

Section 3. UML

1. UML(Unified Modeling Language)

(1) UML 개념

- 프로그램 설계를 표현하기 위해 사용하는 표기법
- 시스템 개발 과정에서 이해관계자 사이에 의사소통을 원활하게 이루어지게 하기 위하여 표준화한 모델링 언어
- 소프트웨어 시스템, 업무 모델링, 시스템의 산출물을 규정하고 시각화, 문서화하는 언어
- 프로그램언어가 아닌 기호와 도식을 이용하여 표현하는 방법을 정의한다.

(2) UML 특징

- 가시화 언어 의사소통
 - 소프트웨어의 개념-모델을 시각적인 그래픽 형태로 작성한다.
- 명세화 언어
 - 분석, 설계, 구현 단계의 각 과정에서 필요한 모델을 명세화 할 수 있는 언어
- 구축 언어
 - 명세화된 설계모델은 다양한 언어의 소스코드로 변환하여 구축할 수 있다.
- 문서화 언어
 - 일련의 과정을 문서로 남겨 계속 유지 보수한다.

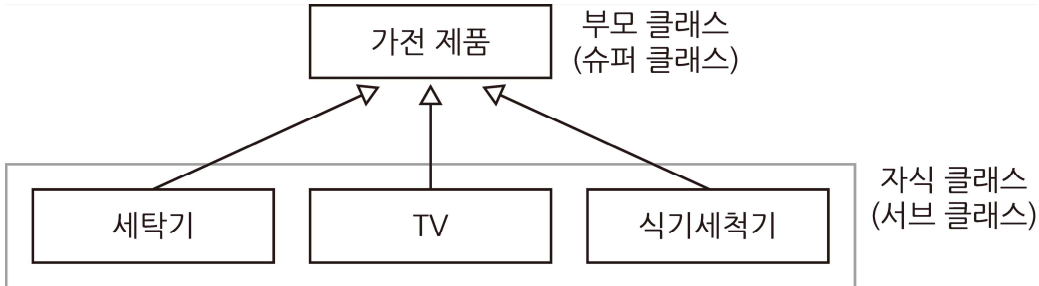
2. UML 구성요소

(1) 사물(Things)

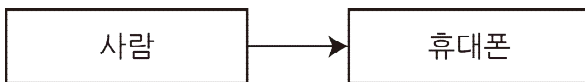
- 구조사물
 - 시스템의 개념적, 물리적 요소
 - 예) 클래스, 유즈케이스, 컴포넌트 등
- 행동사물
 - 시간과 공간에 따른 요소들의 행위
 - 예) 상호작용, 상태머신
- 그룹사물
 - 요소들을 그룹으로 묶은 것
 - 예) 패키지
- 주해사물
 - 부가적 설명이나 제약조건
 - 예) 주석, 노트

★ (2) 관계(Relationships)

- 일반화 관계(Generalization) **상속**
 - 한 클래스가 다른 클래스를 포함하는 상위 개념일 때의 관계
 - 객체지향 개념에서는 일반화 관계를 상속관계(Inheritance)라고 함



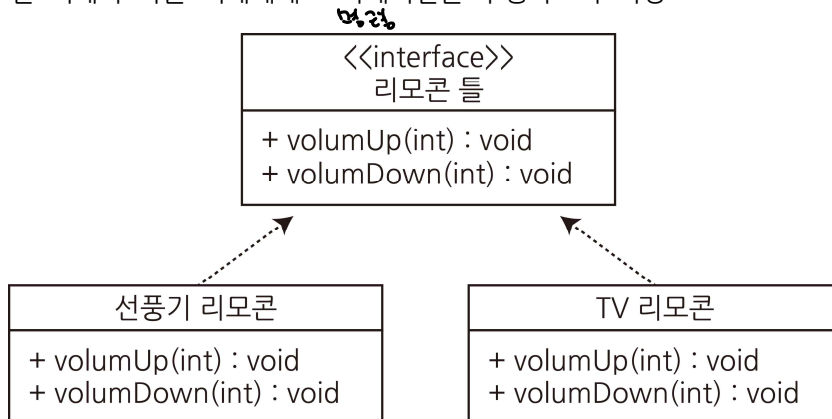
- 연관관계(Accociation)
 - 2개 이상 사물이 서로 관련된 관계
 - 한 클래스가 다른 클래스에서 제공하는 기능을 사용할 때 표시



