

## Chapter 3

# 요구분석과 모델링

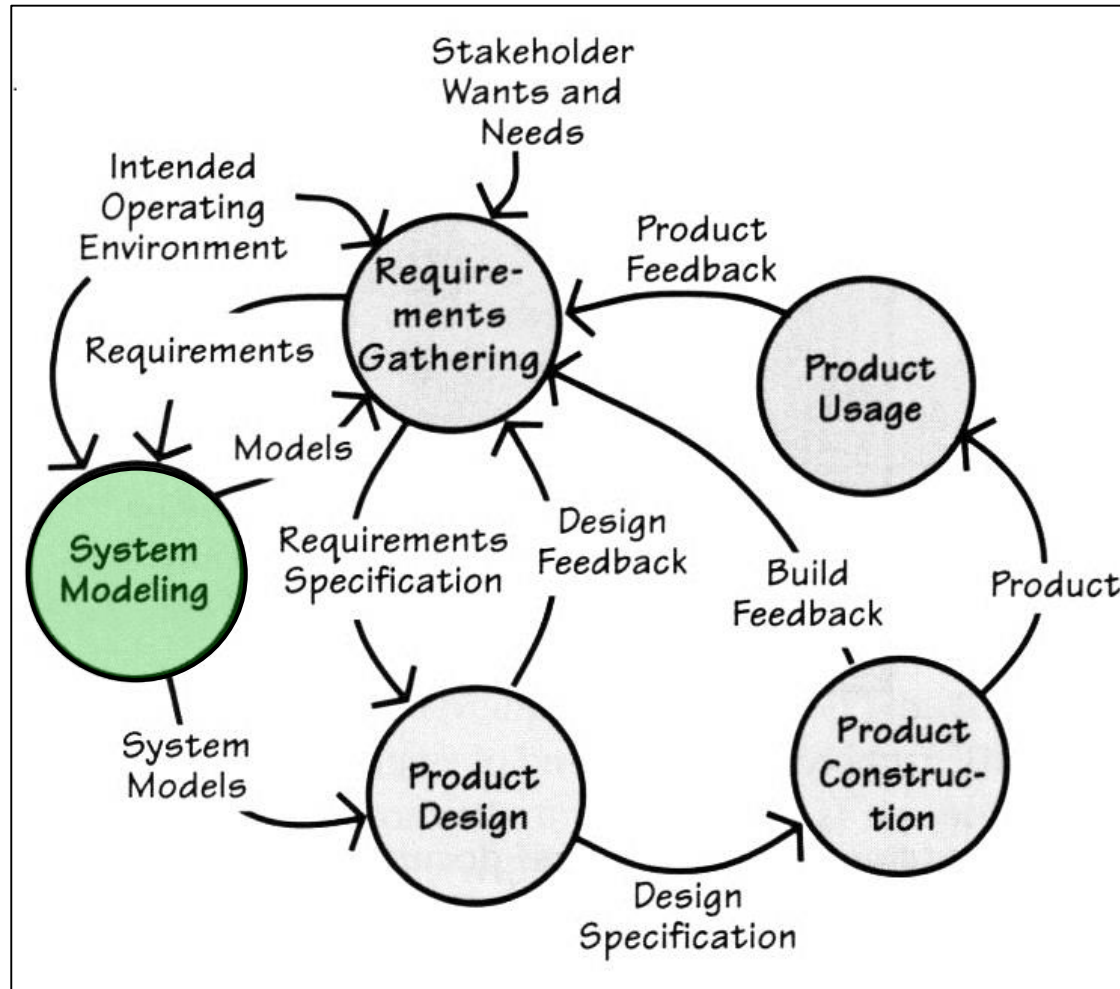
# 목 차

## Chapter 3. 요구분석과 모델링

- 요구 모델링 개요
- 요구모델링 활동
- 요구모델링 기법
- 요구모델의 분석

# 1. 요구 모델링 개요

## 모델링 단계



# 1. 요구 모델링 개요

## 모델은 Specification인가?

the task of communicating with users to determine what their requirements are

eliciting requirements

analyzing requirements

determining whether the stated requirements are unclear, incomplete, ambiguous, or contradictory, and then resolving these issues



recording requirements

documented in various forms, such as natural-language documents, use cases, user stories, or process specifications.

# 1. 요구 모델링 개요

## 요구 모델링의 목적

추상적 요구사항을 정확하고 상세하게 정의

- 식별된 고객의 needs 만족을 확인하는 방법 제공
- business owner's view를 architect's view으로 변환



- ♦ 참여자 요구 만족에 초점
- ♦ 불완전, 충돌, 중복, 누락 등 분석



고객의 요구에 대하여  
무엇을 수행할 것인가?

시스템 기능, 성능 및  
인터페이스 등을 정의



불완전하고 추상적인  
초기 요구사항  
(Candidate Requirement)



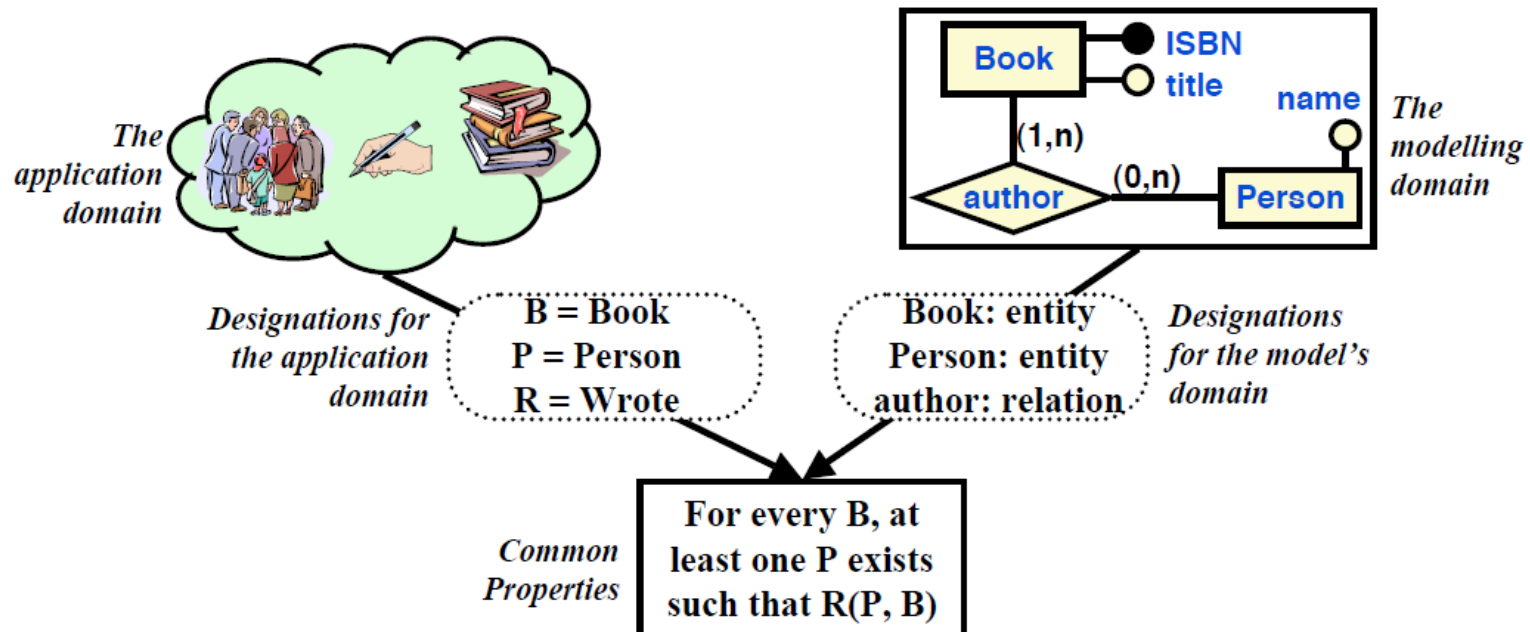
정확성, 일관성을 가진  
동의된 요구사항  
(Agreed Requirements)

- **요구사항 모델링** : 추상적 요구사항의 상세화
- **요구사항 우선순위화** : 위험 관리/ 충돌 해결을 위한 우선순위
- **요구사항 선정** : 다음 릴리즈에 포함될 요구사항 선정

# 1. 요구 모델링 개요

## 모델이란?

- ❑ 모델은 단순한 서술(description) 그 이상의 정보를 표현한다
- ❑ 모델은 스스로의 현상((phenomenon)과 이들의 관계를 표현하는 정보를 갖는다.
- ❑ 모델은 대상이 되는 현실의 현상을 충실히 반영할 때 한해 의미를 갖는다

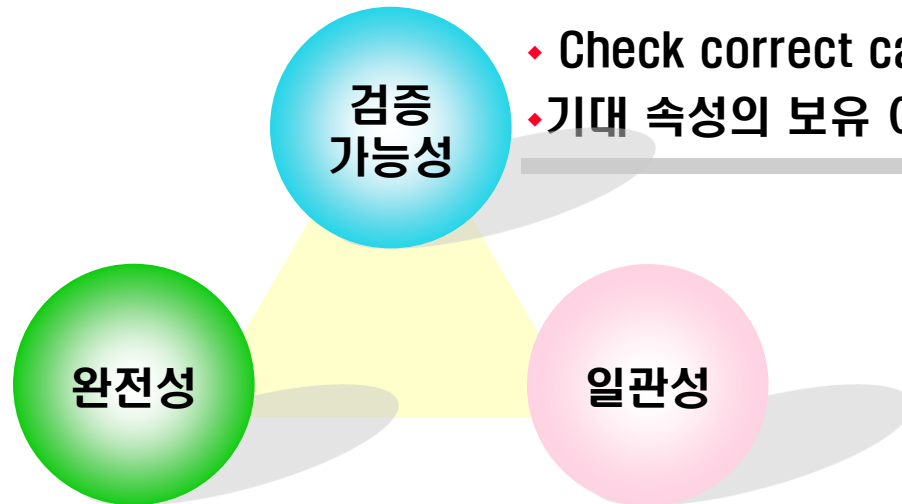


# 1. 요구 모델링 개요

## Communication 수단으로서의 모델

참여자 요구의 명확한 이해를 지원 : Visualization and Clarity

- 자연어의 **모호성 감소** : 수정 용이한 표기 - 특정 표기, 언어, 심볼 사용
- 요구사항의 **계층**을 생성
- 요구사항의 **기준선** 설정에 사용



- ♦ Check correct capturing of user needs
- ♦ 기대 속성의 보유 여부, 결론 예측, 요구사항 검증

- ♦ Guides elicitation
- ♦ 모델의 완전성 = 추출의 완전성

- ♦ Uncover Problems
- ♦ 모델 불일치 = 누락, 충돌, 불일치, 모호성

# 1. 요구 모델링 개요

## 모델링이란?

Modeling – the construction of **abstract descriptions** that are amenable to interpretation



- ❑ Models and problems must be partitioned in a manner that uncovers detail in layers
- ❑ Analysis proceeds from essential information toward implementation detail
- ❑ Information domain of problem must be presented & understood
- ❑ Models depicting systems information, functions, and behavior should be developed
- ❑ Must be traceable



# 1. 요구 모델링 개요

## 요구 모델링의 원리

### 추상화(Abstraction)

- Essential 측면 강조 : 상세함 숨김
- Concept 사이의 유사성 : 상세함 배제
- is\_a association

### 분할(Partitioning)

- 문제의 분할 – not a design, but a problem decomposition
- Aggregation/part-of

### 관점(Perspective)

- 다중 관점으로 모델의 aspect를 분리
- view of 관계 정의

# 1. 요구 모델링 개요

## 요구 모델링의 원리 - 추상화

- ❑ A way of finding similarities between concepts by ignoring some details
- ❑ Focuses on the general/specific relationship between phenomena
- ❑ Examples: Requirements is to handle faults on the spacecraft might group different faults into fault classes

### **Based on location**

- Instrumentation fault
- Communication fault
- Processor fault
- etc

### **Based on symptoms**

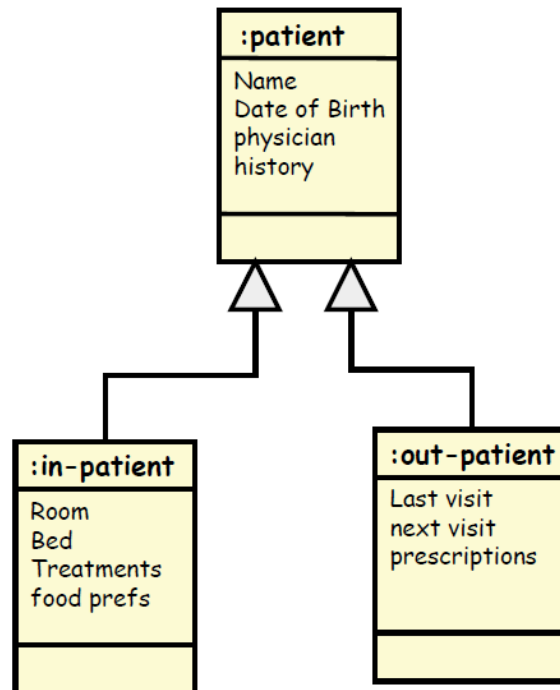
- No response from device
- Incorrect response
- Self-test failure
- etc

# 1. 요구 모델링 개요

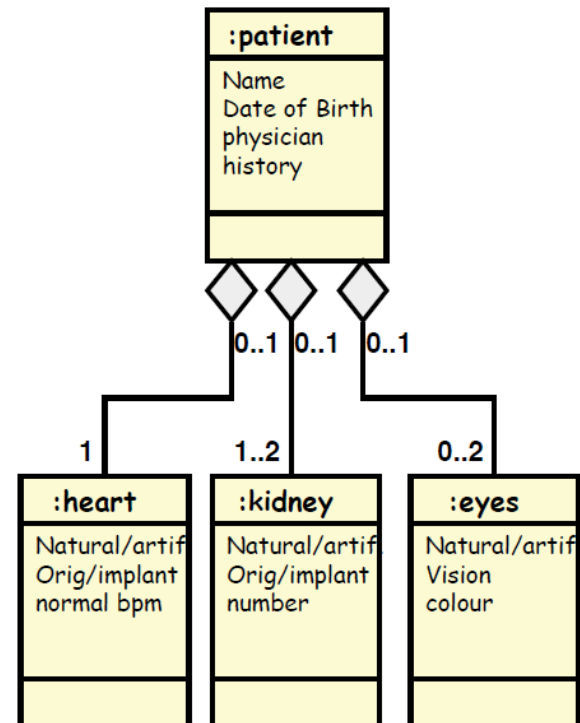
## 요구 모델링의 원리 - 분할

- Partitioning captures aggregation/part-of relationship
- Example: spacecraft problems have a partition of :
  - Guidance and navigation, Data handling
  - Command and control, environment control, etc.

