

■ 주요 내용

- 01 UML 구성 요소
- 02 UML 뷰
- 03 UML 특성

■ 학습목표

- UML의 구성 요소와 관계를 이해한다.
- UML의 뷰의 개념과 종류를 이해한다.
- UML의 특성을 이해한다.

01 UML 구성 요소

■ UML 구성요소

- 사물 Things, 관계 Relationship, 다이어그램 Diagram의 세 가지 구성 요소로 이루어짐

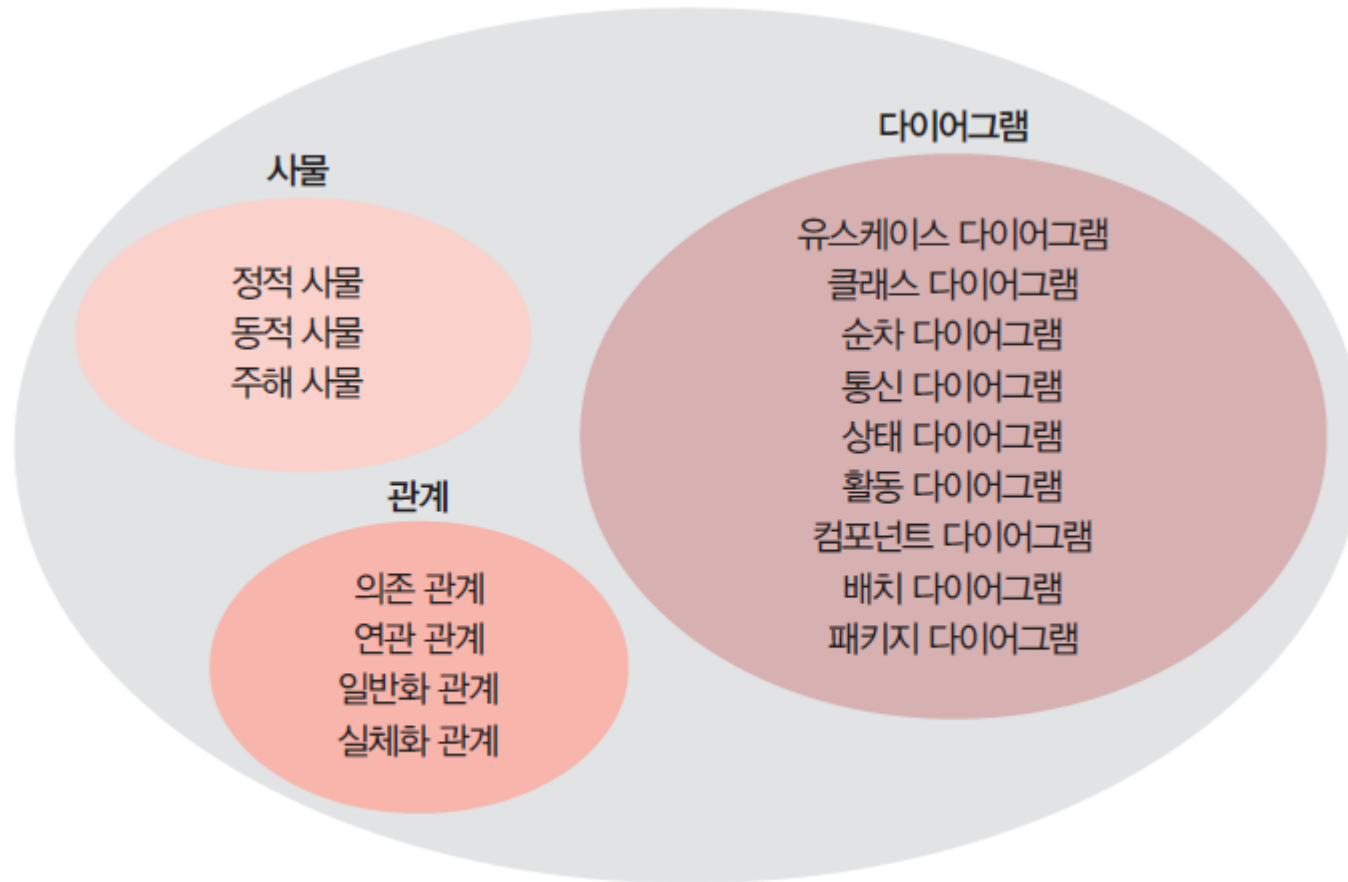


그림 2-1 UML 구성 요소

01 UML 구성 요소

■ 사물

■ 정적사물 Structural Things

- 모델의 구조, 즉 개념적·물리적 요소를 표현하는 명사
- 클래스, 인터페이스, 통신, 컴포넌트, 패키지, 노드 등이 있다.

• 클래스 Class

- 동일한 속성, 오퍼레이션, 관계, 의미를 공유하는 객체를 기술한 것
- 직사각형 표시