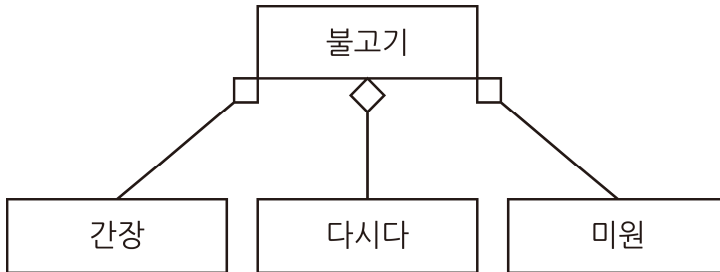
- 의존관계(Dependency)
 - 연관 관계와 같이 한 클래스가 다른 클래스에서 제공하는 기능을 사용할 때 표시
 - 연관 관계와 차이점은 두 클래스의 관계가 한 메서드를 실행하는 동안과 같이 매우 짧은 시간만 유지
 - 한 클래스의 명세가 바뀌면 다른 클래스에 영향을 줌
 - 한 클래스가 다른 클래스를 오퍼레이션의 매개변수로 사용하는 경우



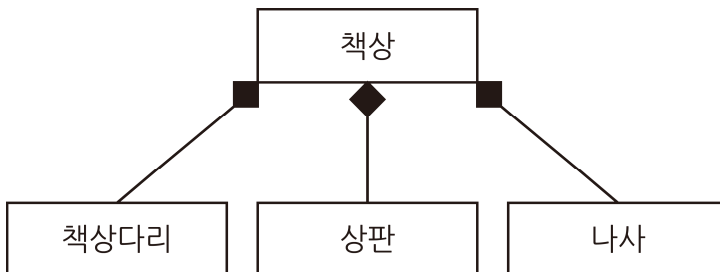
- 실체화 관계 (Realization) **계정**
 - 인터페이스를 구현받아 추상 메서드를 오버라이딩 하는 것을 의미
 - 한 객체가 다른 객체에게 오퍼레이션을 수행하도록 지정



- 집합 관계 - 집약관계 (Aggregation)
 - 한 객체가 다른 객체를 소유하는 'has a' 관계
 - 전체 객체의 라이프타임과 부분 객체의 라이프타임은 독립적
 - 전체 객체가 사라진다 해도 부분 객체는 사라지지 않음




- 집합관계 - 합성관계 (Composition)
 - 부분 객체가 전체 객체에 속하는 관계로 긴밀한 필수적 관계
 - 전체 객체의 라이프타임과 부분 객체의 라이프 타임은 의존적
 - 전체 객체가 없어지면 부분 객체도 없어짐





(3) 다이어그램(Diagram)

• 구조 다이어그램

종류	설명
 클래스 다이어그램	- 클래스의 속성과 클래스 사이의 관계를 표현 - 시스템 구조 파악 및 구조상 문제점을 도출
객체 다이어그램	- 클래스에 속한 객체(인스턴스)를 특정 시점의 객체와 객체 사이 관계로 표현
컴포넌트 다이어그램	- 컴포넌트 사이 관계나 인터페이스를 표현
배치 다이어그램	- 결과물, 프로세스, 컴포넌트 등 물리적 요소들의 <u>위치</u> 를 표현 - 노드와 통신 경로를 표현
복합체 다이어그램	- 클래스나 컴포넌트가 복합구조를 가질 시 그 내부 구조를 표현
패키지 다이어그램	- 유즈케이스나 클래스 등 모델 요소들을 그룹화한 패키지들의 관계 표현