**Generalization**  
(an abstraction hierarchy)



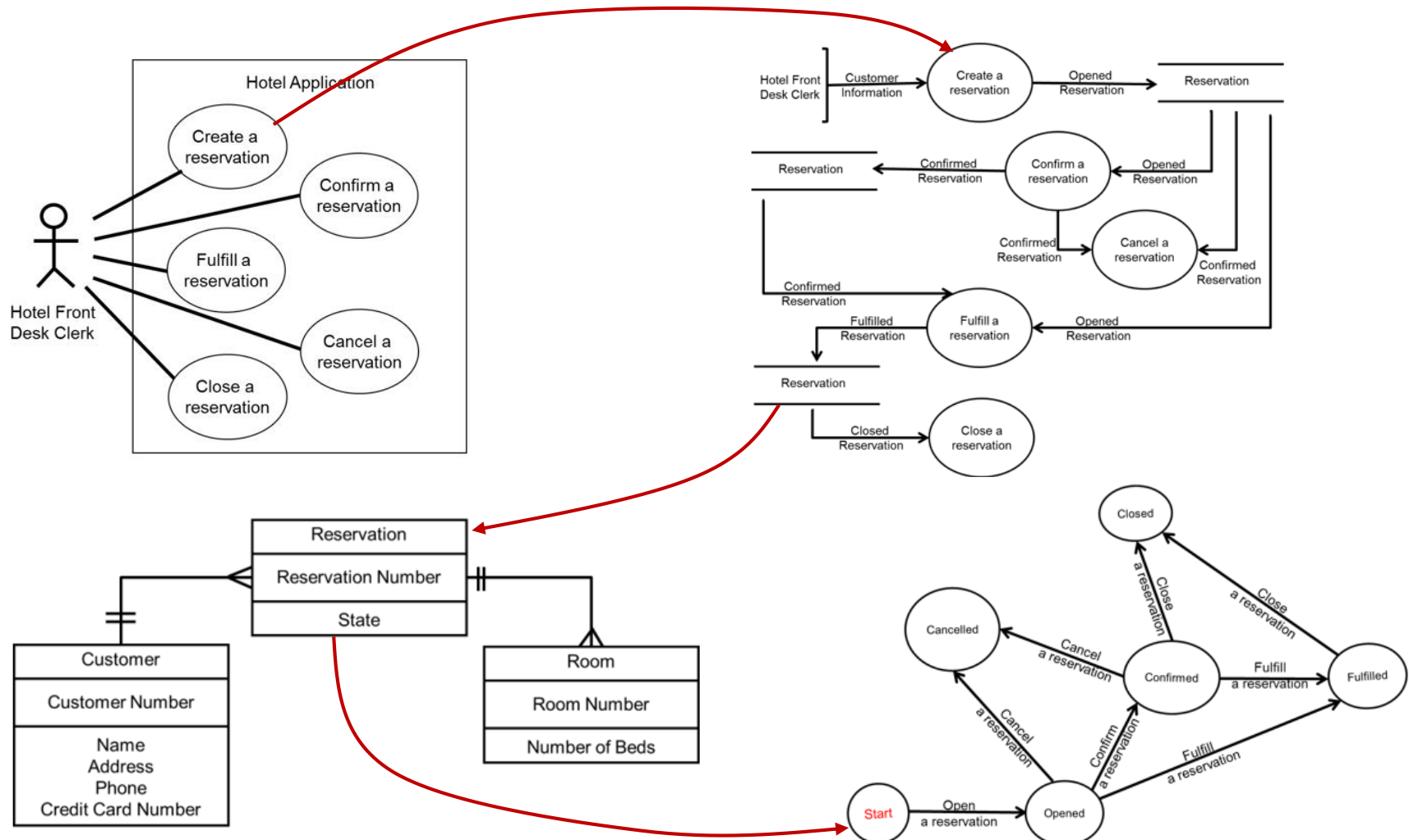
**Aggregation**  
(a partitioning hierarchy)



# 1. 요구 모델링 개요

## 요구 모델링의 원리 - 관점

- Projection separates aspects of model into multiple viewpoints



## 2. 요구 모델링 관점

### 모델링 관점의 선택

- ❑ **구조 관점:** Structured Analysis
  - Data Flow Diagram (DFD)
  - Entity-Relation Diagram (ERD)
  - State Transition Diagram (STD)
  - Class Diagram
- ❑ **기능 관점 :** Use Case Analysis, Goal and Scenario Based Analysis
  - Use Case Modeling (UC)
  - Goal-Scenario Modeling (GS)
- ❑ **행위 관점:** Behavior Analysis
  - Sequence Diagram, Communication Diagram (UML)

# 2. 요구 모델링 관점

## 5개의 요구 모델링 활동

### ❑ Enterprise Modeling

- deals with understanding an organisation's structure; business rules that affect its operation, the goals, tasks and responsibilities of its constituent members

### ❑ Data Modeling

- models information needs to be understood, manipulated and managed. e.g. Entity-Relationship-Attribute models, class models

### ❑ Behavioral Modeling

- models the dynamic or functional behaviour of stakeholders and systems, both existing and required.

### ❑ Domain Modeling

- domain model provides an abstract description of the **world in which an envisioned system will operate**

### ❑ Non-Functional Requirements Modeling

## 2. 요구 모델링 관점

### 모델링 관점에 따른 표현 대상

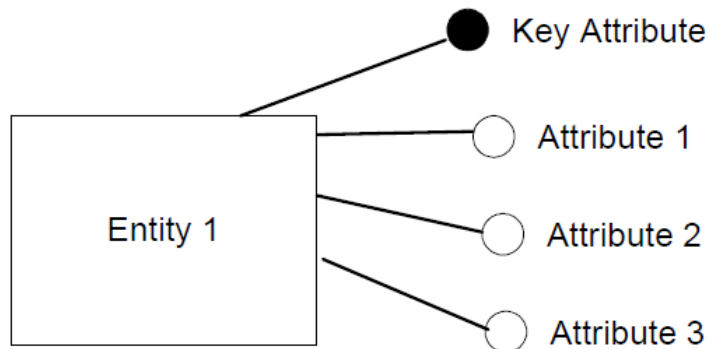
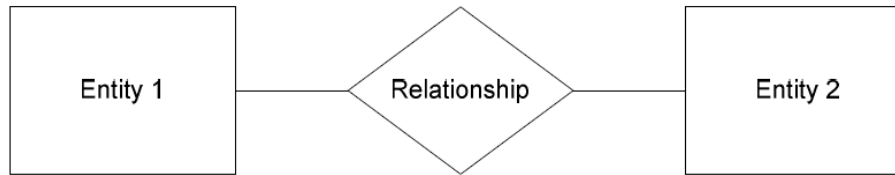
기술대상 정보	표현기술
시스템 외부 인터페이스	<ul style="list-style-type: none"><li>- 컨텍스트(Context) 다이어그램 (고수준의 표현)</li><li>- 유스케이스, Goal-Scenario 모델링</li></ul> <p>상세한 외부 인터페이스는 입출력 파일 포맷, 보고서 레이아웃 포함</p>
비즈니스 프로세스 흐름	<ul style="list-style-type: none"><li>-최상위수준 DFD (추상화된 비즈니스 프로세스 표현)</li><li>-플로우차트, 액티비티 다이어그램</li><li>-자연어 서술</li></ul> <p>정밀한 수준의 DFD, Swim lane 다이어그램은 정밀한 비즈니스 프로세스 표현이 가능</p>
데이터 정의 및 데이터 객체 관계	<ul style="list-style-type: none"><li>-ERD</li><li>-자료사전(Data Dictionary)</li></ul> <p>-자료사전은 데이터 구조와 개별 데이터 항목의 상세한 표현이 가능하다</p>
시스템 상태	<ul style="list-style-type: none"><li>-상태전이 다이어그램</li><li>-이벤트 응답표</li></ul>

# 3. 요구 모델링 기법

## 구조적 모델링 기법 : E-R Modeling

### ERD의 구성

- entities are typically the nouns and noun-phrases in the descriptive data produced in your analysis
- relationships means the semantic relationships between entities

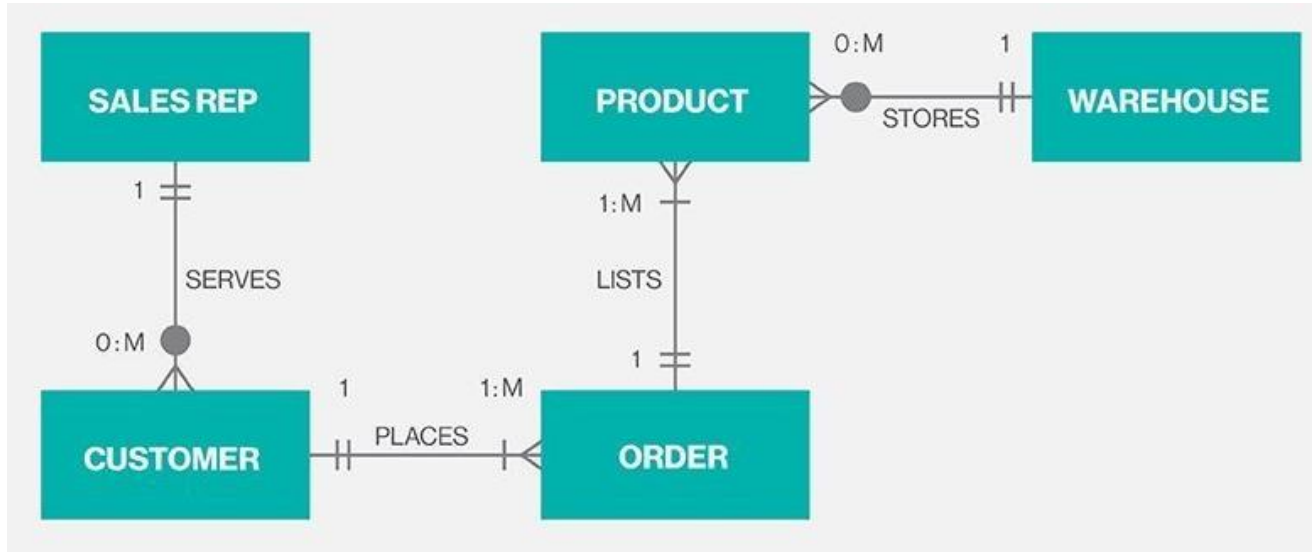




# 3. 요구 모델링 기법

## 구조 모델링 기법 : E-R Modeling

### ❑ Sales department ERD



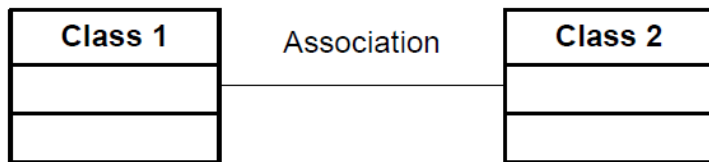
### ❑ Cardinalities

- A one-to-one relationship (1:1) : each customer in a database is associated with one mailing address.
- A one-to-many relationship (1:M): a single customer might place an order for multiple products.
- A many-to-many relationship (M:N): all call center agents work with multiple customers

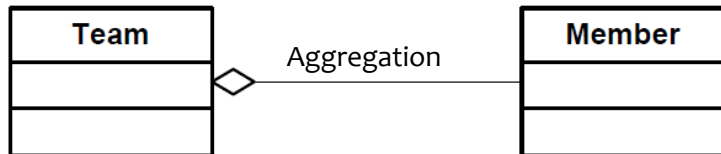
# 3. 요구 모델링 기법

## ERD와 클래스 다이어그램

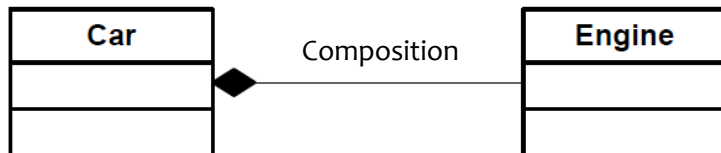
- ❑ Class diagrams and ERDs both model the structure of a system.
- ❑ Class diagrams represent the dynamic aspects of a system: both the structural and behavioral features.
- ❑ ERDs, depicting only structural features provide a static view of the system



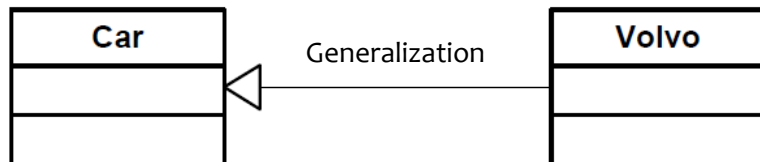
A (semantic) relationship between classes.  
A line that joins two classes.



"has-a" relationship



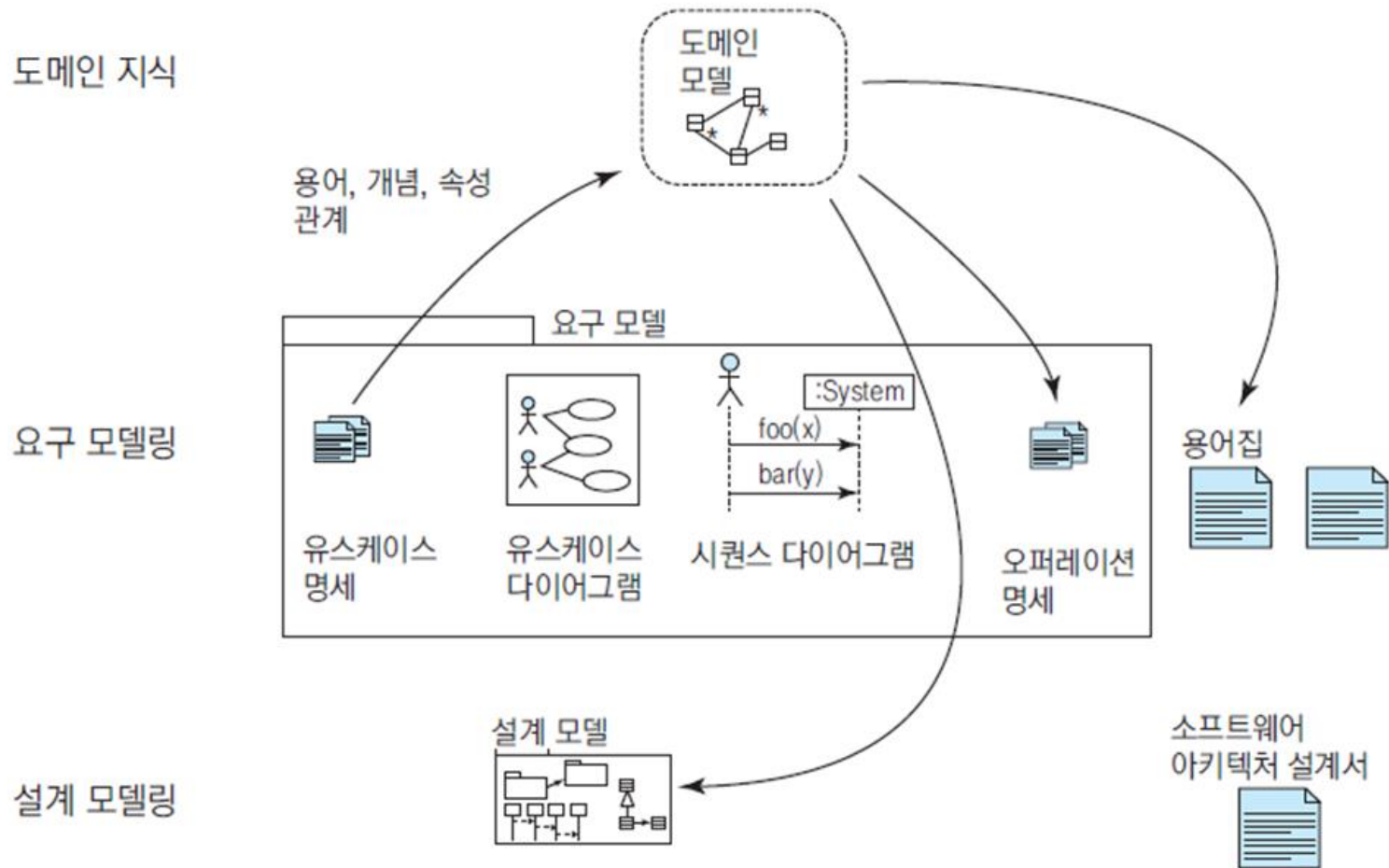
"is-composed-of" relationship



"is-a-kind-of" relationship

# 3. 요구 모델링 기법

## 구조 모델링 기법 : Class Model의 활용



# 3. 요구 모델링 기법

## 구조 모델링 기법 : Class Modeling

The University of Alpha has several departments. Each department is managed by a chair, and at least one professor. Professors must be assigned to one, but possibly more departments. At least one professor teaches each course, but a professor may be on sabbatical and not teach any course. Each course may be taught more than once by different professors. We know of the department name, the professor name, the professor employee id, the course names, the course schedule, the term/year that the course is taught, the departments the professor is assigned to, the department that offers the course