• 인터페이스 Interface

- 클래스 또는 컴포넌트의 서비스를 명세화하는 오퍼레이션을 모아놓은 것
- 특정 클래스나 컴포넌트의 전체 또는 일부 동작을 나타낼 수 있음
- 원 표시

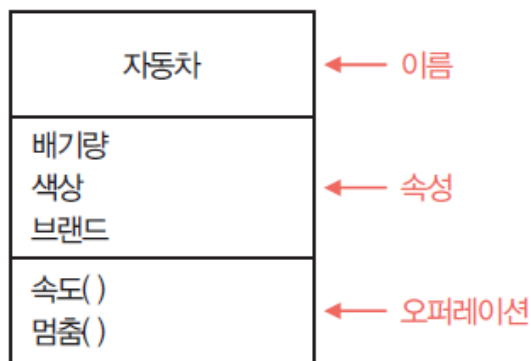
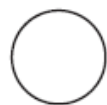


그림 2-2 클래스의 예



자동차

그림 2-3 인터페이스의 예



■ 사물

■ 정적사물Structural Things

- 통신Communication
 - 서로 다른 요소와 역할이 모여 교류Interaction를 정의
 - 동작과 구조에서 중요하며 클래스 하나가 다수의 통신에 참여할 수 있음
 - 실선 사각형 표시
- 컴포넌트Component
 - 전체 시스템을 구성하는 단위
 - 독립적으로 개발되고 배포되며 조립, 교환이 가능한 응집도가 높은 소프트웨어 산출물
 - 탭이 달린 직사각형 표시