• 행위 다이어그램

종류	설명
 유즈케이스 다이어그램	- 유저의 요구를 분석하며 <u>기능 모델링</u> 작업에 사용 - 시스템의 기능을 나타내기 위해 사용자의 요구를 추출하고 분석하는 데 사용 - 외부에서 보는 시스템의 동작으로, 객체들이 어떻게 상호작용하는지 모델링 - 구성요소 : Actor, Use Case, System
 시퀀스 다이어그램	- 특정 행동이 <u>어떠한 순서</u> 로 어떤 객체와 <u>상호작용</u> 하는지 표현 - 현재 시스템이 어떠한 시나리오로 움직이고 있는지를 나타냄 - 구성요소 : 활성객체, 메시지, 생명선, 제어사각형 - 메시지유형 : 동기 메시지, 비동기 메시지, 반환 메시지, 자체 메시지
커뮤니케이션 다이어그램	- 동작에 참여한 객체들이 주고받는 메시지와 객체 간 연관까지 표현
상태 다이어그램	- 객체가 자신이 속한 <u>클래스의 상태</u> 변화 및 다른 <u>객체 간 상호작용</u> 에 따라 상태 변화 표현
활동 다이어그램	- 시스템이 어떤 기능을 수행하는지에 따라 객체 처리 로직이나 조건에 따른 처리 <u>흐름을 순서에 따라 표현</u>
상호작용 다이어그램	- 상호작용 다이어그램 간 제어 흐름 표현
타이밍 다이어그램	- 객체 상태 변화와 시간 제약 명시적 표현