# 3. 요구 모델링 기법

## 구조 모델링 기법 : Class Modeling

### 1. 클래스의 식별

candidates are departments, chair, professor, course, course section

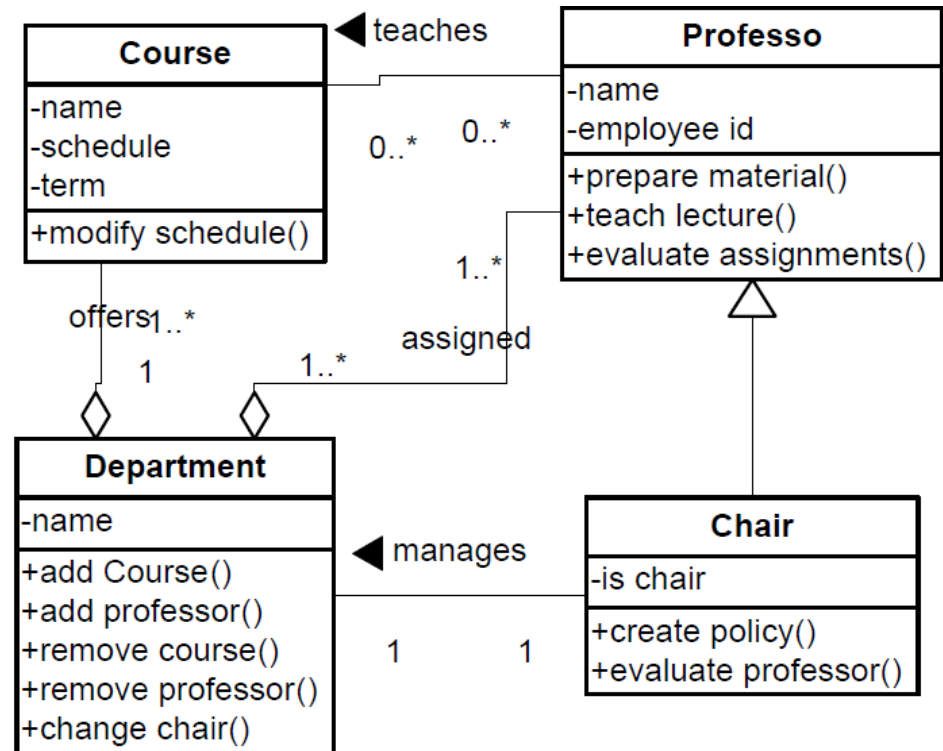
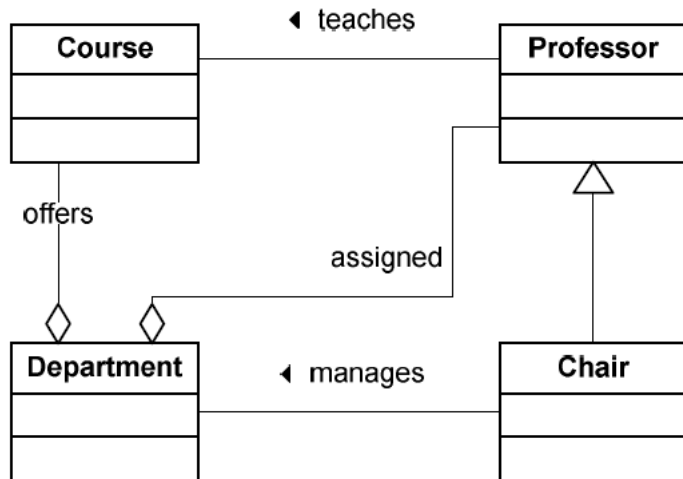
### 2. 연관 관계(Associations)정의 – class reference table

	department	chair	professor	course
department		managed by	is assigned (aggregate)	offers
chair	manages		is a	
professor	assigned to (aggregate)			teaches
course	offered by		taught by	

# 3. 요구 모델링 기법

## 구조 모델링 기법 : Class Modeling

- 3. 개략적인 클래스 다이어그램 구성
- 4. 다중성 (Multiplicity) 정보 추가
- 5. 속성정보 (Attributes) 식별
- 6. 오퍼레이션 정보 식별



# 3. 요구 모델링 기법

## Finite State Machine

- ❑ 상태: 시스템을 구성하는 값들의 집합으로 시간의 흐름에 따라 변한다
- ❑ 사용자가 요구하는 시스템의 작동 과정은 상태의 집합으로 표현이 가능하다
- ❑ 상태 기계의 구성을 통해 시스템이 작동하는 과정을 검증할 수 있다
- ❑ 프로그램: 시작상태 → 종료상태로 값들을 이동시킨다
- ❑ 유한상태기계의 구성
  - 상태 집합 (state set)
  - 천이 함수 (Transition function)
  - 오퍼레이션 : entry(), exit(), do()
- ❑ 상태 기계의 표현은 상태 천이 다이어그램 (State Transition Diagram) 혹은 상태 천이표 (State Transition Table) 로 표현한다

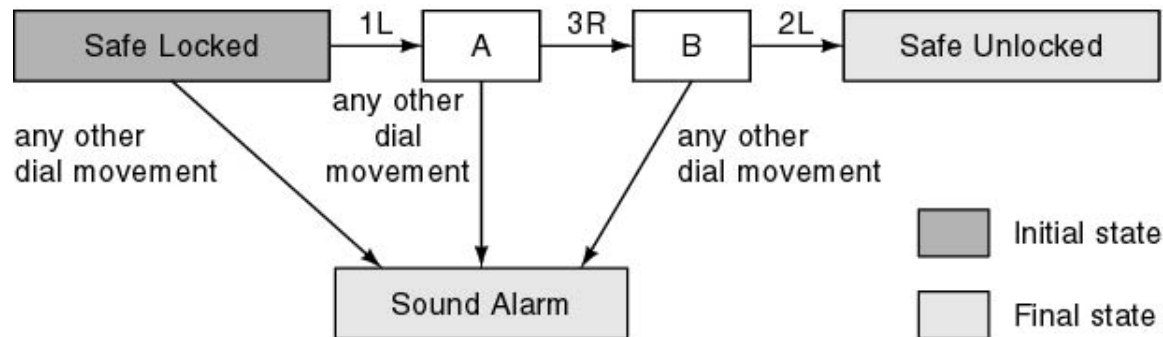
# 3. 요구 모델링 기법

## Finite State Machine

### ❑ Safe case study

A safe has a combination lock that can be in one of three positions, labeled 1, 2, and 3. The dial can be turned left or right (L or R). Thus, there are six possible dial movements, namely 1L, 1R, 2L, 2R, 3L, and 3R. The combination to the safe is **1L, 3R, 2L**. Any other dial movement will cause the alarm to go off

### ❑ State Transition Diagram for a Safe





# 3. 요구 모델링 기법

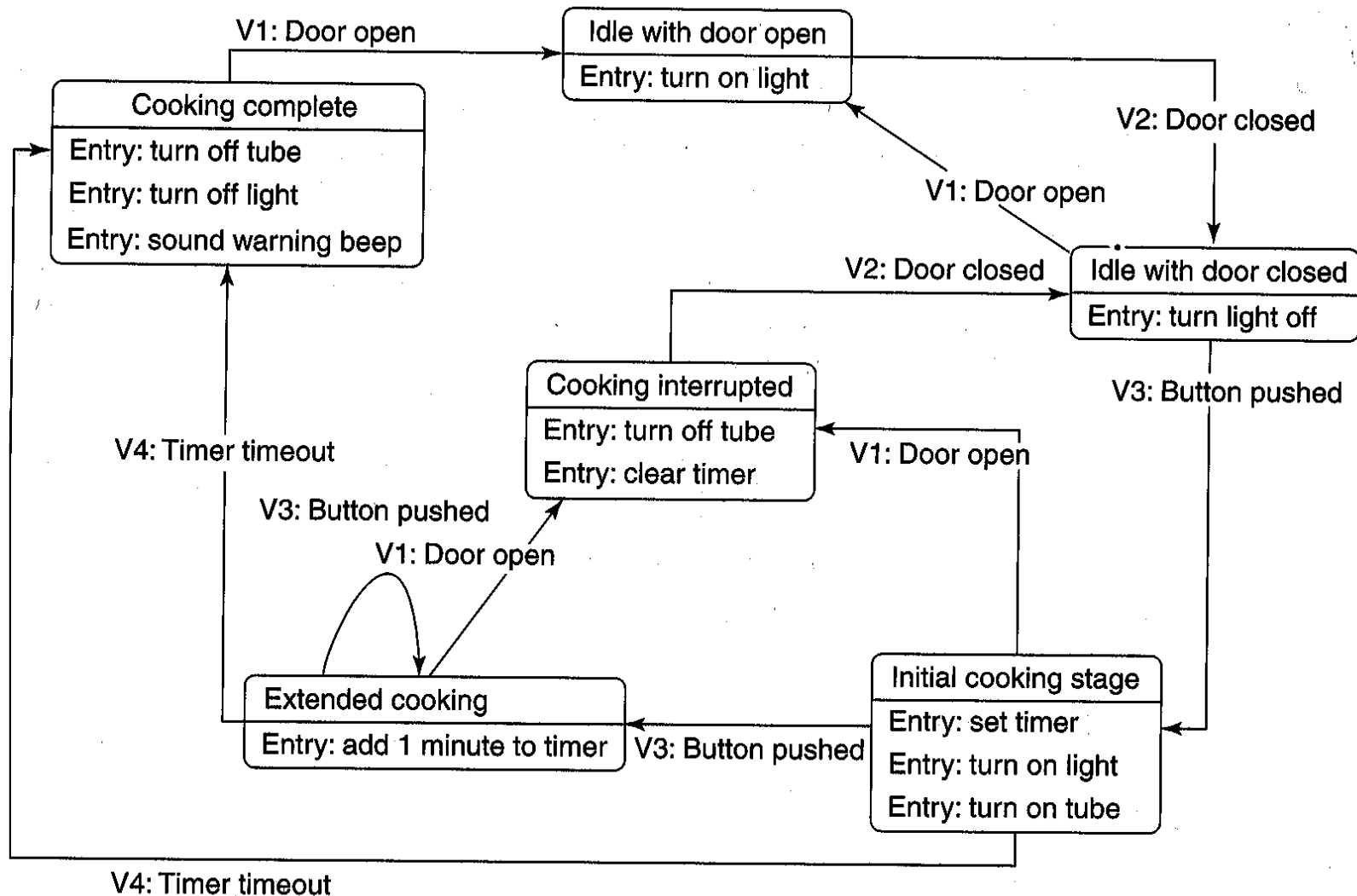
## Finite State Machine

### Transition table

		Table of Next States		
		Current state	A	B
Dial movement		Safe locked		
	1L	A	Sound Alarm	Sound Alarm
	1R	Sound Alarm	Sound Alarm	Sound Alarm
	2L	Sound Alarm	Sound Alarm	Safe Unlocked
	2R	Sound Alarm	Sound Alarm	Sound Alarm
	3L	Sound Alarm	Sound Alarm	Sound Alarm
	3R	Sound Alarm	B	Sound Alarm

# 3. 요구 모델링 기법

## Microwave Oven의 State Machine



# 3. 요구 모델링 기법

## 엘리베이터 문제 서술

**m 개 층으로 구성된 건물의 각 층에는 n 개의 엘리베이터가 있다. 엘리베이터 운행시 제어시스템은 다음 조건을 만족해야 한다.**

- 조건1. 한 엘리베이터 내에는 각 층을 의미하는 m 개의 버튼이 있다.**
- 조건2. 버튼이 눌러지면 버튼램프가 켜져 운행 층이 지정됨을 알린다.**
- 조건3. 해당 층에 도착하면 버튼램프가 꺼진다.**
- 조건4. 1층을 제외한 각 층에는 상, 하행용 2개의 호출버튼이 있다.**
- 조건5. 버튼이 눌러지면 버튼램프가 켜진다.**
- 조건6. 엘리베이터가 도착하여 원하는 층으로 이동이 시작되면 버튼램프가 꺼진다.**
- 조건7. 만일 엘리베이터에 대한 호출이 없으면 현재 층에서 문을 닫은 채 대기한다.**

# 3. 요구 모델링 기법

## 엘리베이터 상태 모델

### □ 버튼

- Elevator buttons

$EB(e, f)$ : Elevator Button in elevator  $e$  pressed to request floor  $f$

### □ 상태

$EBON(e, f)$ : Elevator Button  $(e, f)$  ON

$EBOFF(e, f)$ : Elevator Button  $(e, f)$  OFF

### □ 이벤트

$EBP(e, f)$ : Elevator Button  $(e, f)$  Pressed

$EAF(e, f)$ : Elevator  $e$  Arrives at Floor  $f$



# 3. 요구 모델링 기법

## 엘리베이터 상태 모델

### □ 전역 술어

$V(e,f)$ : Elevator  $e$  is Visiting (stopped at) floor  $f$

### □ 천이규칙

- If button is on **and** elevator arrives at floor  $f$ , **then** light turned off
- If light is off **and** button is pressed, **then** light comes on

$EBOFF(e,f) \wedge EBP(e,f) \wedge \neg V(e,f) \Rightarrow EBON(e,f)$

$EBON(e,f) \wedge EAF(e,f) \Rightarrow EBOFF(e,f)$

# 3. 요구 모델링 기법

## 엘리베이터 상태 모델

### ■ 층 버튼

FB(Floor Button on floor *f* that requests elevator traveling in direction *d*

### ■ 상태

FBON(*d*, *f*): Floor Button (*d*, *f*) ON

FBOFF(*d*, *f*): Floor Button (*d*, *f*) OFF

### ■ 이벤트

FBP(*d*, *f*): Floor Button (*d*, *f*) Pressed

EAF(1..*n*, *f*): Elevator 1 or ... or *n* Arrives at Floor *f*



# 3. 요구 모델링 기법

## 엘리베이터 상태 모델

### □ 술어

$S(d, e, f)$ : elevator  $e$  is visiting floor  $f$   
up ( $d = U$ ), down ( $d = D$ ), no requests ( $d = N$ )

### □ 천이 규칙

- If floor button is on and an elevator arrives at floor  $f$ , traveling in correct direction  $d$ , then light is turned off
- If light is off and a button is pressed, then light comes on

$$\begin{aligned} \text{FBOFF}(d, f) \wedge \text{FBP}(d, f) \wedge \neg S(d, 1..n, f) &\Rightarrow \text{FBON}(d, f) \\ \text{FBON}(d, f) \wedge \text{EAF}(1..n, f) \wedge S(d, 1..n, f) &\Rightarrow \text{FBOFF}(d, f), \\ &d = U \text{ or } D \end{aligned}$$

# Workshop: 엘리베이터 상태모델

## 엘리베이터 상태 모델링

엘리베이터의 문제 서술서를 바탕으로 엘리베이터의 작동에 관한 상태 모델을 작성하시오 (30분)

고려사항:

- 엘리베이터가 이동시 안전하게 이동하는 메커니즘은?
- 엘리베이터버튼, 층버튼의 호출에 대한 반응은?
- 엘리베이터의 상행, 하행에 관한 표현은?

