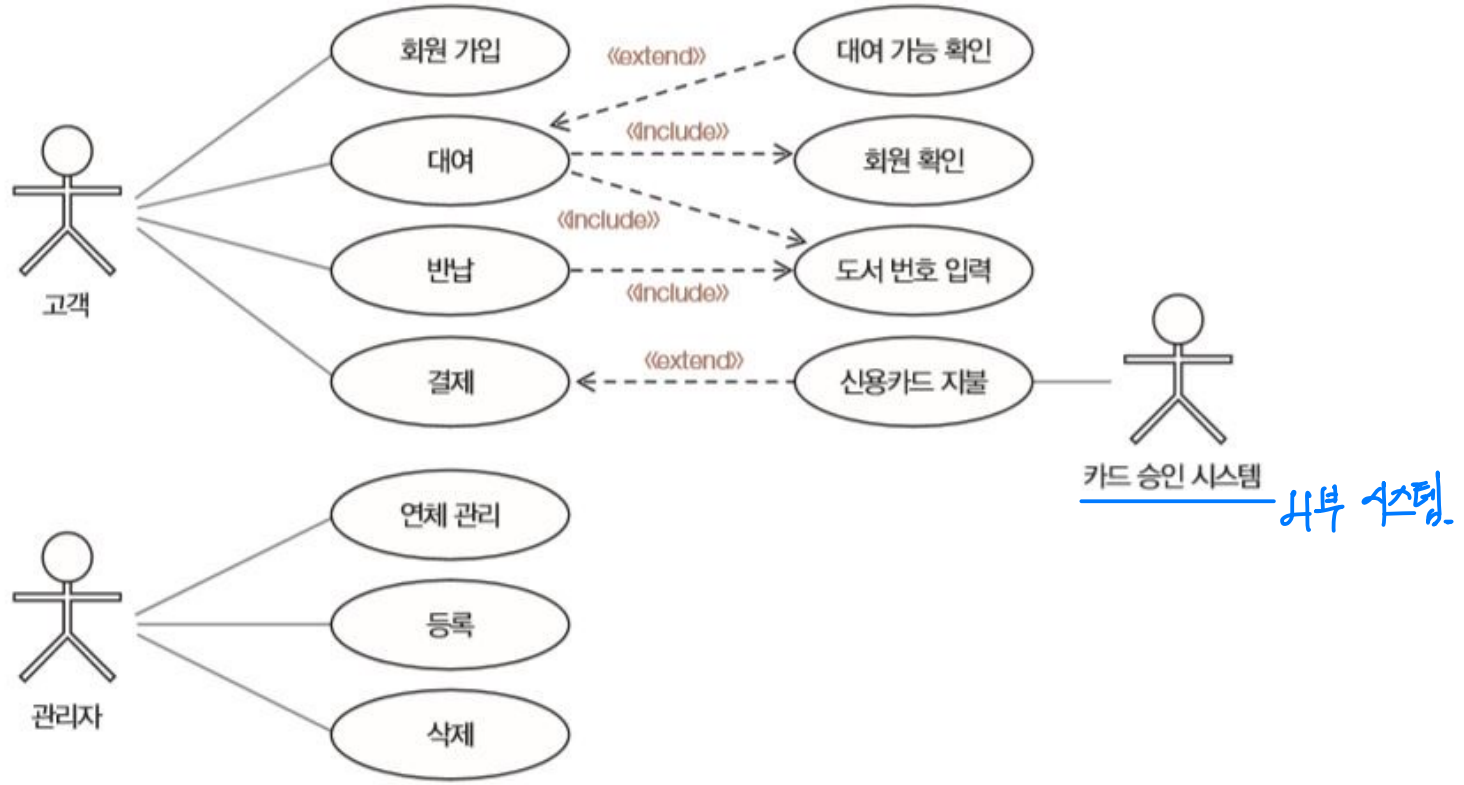



그림 3-21 깨비책방 관리 시스템의 유스케이스 다이어그램

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 5단계 : 유스케이스 명세서 작성

▫ 유스케이스명 : 회원 가입

▫ 액터명 : 고객(비회원)

▫ 유스케이스 개요 : 고객이 깨비책방 관리 시스템을 사용하기 위해 회원 가입을 하는 유스케이스다.

▫ 사전 조건 : 회원으로 가입되어 있지 않은 상태여야 한다.

▫ 이벤트 흐름

– 정상 흐름

① 기존 가입 회원인지 이름과 전화번호를 검색하여 확인한다(시스템).

② 회원 가입을 요청한다(액터).

③ 회원 약관을 보여준다(시스템).

④ 회원 약관에 동의한다(액터).

⑤ 회원 정보 입력 항목을 보여준다(시스템).

⑥ 회원 정보, 항목(이름, 전화번호)을 입력하고 등록을 요청한다(액터).

⑦ 입력된 정보를 확인한다(시스템).

⑧ 회원 정보를 저장, 등록한다(시스템).

– 선택 흐름

▶ 이미 가입된 회원인 경우 "이미 가입된 회원입니다" 메시지를 보여준다.

▶ 회원 약관에 동의하지 않을 경우 약관 동의하에 회원 가입 가능 오류 메시지를 보여주고 동의를 요청한다.

▶ 회원 정보 입력 항목 중 입력하지 않은 항목이 있을 경우, 오류 메시지를 띄우고 재입력을 요청한다.

▶ 등록 번호의 형식이 틀린 경우 메시지를 보여주고 재입력을 요청한다.

# 유스케이스 다이어그램의 단계별 모델링

- 도서 관리 시스템의 유스케이스 모델링
  - 6단계 : 유스케이스 실체화
    - 유스케이스 실체화는 도출된 유스케이스를 구현 시스템의 구성 요소로 구체화하는 과정
    - UML의 순차 다이어그램과 활동 다이어그램이 사용
      - 순차 다이어그램 : 이벤트 흐름을 나타냄
      - 활동 다이어그램 : 화면 흐름을 표현

표 3-6 요구 사항 정의 활동의 산출물

| 산출물 | | UML 다이어그램 | 필수 여부 |
|---|---|---|---|
| 요구 사항 모델 | 유스케이스 모델 | 유스케이스 다이어그램 | 필수 |
| | 유스케이스 명세서 | 이용 안 함 | 필수 |
| | 이벤트 흐름 모델 | 순차 다이어그램 | 선택 |
| | 화면 흐름 모델 | 활동 다이어그램 | 선택 |

# 요약 정리

- 요구사항의 표현
  - 객체지향 방법: 유스케이스 다이어그램
- 유스케이스 다이어그램 (Use Case Diagram)
  - 시스템 기능에 대한 사용자 입장을 표현한 다이어그램
- 유스케이스 다이어그램의 단계별 모델링
  1. 시스템 상황 분석
  2. 액터 식별
  3. 유스케이스 식별
  4. 유스케이스 다이어그램 작성
  5. 유스케이스 명세서 작성
  6. 유스케이스 실체화