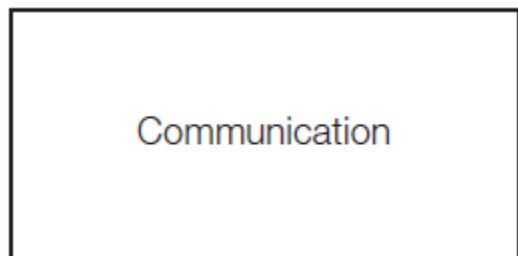


그림 2-4 통신의 예

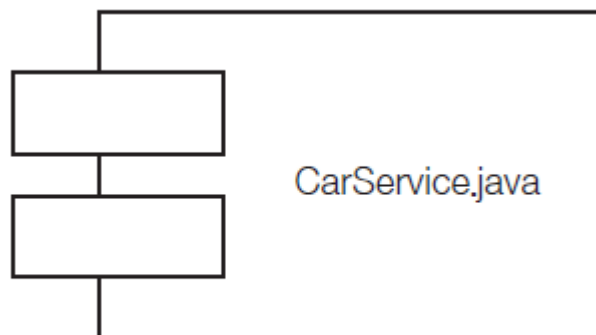


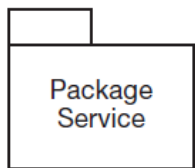
그림 2-5 컴포넌트의 예

01 UML 구성 요소

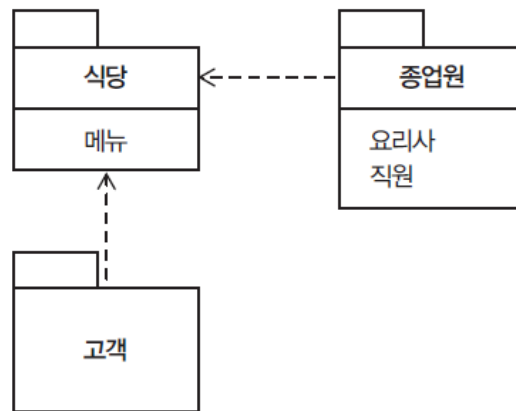
■ 사물

■ 정적사물 Structural Things

- 패키지 Package
 - 요소를 그룹으로 묶음, 정적 사물이나 동적 사물도 하나에 들어갈 수 있음
 - 컴포넌트가 물리적인 데 반해 패키지는 개념적
 - 탭이 달린 폴더 형태 표시
- 노드 Node
 - 실행할 때 존재하는 물리적 요소
 - 전산 자원(기계)을 나타내고 약간의 메모리와 처리 능력을 가짐
 - 컴포넌트가 노드에 존재하며, 노드에서 노드로 이동할 수 있음
 - 육면체 표시



(a) 패키지의 예



(b) 패키지 다이어그램의 예

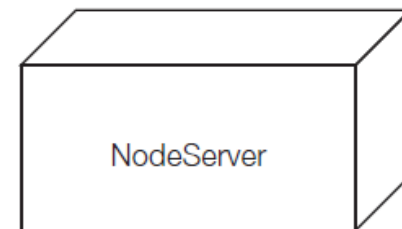


그림 2-7 노드의 예

그림 2-6 패키지 형태와 예

■ 사물

■ 동적사물 Behavioral Thing

- 주로 모델의 동적인 부분을 동사로 표시
- 교류, 유스케이스, 상태 머신 등

• 교류 Interaction

- 목적을 달성하기 위해 특정 문맥에 속한 객체들 간에 주고받는 메시지로 구성된 동작
- 직선 표시

• 유스케이스

- 시스템이 수행하는 활동들을 순차적으로 기술
- 액터 Actor에게 의미 있는 결과 값을 제공
- 실선 타원 표시

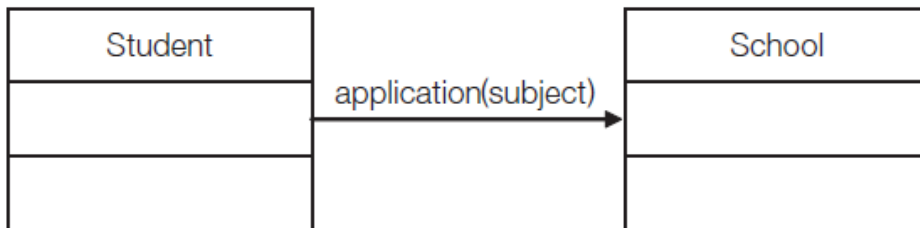


그림 2-8 메시지



그림 2-9 유스케이스

■ 사물

■ 동적사물 Behavioral Thing

- 상태 머신 state machine
 - 외부 이벤트에 대한 객체의 상태와 상태의 변화 순서를 기술
 - 서로 다른 요소들이 들어있음
 - » 상태전이 : 상태에서 다른 상태로의 흐름
 - » 사건 : 전이를 유발하는 것
 - » 활동 : 전이에 따른 응답
 - 모서리가 둥근 직사각형 표현