Events

DC(e,f): Door Closed for elevator e, floor f

ST(e,f): Sensor Triggered as elevator e nears floor f

RL:        Request Logged (button pressed)



# Workshop: 엘리베이터 상태모델

## 엘리베이터 상태 모델링

### Events

DC(e,f): Door Closed for elevator e, floor f

ST(e,f): Sensor Triggered as elevator e nears floor f

RL: Request Logged (button pressed)

# 4. 유스케이스의 배경

## 유스케이스는 단순한 모델링 기법?

### 3 Amigos



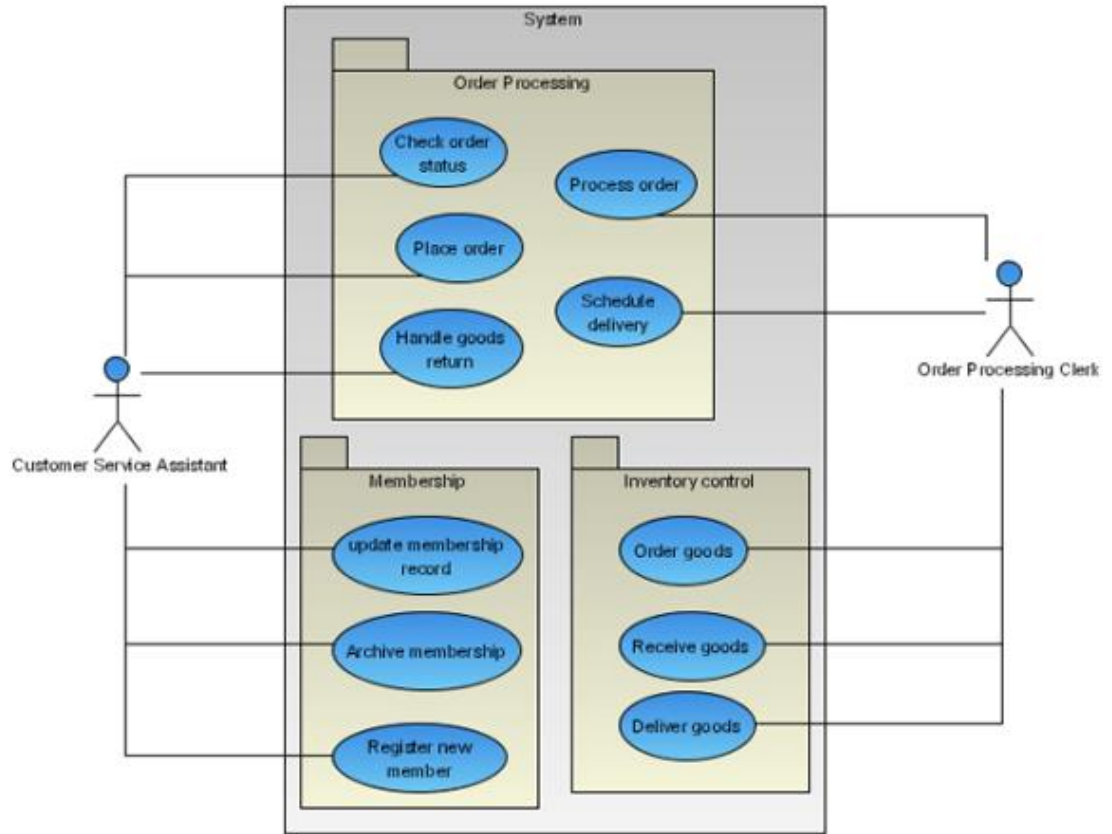
Grady Booch



Ivar Jacobson



James Rumbaugh



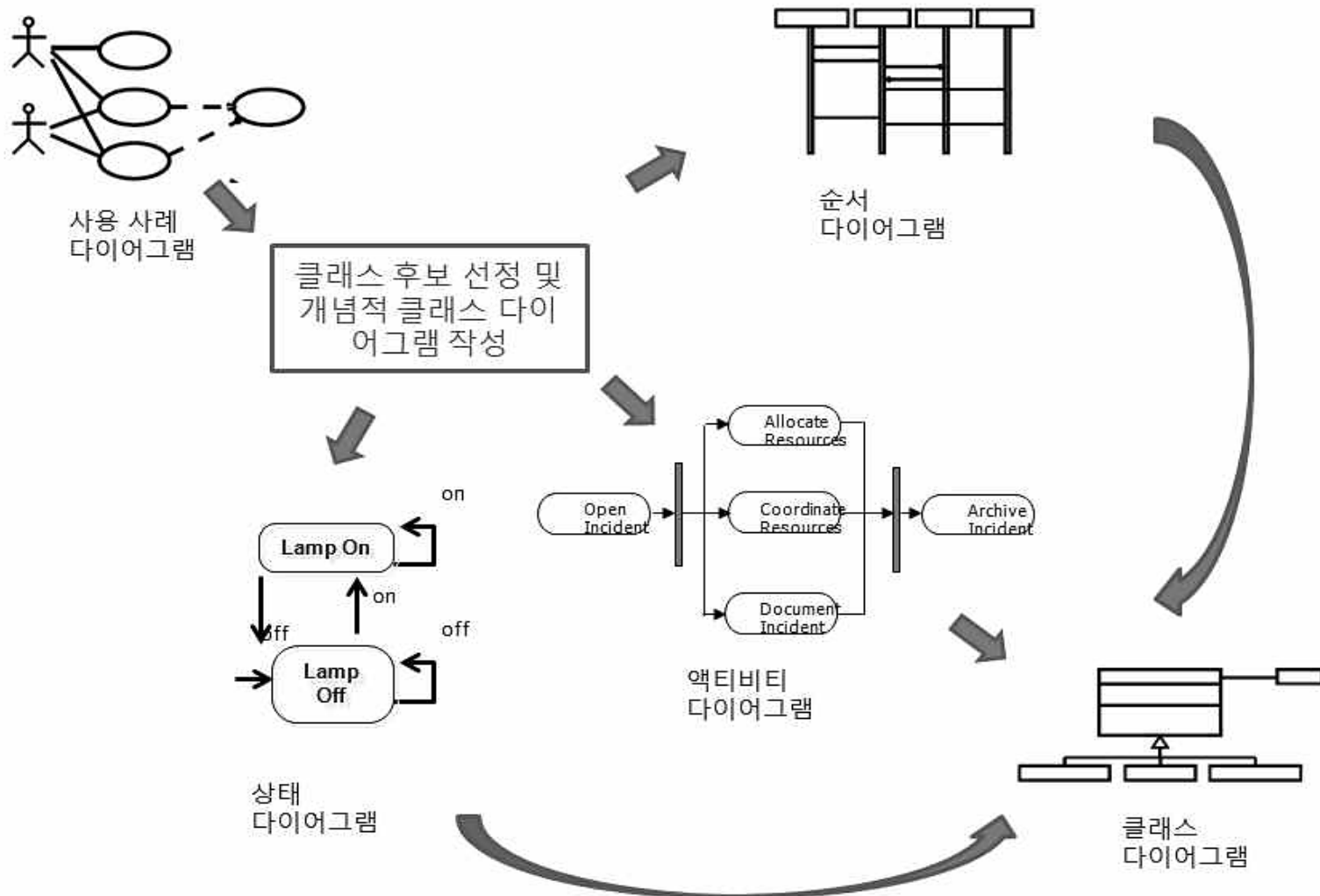
# 4. 유스케이스의 배경

## 유스케이스의 활용

- ❑ 전통적인 모델링 기법과 비교할 때 유스케이스는 상대적으로 쓰기 쉽고 읽기 쉽다
- ❑ 개발자로 하여금 사용자 관점에서 시스템을 서술할 수 있게 한다
- ❑ 요구가 충족되는 과정에서 발생하는 순서(ordering) 메커니즘을 제공한다
- ❑ 요구 파악 이후에 일어나는 분석, 설계, 구현 등 개발 활동에 중요한 정보를 제공한다
- ❑ 시스템이 요구 목표에 부합하는지를 판단할 수 있는 근거를 제공한다
- ❑ 기능 테스트를 위한 테스트 케이스 생성의 근거를 제공한다

# 4. 유스케이스의 배경

## 유스케이스의 활용



# 4. 유스케이스의 배경

## 유스케이스와 목표수준과 기능

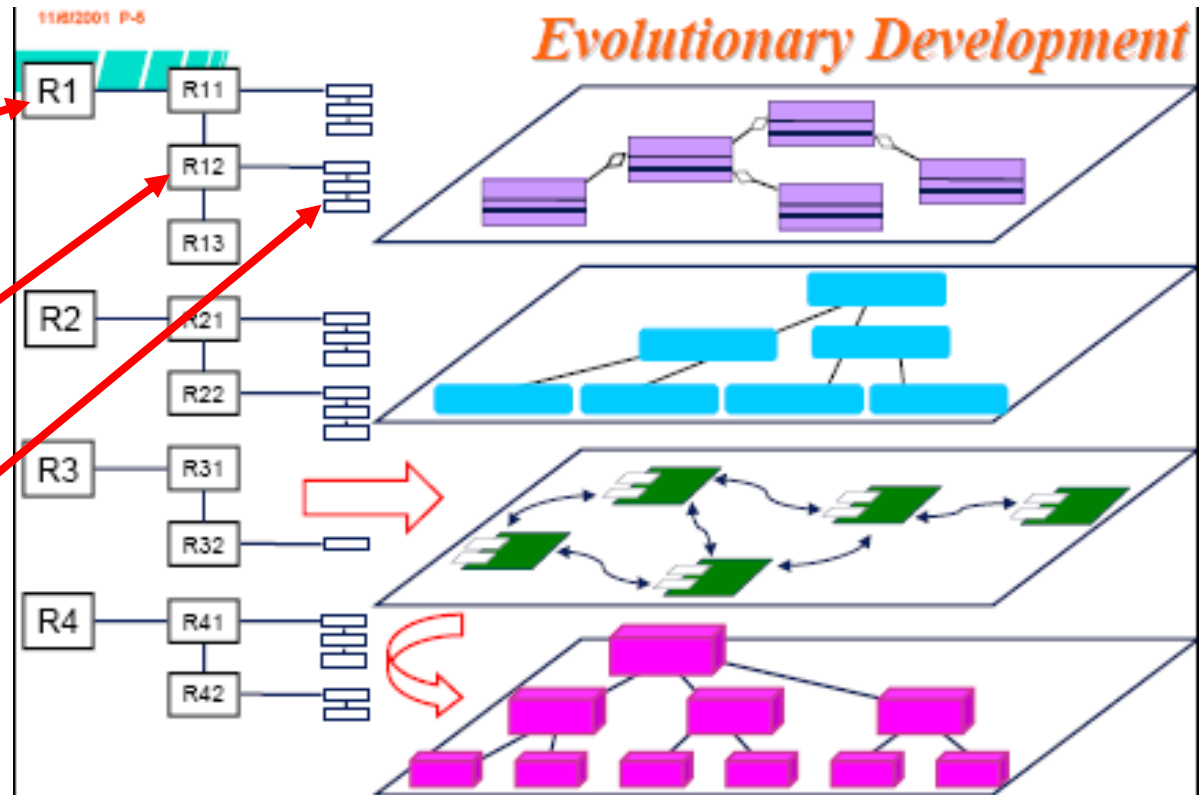
### 요구사항 정제의 필요성

- 요구사항의 모호성 발생 시
- 참여자간에 우선순위에 대해 다른 의견 발생 시
- 참여자간에 effort에 대한 다른 의견 발생 시

Define overall objective and operational scenarios

Define operational scenarios

Define functional requirements



# 5. 유스케이스의 개념

## 유스케이스 (use case)란?

- 유스케이스는 특정 **액터가 관찰할 수 있는 시스템의 수행 결과 값들을 생성하는 일련의 액션을 명세한다** (Leffingwell & Widrig '05)
  - 특정한 액터(actor): 개별 사람, 기계, 소프트웨어 등
    - ◆ 액션을 유발하는 주체 (관리인 홍길동)
  - 관찰할 수 있는 결과 값(observable result of value):
    - ◆ 가장 중요한 항목 중 하나임
    - ◆ 관리인이 점검 버튼을 누른다 ?
      - ↳ 관리인이 점검버튼을 누르면 시스템은 버튼의 불을 들어오게 한다
  - 일련의 액션(sequence of action):

# 5. 유스케이스의 개념

## 유스케이스와 시나리오



### □ 이해관계자 관점의 유스케이스

- 이해 관계자 중 일차 액터는 어떤 목표를 달성하기 위해 시스템과 상호작용을 시작한다
- 유스케이스는 일차 액터에 대한 시스템의 응답으로 다양한 조건하에 있는 시스템을 서술한다
- 유스케이스는 시스템 이해관계자들 간의 계약을 행위 중심으로 파악한다
- 유스케이스는 외부사용관점에서 시스템을 보는 것이다. 따라서 개발자가 아닌 사용자가 시스템을 보아야 한다