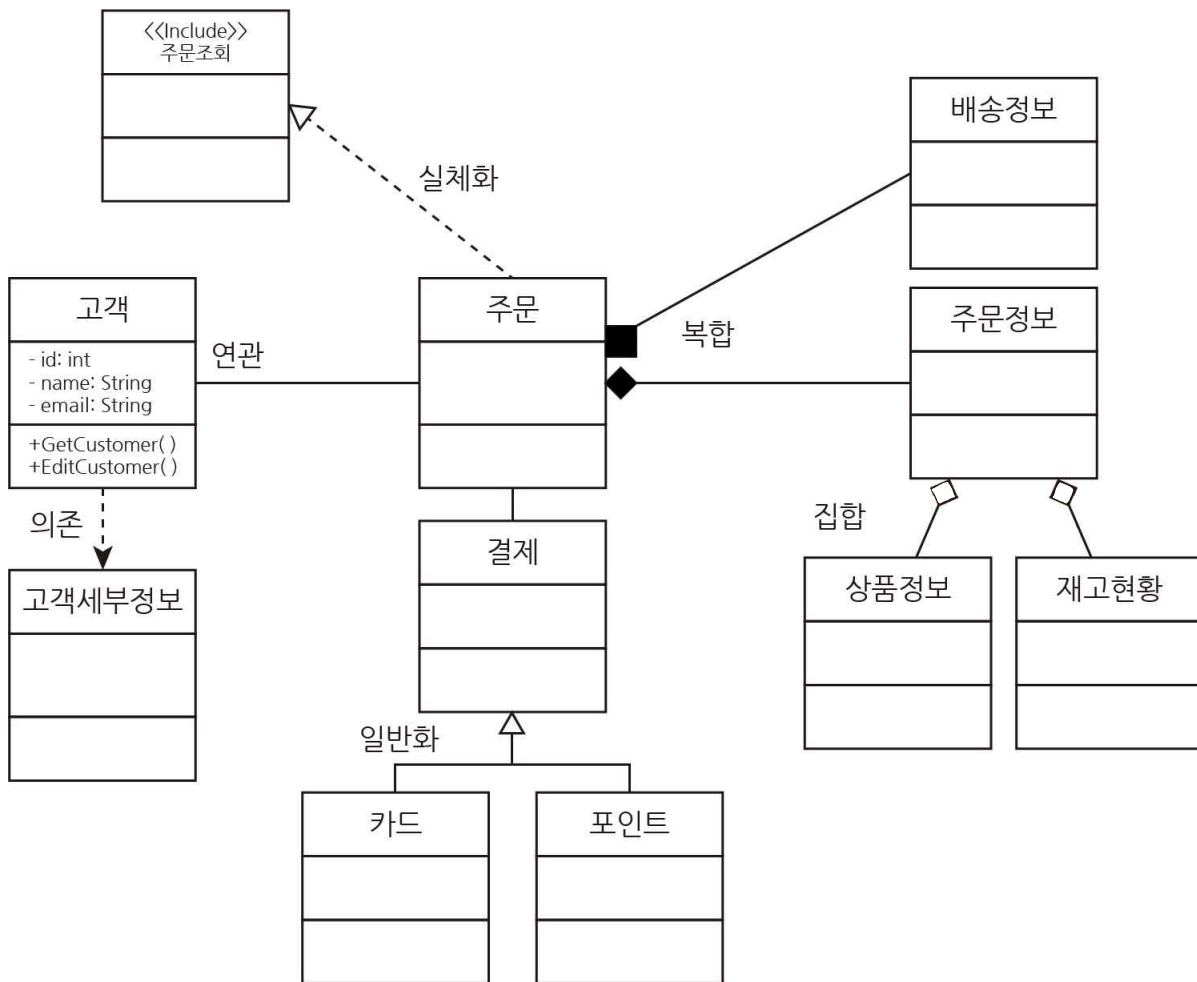
3. 주요 다이어그램

(1) 클래스 다이어그램 구조

① 클래스 다이어그램 개념

- 자기만의 속성(attribute)과 일정한 행동(behavior)으로 구성
- 여러 개의 클래스들은 서로 연관이나 상속, 의존 관계 등으로 서로 간의 상호작용을 표현

② 클래스 다이어그램 표현



③ 접근 제한자 표기법

표기법	접근 제한자	사용 범위
-	private	해당 클래스 내에서만 접근 가능
#	protected	상속, 동일 패키지 내에서만 접근 가능
+	public	어디서든 접근 가능

(2) 유스케이스 다이어그램

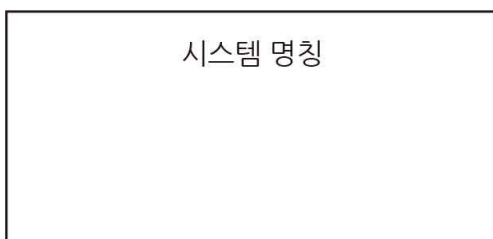
① 유스케이스 다이어그램 개념

- 시스템과 사용자의 상호작용을 다이어그램으로 표현
- 사용자의 관점에서 시스템의 서비스 혹은 기능 및 그와 관련한 외부 요소를 보여준다.
- 프로젝트에 대한 요구사항을 정의하고 세부기능을 분석하며 개발 범위를 정할 때 작성한다.

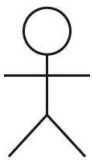
② 유스케이스 다이어그램 구성요소

- 시스템(System)
 - 만들고자 하는 프로그램 명칭

〈시스템의 표현방법〉



- 액터(Actor)
 - 시스템의 외부에 있고 시스템과 상호작용을 하는 사람 시스템을 표현



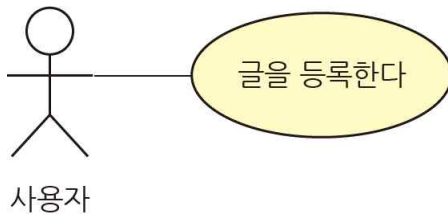
액터명

- 유스케이스(Usecase)
 - 사용자 입장에서 바라본 시스템의 기능

〈유스케이스의 표현방법〉

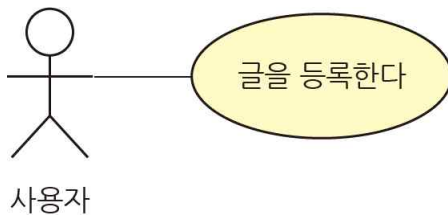


- 관계(Relation)
 - 액터와 유스케이스 사이의 의미있는 관계



③ 유스케이스 다이어그램

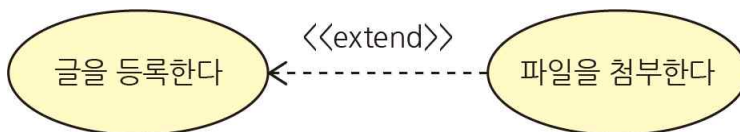
- 연관관계(Association)
 - 유스케이스와 액터간의 상호작용이 있음을 표현
 - 유스케이스와 액터를 실선으로 연결



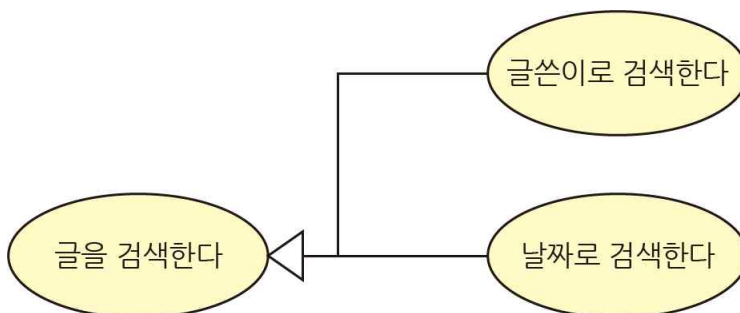
- 포함 관계(Include)
 - 유스케이스를 수행 할 때 반드시 실행되어야 하는 경우