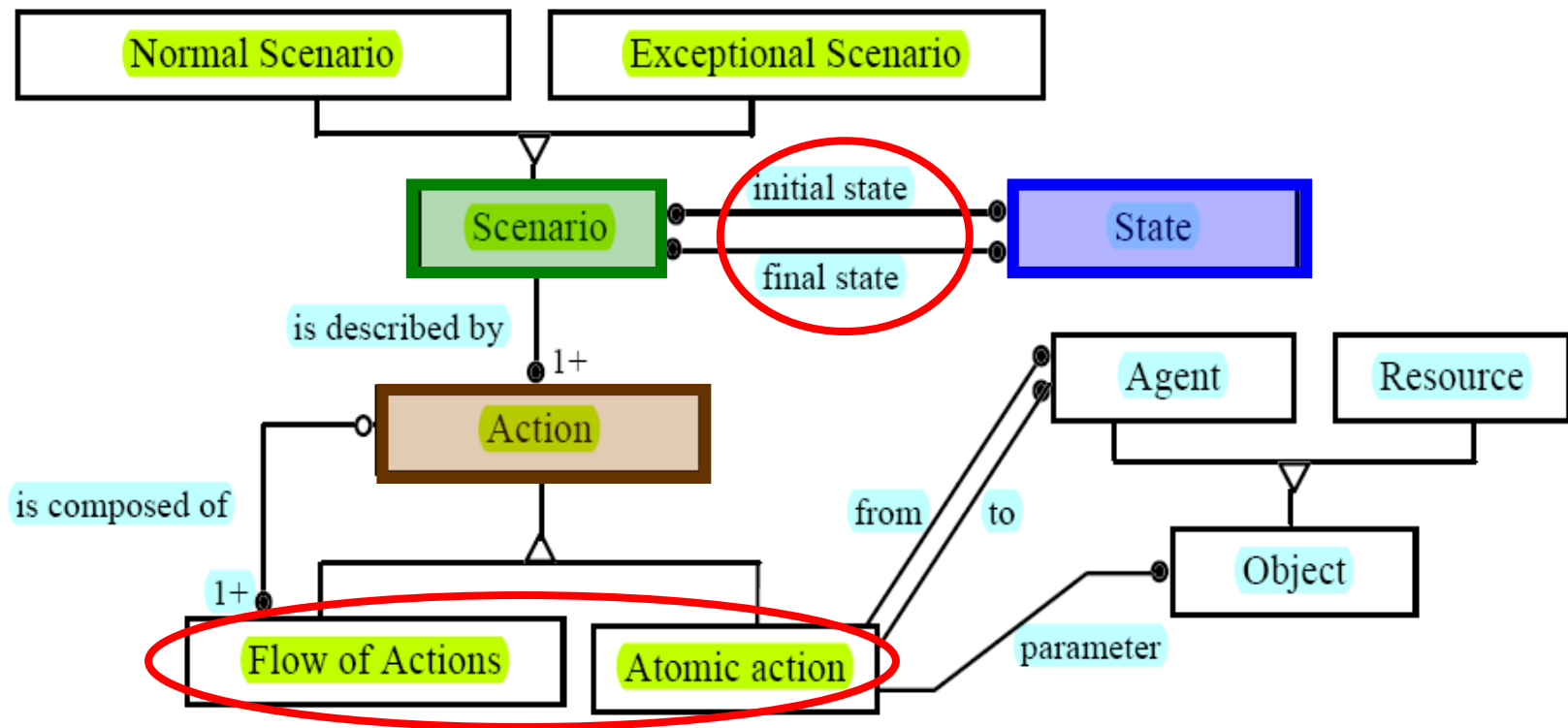
### □ 시나리오는 유스케이스의 현실적 전개

- 특별한 요청과 조건에 따라 서로 다른 일련의 시나리오가 전개된다
- 따라서 시나리오는 유스케이스의 인스턴스이다
- 예: VM의 "Buy Drink"부터 생성 가능한 시나리오는 매우 많음
- 개념적 구분을 떠나 시나리오와 유스케이스는 혼용해서 사용되기도 함

# 5. 유스케이스의 개념

## 유스케이스의 개념모델

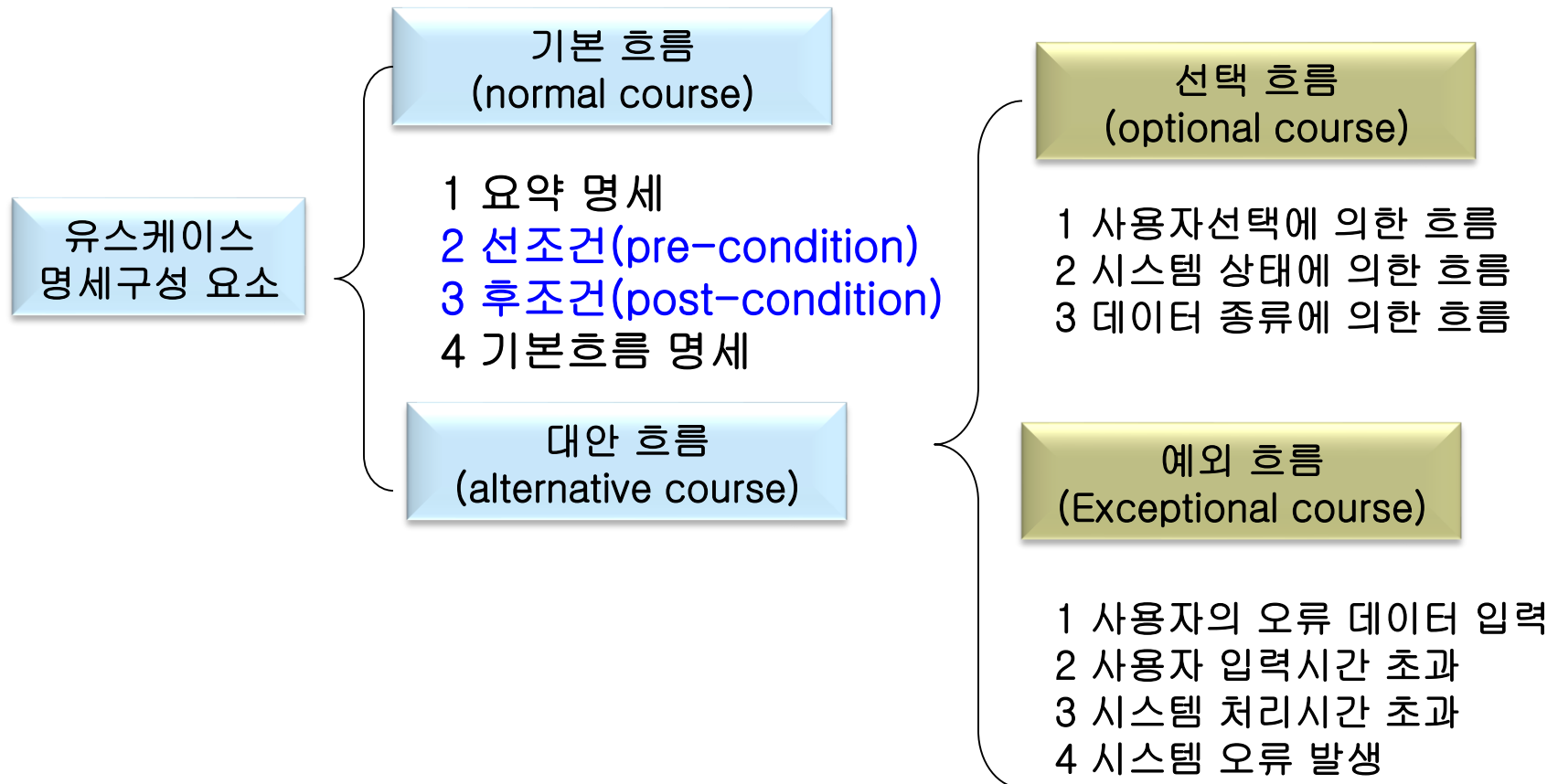
시나리오의 conceptual model





# 6. 유스케이스 명세 구성

## 기본흐름과 대안흐름



# 6. 유스케이스 명세 구성

## 선조건과 후조건

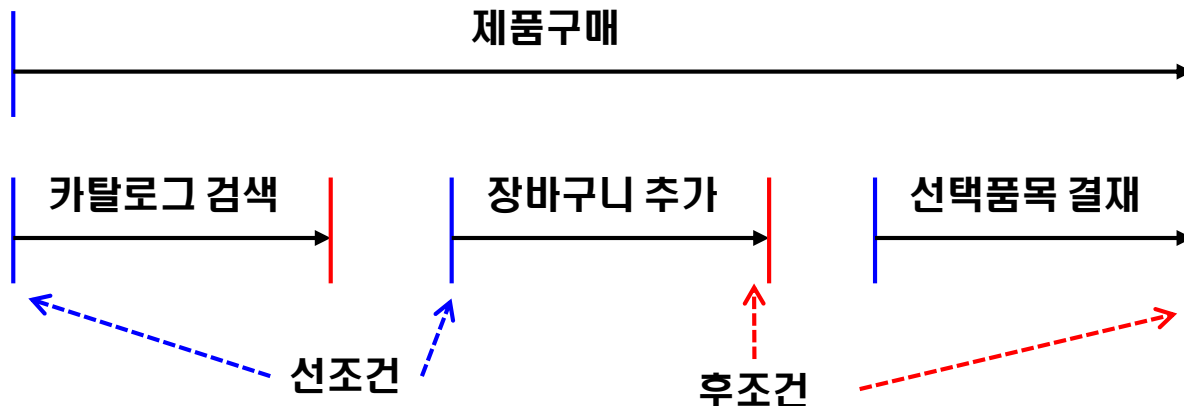
### 선조건(pre-condition)

- 유스케이스 진입을 위해 사전에 참으로 전제되어야 하는 조건 (들)
- 입력자료의 범위, 포맷 등

### 후조건(post-condition)

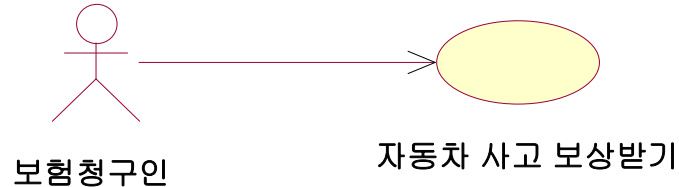
- 유스케이스 종료 후 항상 참으로 확인되어야 하는 조건 (들)
- 주로 출력자료의 형태, 조건 등

선조건과 후조건은 일련의 유스케이스 흐름을 더 큰 작업으로 묶는데 사용함



# 6. 유스케이스 명세 구성

## 자동차 사고 처리 유스케이스- 기본흐름



유스케이스명: 자동차 사고 보상받기

일차 액터: 보험청구인

선조건: 없음

후조건: 청구인과 보험회사가 보상금 액수에 합의하고 보상금이 지불된다

기본흐름

1. 청구인이 사실자료와 함께 지급요구서를 제출한다
2. 보험회사는 청구인의 자격을 확인하기 위해 보험증서를 검사한다
3. 보험회사가 이 사건을 조사할 대리인을 지정한다
4. 보험회사는 모든 세부사항이 보험증서의 약관과 일치하는지 검사한다
5. 보험회사는 청구인에게 보상금을 지불하고 사건을 종료한다

# 6. 유스케이스 명세 구성

## 대안흐름의 예

일차 액터: 보험청구인

선조건: 없음

후조건: 청구인과 보험회사가 보상금 액수에 합의하고 보상금이 지불된다

기본흐름

1. 청구인이 사실자료와 함께 지급요구서를 제출한다
2. 보험회사는 청구인의 자격을 확인하기 위해 보험증서를 검사한다
3. 보험회사가 이 사건을 조사할 대리인을 지정한다
4. 보험회사는 모든 세부사항이 보험증서의 약관과 일치하는지 검사한다
5. 보험회사는 청구인에게 보상금을 지불하고 사건을 종료한다

대안흐름

1a 제출된 자료가 충분치 않다

1a1. 보험회사는 누락된 정보를 요청한다

1a2. 청구인은 누락된 정보를 제출한다

2a 청구자가 유효한 보험증서를 가지고 있지 않다

2a1 보험회사는 지급요구를 거절하고, 청구인에게 통지 후 이 사실을 기록하고 사건을 종료한다

4a 사고내용이 보험증서 약관에 위배된다

4a1 보험회사는 지급요구를 거절하고, 청구인에게 통지 후 이 사실을 기록하고 사건을 종료한다

4b 사고내용이 보험증서 약관의 사소한 사항에 위배된다

4b1 보험회사가 보상금 액수에 대해 청구인과 협상을 시작한다

# 7. 유스케이스 모델링

## 유스케이스 개발 단계

**단계 1: 액터 파악 및 서술**

**단계 2: 유스케이스 파악**

**단계 3: 개별 유스케이스의 개요 서술**

**단계 4: 유스케이스의 정제**

# 7. 유스케이스 모델링

## 액터 파악

- ❑ 시스템과 상호작용하는 사람(someone) 혹은 어떤 것(something)
- ❑ 다음의 질문을 통해 액터에 대한 필요 정보를 얻음
  - 누가 시스템을 사용하는가?
  - 누가 시스템으로부터 정보를 얻게 되는가?
  - 누가 시스템에 정보를 입력하는가?
  - 어디에 시스템이 사용되게 되는가?
  - 누가 시스템을 지원하거나 유지보수하는가?
  - 시스템을 사용하는 다른 어떤 시스템이 존재하는가?
- ❑ UML 2 은 액터간의 직접적인 연관 (associations)은 허용하지 않음  
(예외적으로 generalization/specialization relationship은 허용함)

# 7. 유스케이스 모델링

## 유스케이스의 파악

- ❑ 분석가는 사용자들이 새로운 시스템을 통해 얻고자 하는 목적을 잘 설명할 수 있는 질문을 해야 함
- ❑ 유스케이스 파악을 위해 물어보는 질문들
  - 액터는 어떤 용도로 시스템을 사용하려는가?
  - 액터는 시스템에 자료를 입력하거나 저장, 변경, 제거, 혹은 참조하게 되는가?
  - 액터는 어떤 외부 이벤트나 이의 변화를 시스템에 알려주는가?
  - 시스템은 액터에게 어떤 특정한 변화를 알려주는가?
- ❑ 유스케이스가 되기 위해서는 “그것이 범위 안에 있나요?” 라는 질문을 통과해야 함

# 7. 유스케이스 모델링

## 유스케이스 범위 파악하기

### □ 항공예약시스템의 예

- 여향을 위한 항공편 검색하기 (find available flights for an itinerary)
- 항공편 예약하기 (make a flight reservation)
- 좌석 선택하기 (select seats)
- 여행일정 출력하기 (print an itinerary)
- 예약변경하기 (change reservation)
- 예약 취소하기 (cancel a reservation)

- 항공편 상황 확인하기 (check on flight status) ← 범위 안에 있는가?



# 7. 유스케이스 모델링

## 유스케이스 개요 서술

### □ 기본흐름의 파악을 위한 질문들

- 어떤 이벤트가 유스케이스를 시작시키는가?
- 유스케이스는 어떻게 끝나는가?
- 어떻게 유스케이스는 특정 행위를 반복하는가?

### □ 대안 흐름의 파악을 위한 질문들

- 유스케이스에는 선택적인 상황이 존재하는가?
- 특별한 케이스가 발생할 수 있는가?
- 유스케이스에 변화(variation)는 가능한가?
- 잘못될 수 있는 경우는 있는가?
- 잘 발생되지 않은 경우는 무엇인가?

# 8. 유스케이스 명세

## 유스케이스 형식- 기본형

유스 케이스 명

Use case: PayVAT

유스케이스 ID

ID: UC1

액터 리스트

**Actors:**

Time

Government

유스케이스 시작 전의  
시스템 상태

**Preconditions:**

1. It is the end of a business quarter

실제 유스케이스 스텝

**Flow of events:**

- 1.The use case starts when it is the end of the business quarter
2. The system determines the amount of VAT owed to the Government.
- 3.The system sends an electronic payment to the Government

유스케이스 종료 후의  
시스템 상태

**Postconditions:**

1 The Government receives the correct amount of VAT

# 8. 유스케이스 명세

## 유스케이스 형식 – if 분기형

유스 케이스 명

Use case: ManageBasket

유스케이스 ID

ID: UC10

액터 리스트

**Actors:**  
Customer

유스케이스 시작 전의  
시스템 상태

**Preconditions:**  
The shopping basket contents are visible

실제 유스케이스 스텝

**Flow of events:**  
1. The use case starts when the Customer selects an item in the basket  
2. **If the Customer selects "delete item"**  
    2.1 The system removes the item from the basket  
3. **If the Customer types in a new quantity**  
    3.1 The system updates the quantity of the item in the basket

유스케이스 종료 후의  
시스템 상태

**Postconditions:**  
The basket contents have been updated

# 8. 유스케이스 명세

## 유스케이스 형식 – 선택 분기

### Use case: DisplayBasket

ID: UC11

#### Actors:

Customer

#### Preconditions:

1. The Customer is logged on the system

#### Flow of events:

1. The use case starts when the Customer selects "display basket"
2. If there are no items in the basket
  - 2.1 The system informs the Customer that there are no items in the basket
  - 2.2 The use case terminates
- 3 The system displays a list of all items in the Customers shopping basket including product ID, name, quantity and item price

#### Postconditions:

#### Alternative flow 1:

1. At any time the Customer may leave the shopping basket screen

#### Alternative flow 2:

1. At any time the Customer may leave the system

# 8. 유스케이스 명세

## 유스케이스 형식 – for 반복

### Use case: FindProduct

ID: UC12

#### Actors:

Customer

#### Preconditions:

1. The Customer is logged on the system

#### Flow of events:

1. The Customer selects “find product”
2. The system asks the Customer for search criteria
3. The Customer enters the requested criteria
4. The system searches for products that match the Customer’s criteria
5. If the system finds some matching products then
  - 5.1 For each product found
    - 5.1.1 The system displays a thumbnail sketch of the product
    - 5.1.2 The system displays a summary of the product details
    - 5.1.3 The system displays the product price
6. Else
  - 6.1 The system tells the Customer that no matching product could be found

#### Postconditions:

#### Alternative flow 1:

1. At any point the Customer may move to different page.

# 8. 유스케이스 명세

## 유스케이스 형식 - 대화형

### Use case: DisplayBasket

ID: UC13

#### Actors:

Customer

#### Preconditions:

1. The Customer is logged on the system

#### Flow of events:

Customer	System
1.selects "find product"	
3. enters the requested criteria	2. asks the Customer for search criteria
	4. searches for products that match the Customer's criteria
	5. If the system finds some matching products then
	5.1 For each product found
	5.1.1 displays a thumbnail sketch of the product
	5.1.2 displays a summary of the product details

#### Postconditions:

#### Alternative flow 1:

1. At any time the Customer may leave the shopping basket screen

# 8. 유스케이스 명세

## 유스케이스 형식 – UML 시퀀스 다이어그램과의 결합

### Use case: Offering

ID: UC14

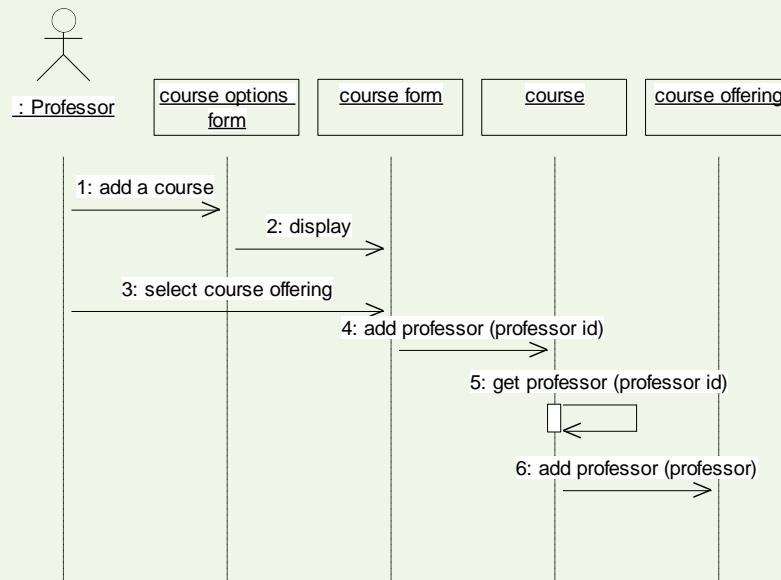
#### Actors:

Professor

#### Preconditions:

1. The Professor is logged on the system

#### Flow of events:



#### Postconditions:

#### Alternative flow 1:

1. At any time the Professor may leave the shopping basket screen

# 8. 유스케이스 명세

## 유스케이스의 추가 기능

### □ 액터 일반화

- 더 일반적인 액터와 더 특정한 액터와의 일반화 관계를 표현할 때 사용한다

### □ 유스케이스 일반화

- 더 일반적인 유스케이스와 더 특정한 유스케이스의 일반화 관계를 표현할 때 사용한다
- 액터와 유스케이스의 일반화는 다이어그램을 간략하게 할 필요가 있을 때만 사용하여야 한다

### □ 유스케이스의 포함 <<include>>

- 다른 유스케이스의 행위(behavior)를 자신의 유스케이스에 포함시키고자 할 때 사용한다

### □ 유스케이스의 확장 <<extend>>

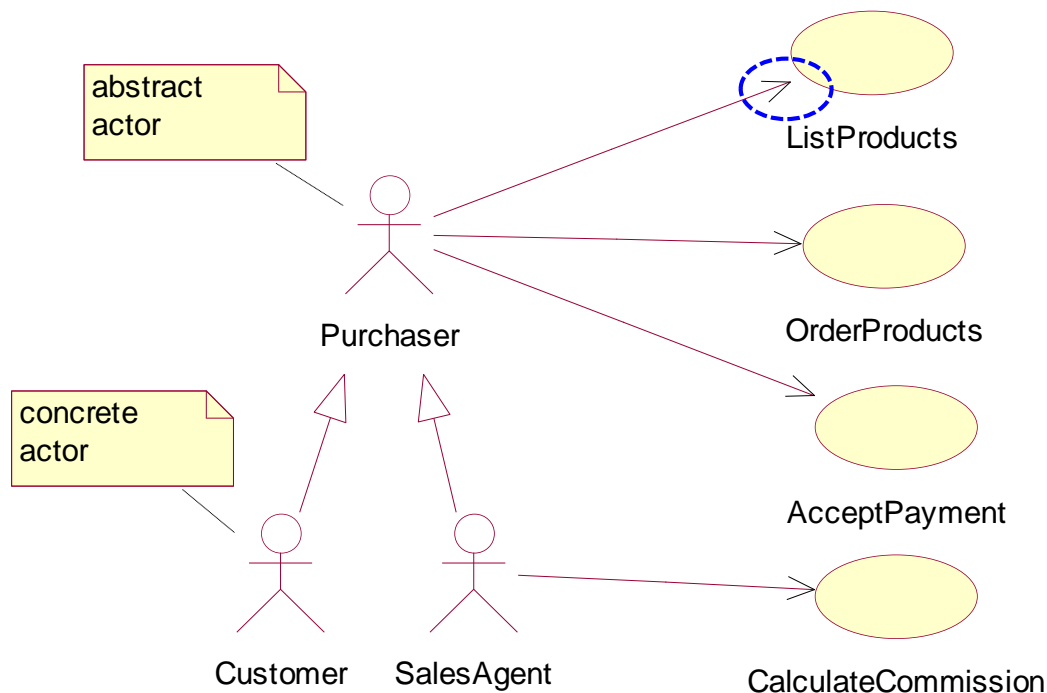
- 다른 유스케이스의 특정 부분을 하나 혹은 그 이상을 자신의 유스케이스에 확장하고자 할 때 사용한다



# 8. 유스케이스 명세

## 액터의 일반화

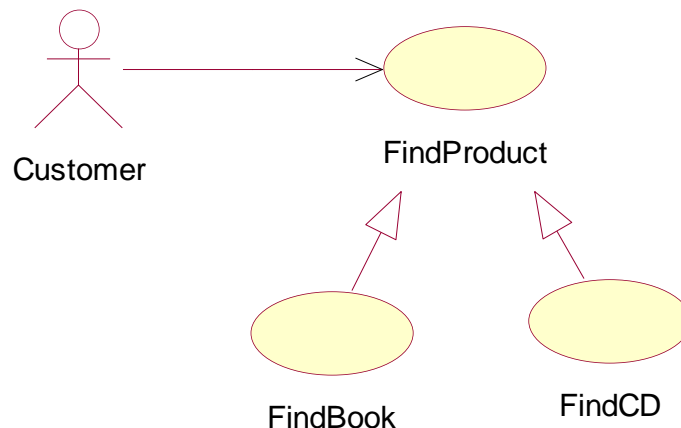
- ❑ 사용시점: 두 액터가 공동된 행위를 통해 커뮤니케이션할 경우
- ❑ 의미: 자식 액터는 부모 액터가 갖는 유스케이스와의 관계와 역할을 그대로 상속받는 개념이다.



# 8. 유스케이스 명세

## 유스케이스의 일반화

- ❑ 자식 유스케이스는 부모 유스케이스의 모든 특징을 상속받는다
- ❑ 이에 더하여 모든 필요 부분을 추가할 수 있으나 재정의(override)는 제한적으로 허용된다
- ❑ 부모 유스케이스는 추상적인 수준에서 서술되므로 모든 행동이 완성적일 필요는 없다
- ❑ 자식 클래스에서는 모든 것이 구체적이고 완성적으로 서술되어야 한다



# 8. 유스케이스 명세

## 일반화 명세의 예 – 부모 유스케이스

### Use case: FindProduct

ID: UC12

#### Actors:

Customer

#### Preconditions:

1. The Customer is logged on the system

#### Flow of events:

1. The Customer selects "find product"
2. The system asks the Customer for search criteria
3. The Customer enters the requested criteria
4. The system searches for products that match the Customer's criteria
5. If the system finds some matching products then
  - 5.1 The system displays a list of matching products
6. Else
  - 6.1 The system tells the Customer that no matching product could be found

#### Postconditions:

#### Alternative flow 1:

1. At any point the Customer may move to different page.

#### Postconditions:

# 8. 유스케이스 명세

## 일반화 명세의 예 – 자식 유스케이스

### Use case: FindBook

ID: UC16

#### Actors:

Customer

#### Preconditions:

1. The Customer is logged on the system

#### Flow of events:

1. The Customer selects "find book"

2. The system asks the Customer for search criteria consisting of author name, title, ISBN, topic

3. The Customer enters the requested criteria

4. The system searches for books that match the Customer's criteria

5. If the system finds some matching books then

5.1 The system displays a page showing details of a maximum of five books

5.2 For each book on the page the system displays the title, author, price and ISBN

5.3 while there are more books

5.3.1 The system gives the Customer the option to display the next page

6. Else

6.1 The system redisplay the "find book" search page

6.2 The system tells the Customer that no matching product could be found

#### Postconditions:

#### Alternative flow 1:

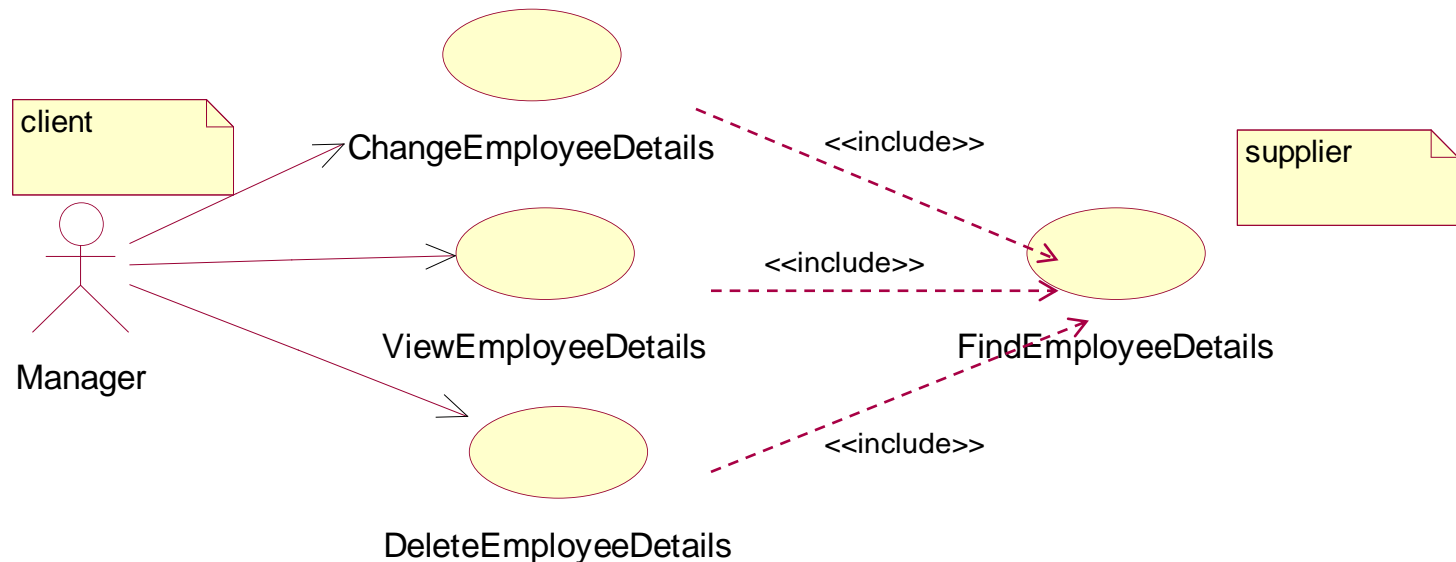
1. At any point the Customer may move to different page.

#### Postconditions:

# 8. 유스케이스 명세

## 포함 유스케이스 <<include>>

- ❑ 재사용의 개념에서 반복되어 서술되는 유스케이스가 있는 경우를 간결한 서술을 하도록 한다
- ❑ 포함을 사용하는 유스케이스는 클라이언트로, 포함을 제공하는 유스케이스를 제공자 역할을 한다



# 8. 유스케이스 명세

## <<include>> 유스케이스 예

### Use case: ChangeEmployeeDetails

ID: UC1

**Actors:**

Manager

**Preconditions:**

1. The valid Manager is logged on the system

**Flow of event:**

1. The Manager enters employee's ID
2. **Include (FindEmployeeDetails)**
3. The Manager selects the part of employee details and change them
4. ...