- 확장 관계(Extend)
 - 유스케이스를 수행 할 때 특정 조건에 따라 확장 기능 유스케이스를 수행하는 경우



- 일반화 관계(Generalization)
 - 유사한 유스케이스 또는 액터를 모아 추상화한 유스케이스



(3) 시퀀스 다이어그램

① 시퀀스 다이어그램 개념

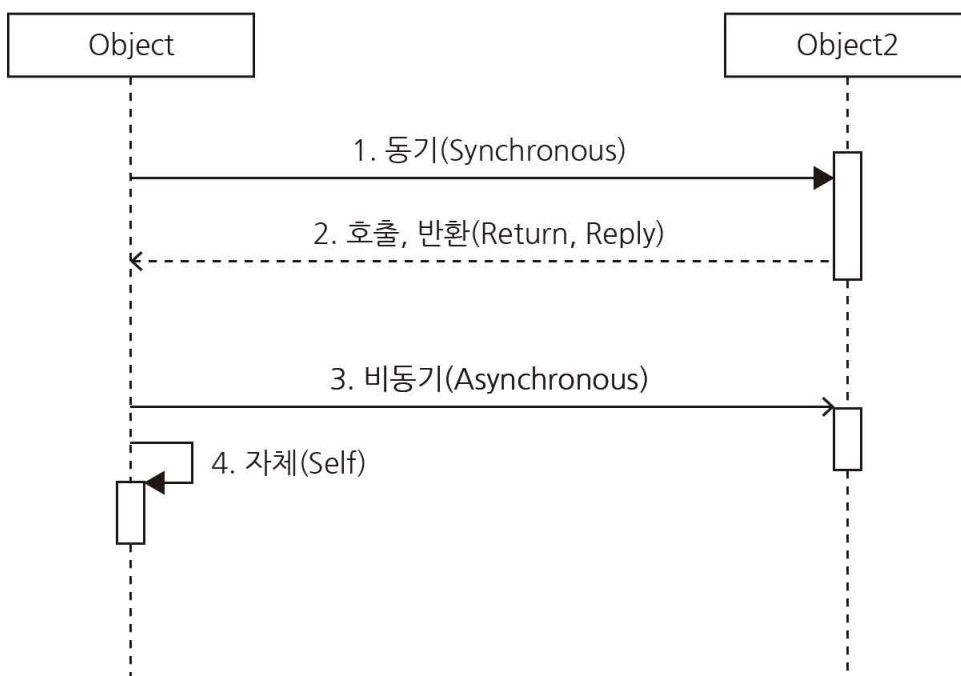
- 객체간의 상호작용 메시지 시퀀스를 시간의 흐름에 따라 나타내는 다이어그램

② 시퀀스 다이어그램 구성요소

- 객체(Object)와 생명선(Lifeline)
 - 객체(활동 주체)는 직사각형으로 표현
 - 라이프라인은 객체에서 이어지는 점선으로 표현
 - 점선은 위에서 아래로 갈수록 시간의 경과를 의미
- 활성화 박스(Activation Box)
 - 생명선상에서 길다란 직사각형으로 표현
 - 현재 객체가 어떤 활동을 하고 있음을 의미
- 메시지(Message)
 - 인스턴스 간 주고 받은 데이터
 - 메시지의 유형

유형	설명
동기 메시지 (Sync message) TCP	요청을 보낸 후에 반환이 올 때까지 대기
비동기 메시지 (Async message) UPP	요청을 보낸 다음 반환을 기다리지 않고 다른 작업을 수행
자체 메시지 (Self message)	자기 자신에게 요청을 보냄
반환 메시지 (Reply/Return message)	요청에 대해 메시지를 반환

- 메시지의 표현



(4) 상태 다이어그램

① 상태 다이어그램 개념

- 한 객체의 상태 변화를 나타내는 다이어그램

② 상태 다이어그램 예시