**Postconditions:**

### Use case: ViewEmployeeDetails

ID: UC2

**Actors:**

Manager

**Preconditions:**

1. The valid Manager is logged on the system

**Flow of event:**

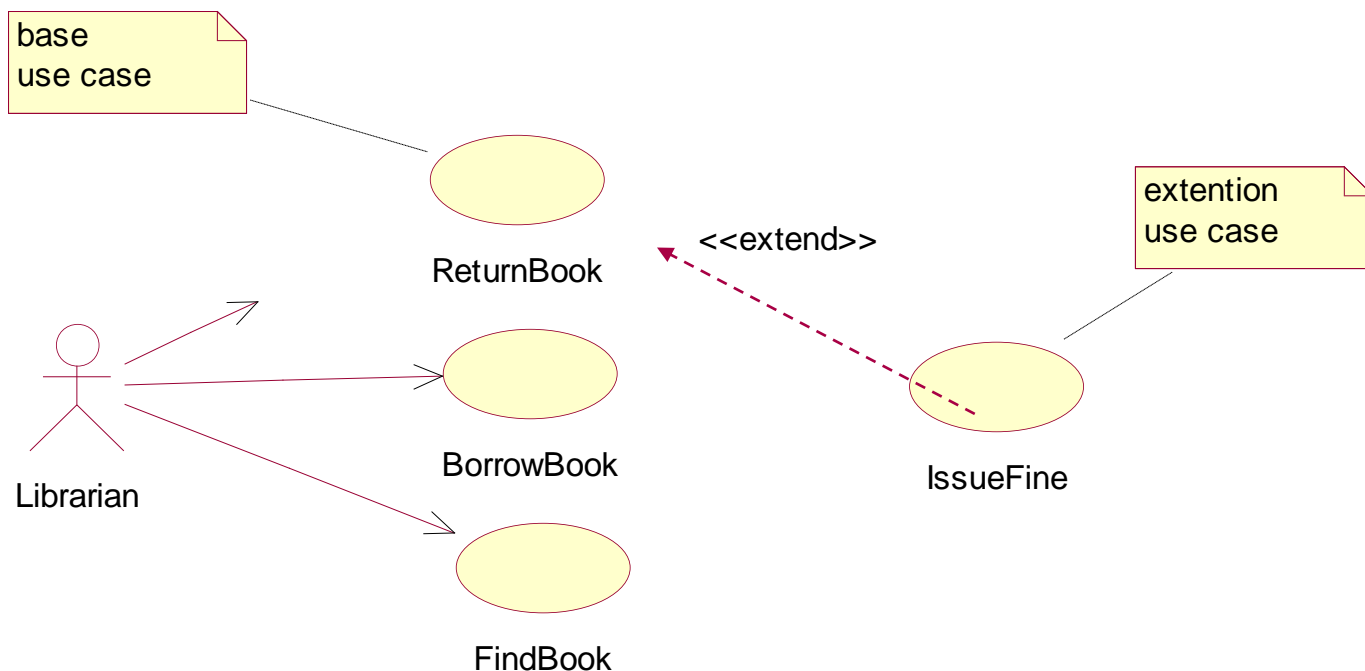
1. The Manager enters employee's ID
2. **Include (FindEmployeeDetails)**
3. The system displays the employee details
4. ...

**Postconditions:**

# 8. 유스케이스 명세

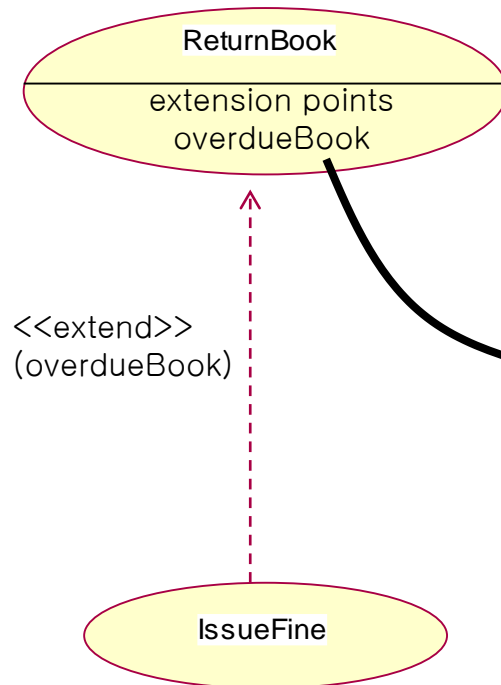
## <<extend>> 유스케이스의 예

- ❑ <<extend>>는 기존의 유스케이스의 행위를 확장 혹은 추가 시키는 용도로 사용된다
- ❑ 기본 (base)유스케이스는 확장 유스케이스가 포함될 지점을 의미하는 확장점을 제공해야 한다



# 8. 유스케이스 명세

## <<extend>> 유스케이스의 예



### Use case: ReturnBook

ID: UC9

**Actors:** Librarian

**Preconditions:**

1. The valid Librarian is logged on the system

**Flow of event:**

1. The Librarian enters the borrower's ID
2. The system displays the borrower's details including the list of borrowed books
3. The Librarian finds the book to be returned in the list of books

**<<overdueBook>>**

4. The Librarian returns the book

...

**Postconditions...**

### Extension Use case: IssueFine

ID: UC9

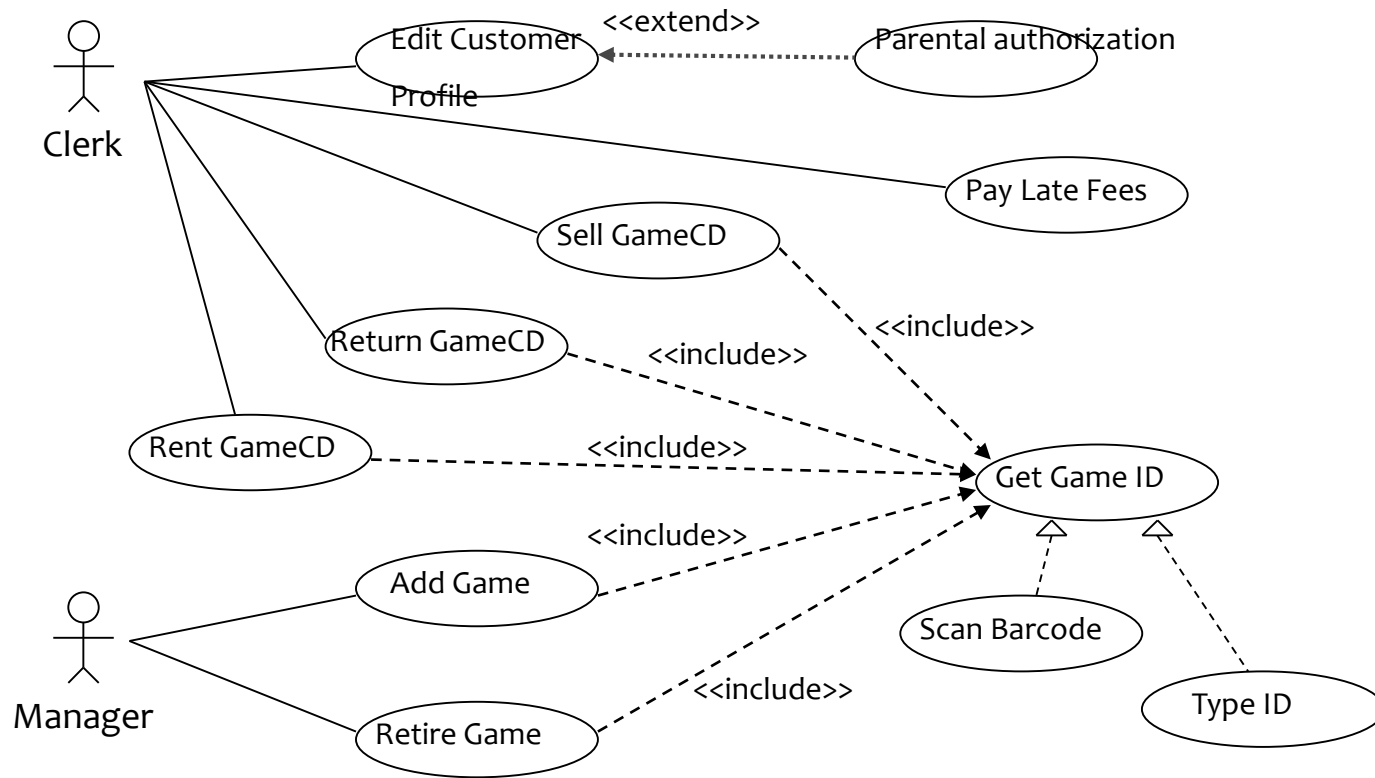
**Insertion segment**

1. The Librarian uses the system to record and print a fine



# 8. 유스케이스 명세

## 유스케이스 다이어그램의 예



# 9. ECP 예제

## Problem Statements

### E-Commerce Platform (ECP) System

#### Introduction

The E-Commerce Platform (ECP) is a web-based selling channel for CVT Limited. The goal of the ECP is to allow CVT customers to order products via the mobile phone from an online catalogue.

The ECP must integrate with the existing inventory and dispatch systems and must also communicate credit card information to the credit card processing company for validation before an order is accepted.

#### ECP function

**Books.** Books are categorized according to subject matter such as Art, Biographies, Children's books, Finance, Computers etc. Each book is identified by its ISBN. Customers can browse the book catalog by category or find a given book based on the following search criteria:

- Author, Title, Publisher, ISBN

#### CDs

CDs are categorized according to subject matter such as Children's music, Classical, Country, Dance, Folk etc. Customers can browse the CD catalog by category or find a given CD based on the following search criteria:

- Artist, Title, Label, Composer

#### Product Catalog

As the user interface prototype shows, we expect the ECP to offer the customer an initial choice of book or CD. On selecting either book or CD the ECP should then list the categories and allow the customer to choose a category or search for a specific product. The result of choosing a category or doing a search is the same - a summary list of products:

- For books this summary should contain at least author, title, publisher, ISBN, price.
- For CDs this summary should contain at least artist, title, label, composer, price.

Clicking on any product in the summary will bring up a full product description that includes all of the product information, the price and an optional picture. Next to the price there is an "Add to basket" button.

#### The shopping basket

When an item is added to the shopping basket, the customer is taken to the shopping basket screen that shows the list of all products currently in the basket. On this screen the customer may:

- Remove an item from the basket
- Change the quantity of an item
- Proceed to checkout

#### Checkout

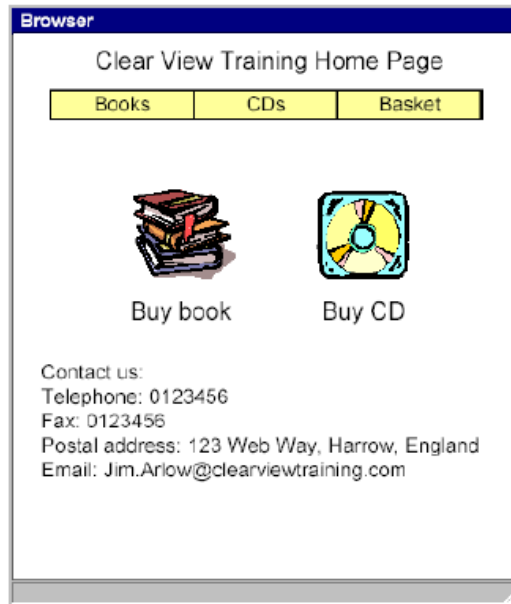
The system presents the customer with a summary of their order. If they click on "confirm" to confirm the order, then the system asks them to log in if they have not already done so. Ideally, the checkout should recognize the customer in which case the log in is automatic. If not, then existing customers must log in by entering a user name and password. New customers must fill out a form that asks for the following details:

- Name
- Address
- Email address
- Phone number
- Fax number
- Credit card details

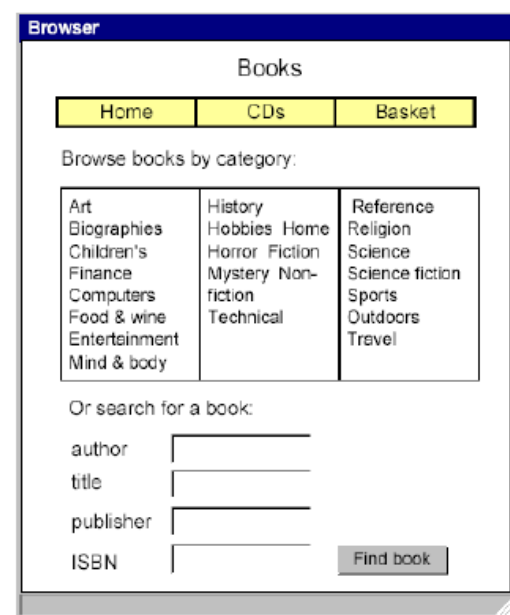
On submitting this form, the customer will be issued with a user name (which should probably be their email address) and is asked to select a password. Order processing then completes.

# 9. ECP 예제

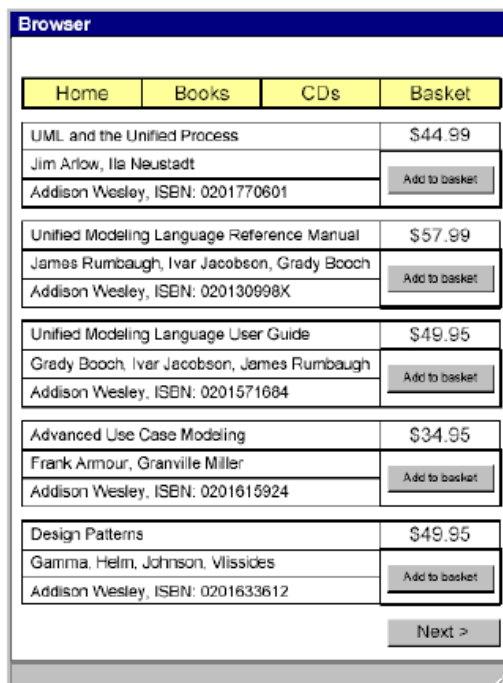
## 문제서술서



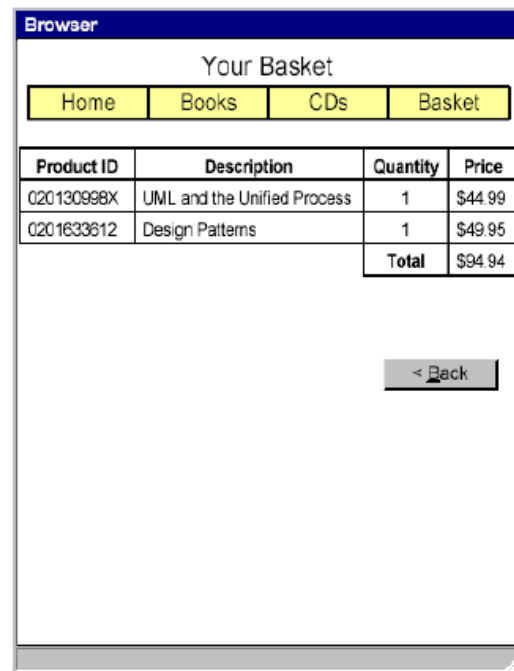
<Fig. 1> ECP Home Page



<Fig. 2> Search page



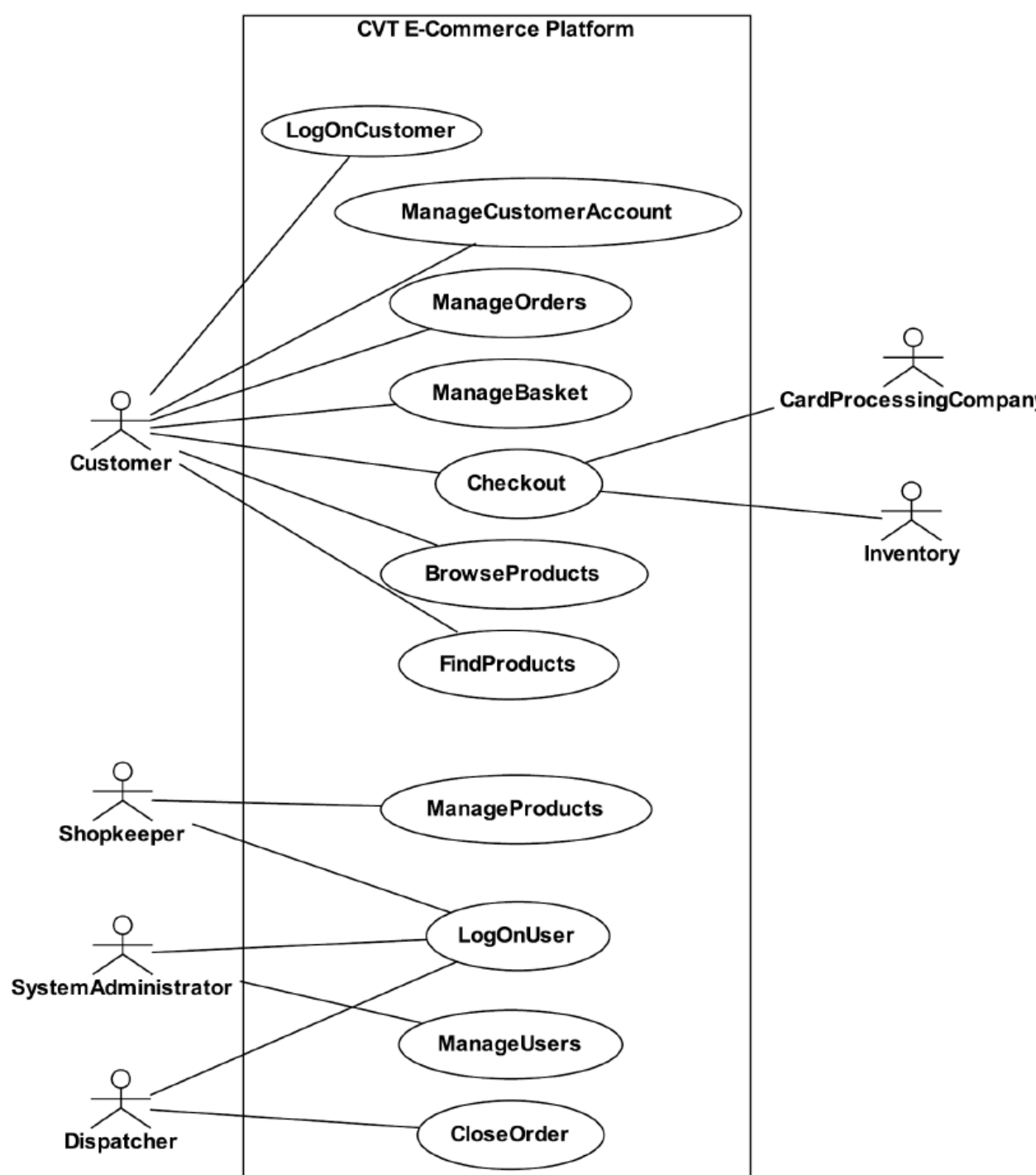
<Fig. 3> Browsing results



<Fig. 4> order page

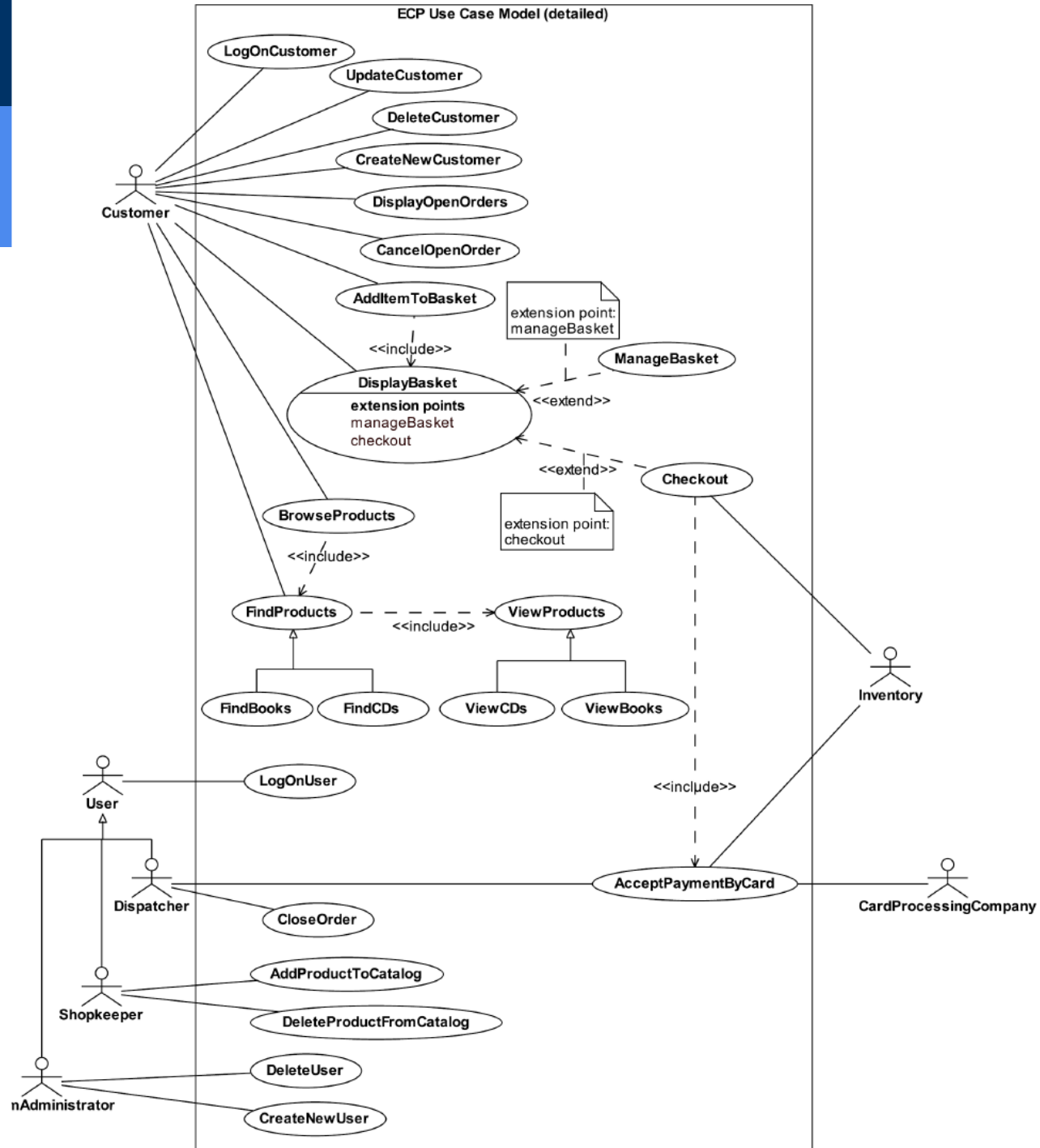
# 9. ECP 예제

Use case diagram  
version 1



# 9. ECP 예제

## Use case diagram version 2



# 9. ECP 예제

## Use case: Checkout

### Actors:

Customer  
Inventory

### Preconditions:

- 1 The Customer is logged on to the system.

### Flow of event:

- 1 The use case begins when the Customer selects “Checkout”.
- 2 The system asks the Inventory actor to provisionally reserve the items in the shopping basket.
- 3 For each item that is out of stock
  - 3.1 The system informs the Customer that the item is currently unavailable and it is removed from the order.
- 4 The system presents the final order to the Customer. The order includes an order line for each product that shows the product identifier, the product name, the quantity, the unit price, the total price for that quantity. The order also includes the shipping address and credit card details of the Customer and the total cost of the order including tax and postage and packing.
- 5 The system asks the Customer to accept or decline the order
- 6 The Customer accepts the order.
- 7 include(AcceptPaymentByCard)

### Postconditions:

- 1 The Customer has accepted the order.
- 2 The ordered items have been reserved by the Inventory actor.



**Use case:  
Accept  
Payment  
By Card**

**Preconditions:**

- 1 The Customer is logged on to the system.
- 2 Inventory items have been provisionally reserved for the customer.

**Primary scenario:**

- 1 The use case begins when the Customer accepts the order.
- 2 The system retrieves the Customer's credit card details.
- 3 The system sends a message to the CardProcessingCompany that includes: merchant identifier, merchant authentication, name on card, number of card, expiry date of card, amount of transaction.
- 4 The CardProcessingCompany authorises the transaction.
- 5 The system notifies the Customer that the card transaction has been accepted.
- 6 The system gives the Customer an order reference number for tracking the order.
- 7 The system tells the Inventory actor to reserve the required items.
- 8 The system sends the order to the Dispatcher.
- 9 The system changes the order's state to pending.
- 10 The system displays an order confirmation that the Customer may print out.

**Secondary scenarios:**

CreditLimitExceeded  
BadCard  
CreditCardPaymentSystemDown

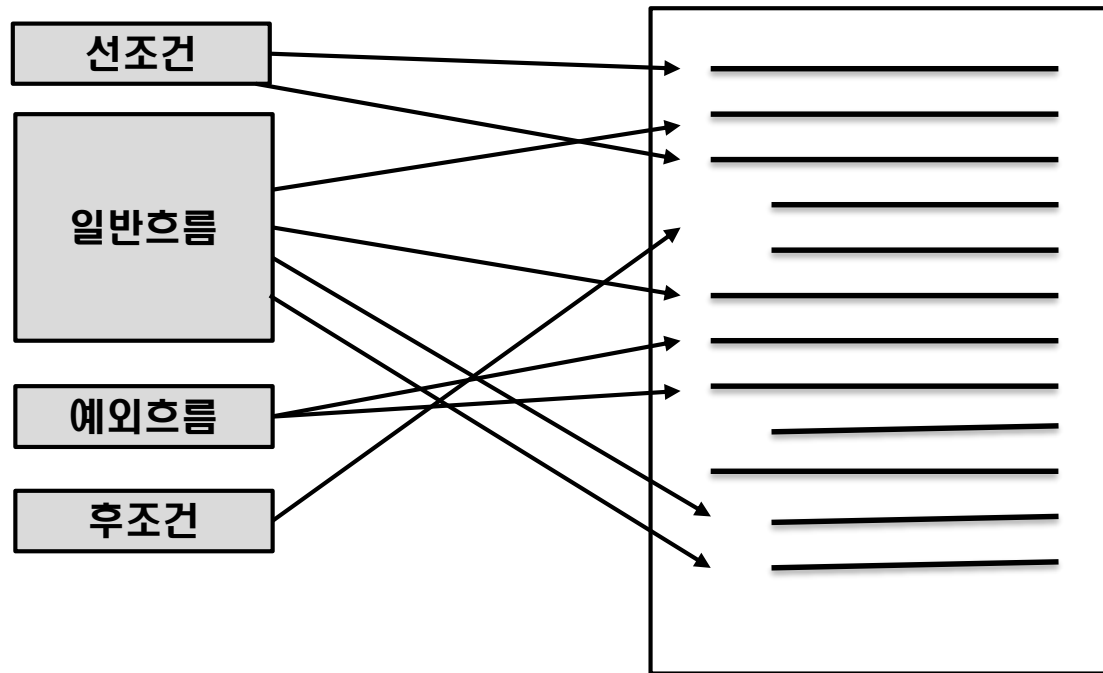
**Postconditions:**

- 1 The order status has been set to pending.
- 2 The Customer's credit card has been debited by the appropriate amount.
- 3 Inventory items have been reserved to cover the order.
- 4 The order has been sent to the Dispatcher.

# 소프트웨어 요구문서 작성 (발췌)

## Use case와 SRS

- ❑ Use case와 SRS는 구조와 목적이 다름
- ❑ 유스케이스로부터 나온 기능요구사항은 계층적으로 구조화되어 SRS 곳곳에 분산됨





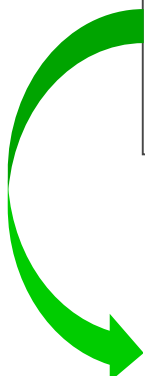
# 소프트웨어 요구문서 작성 (발췌)

## Use case 예: Checkout

Flow of event:

- 1 The use case begins when the Customer selects “Checkout”.
- 2 The system asks the Inventory actor to provisionally reserve the items in the shopping basket.
- 3 For each item that is out of stock
  - 3.1 The system informs the Customer that the item is currently unavailable and it is removed from the order.
- 4 The system presents the final order to the Customer. The order includes an order line for each product that shows the product identifier, the product name, the quantity, the unit price, the total price for that quantity. The order also includes the shipping address and credit card details of the Customer and the total cost of the order including tax and postage and packing.
- 5 The system asks the Customer to accept or decline the order
- 6 The Customer accepts the order.
- 7 include(AcceptPaymentByCard)

SRS:



R3	The ECP shall display detailed product descriptions consisting of name, photograph, price and description on demand.	<ul style="list-style-type: none"><li>• Products</li><li>• Functional</li></ul>	MustHave
	....		
R5	The ECP shall accept all major credit cards.	<ul style="list-style-type: none"><li>• Payment</li><li>• Functional</li></ul>	MustHave
R6	The ECP shall validate payment with the credit card processing company.	<ul style="list-style-type: none"><li>• Payment</li><li>• Functional</li></ul>	MustHave
R7	The ECP shall automatically calculate and add a shipping charge to the order.	<ul style="list-style-type: none"><li>• Payment</li><li>• Functional</li></ul>	Should-Have

# Workshop #3 유스케이스 모델링

- ❑ 각 팀별 case study에 대해 상세화된 버전의 유스케이스를 도출하여 다이어그램으로 표합니다.
- ❑ 워크샵 결과물
  - 결과물 : 1장의 유스케이스 다이어그램, 유스케이스 목록을 포함하는 엑셀파일
  - 수행시간: 1시간
  - 제출형식: pdf 혹은 이미지파일 (Star UML 사용 :<https://staruml.io/>)
- ❑ Star UML의 use case diagram:  
Model-> AddDiagram -> Use Case Diagram