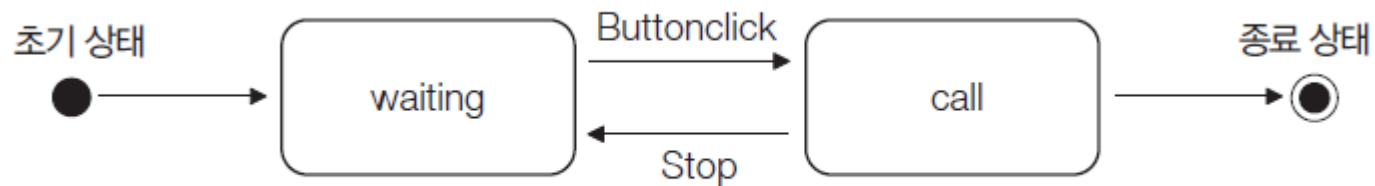


그림 2-10 상태 머신

■ 사물

■ 주해 사물

- 모델링에 참여하지는 않음
- 모델링에 필요한 모든 정보를 표시하기 위해 사용
- 노트가 있음

• 노트 Note

- 첨부되는 주석 또는 제약을 기술
- 모서리가 접힌 직사각형 표현

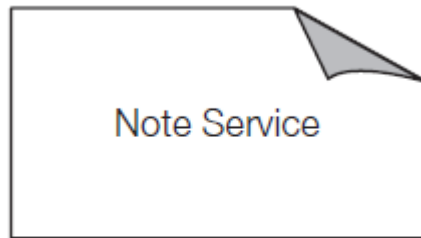


그림 2-11 노트

■ 관계

■ 의존 Dependency 관계

- 두 사물 간의 의미적 관계
- 한 사물의 명세가 바뀌면 다른 사물에 영향을 끼침
- 반드시 반대가 성립하진 않음
- 의존하는 사물을 향하는 점선 표현



그림 2-12 의존 관계

■ 관계

■ 연관 Association 관계

- 객체 사이의 연결 관계
- 지속적으로 유지되는 관계
- 한쪽 객체에서 다른 객체로 옮겨갈 수 있음
- 실선으로 표기
- 이름
 - 연관 관계의 의미설명
 - 원하는 방향으로 방향 삼각형을 표기
- 역할
 - 클래스 옆에 원하는 역할을 써서 연관 관계에서의 역할 표시

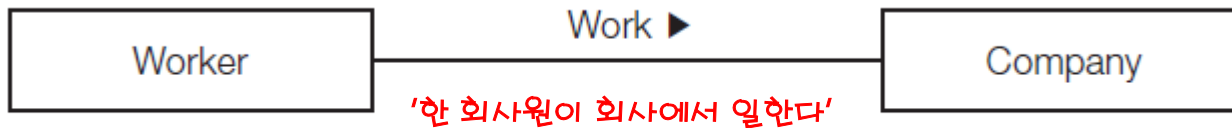


그림 2-13 이름

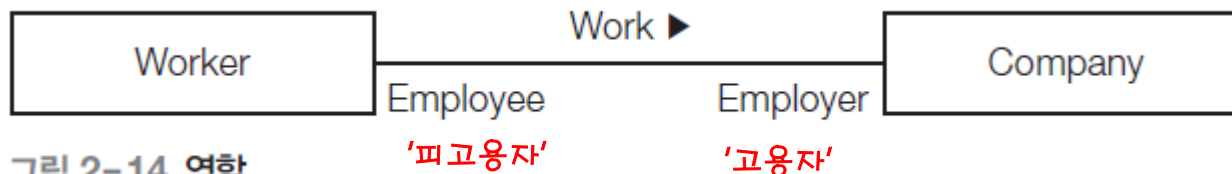


그림 2-14 역할

■ 관계

■ 연관 Association 관계

• 다중성

- 객체 하나에 몇 개의 객체가 연결되어 있는지를 밝히는 것
- 범위 값으로 나타내는 표현식 (1..*)이나 명시적인 값으로 표현
- 하나(1), 제로(0) 혹은 하나(0..1), 다수(0..*), 하나 이상 (1..*) 등으로 표현

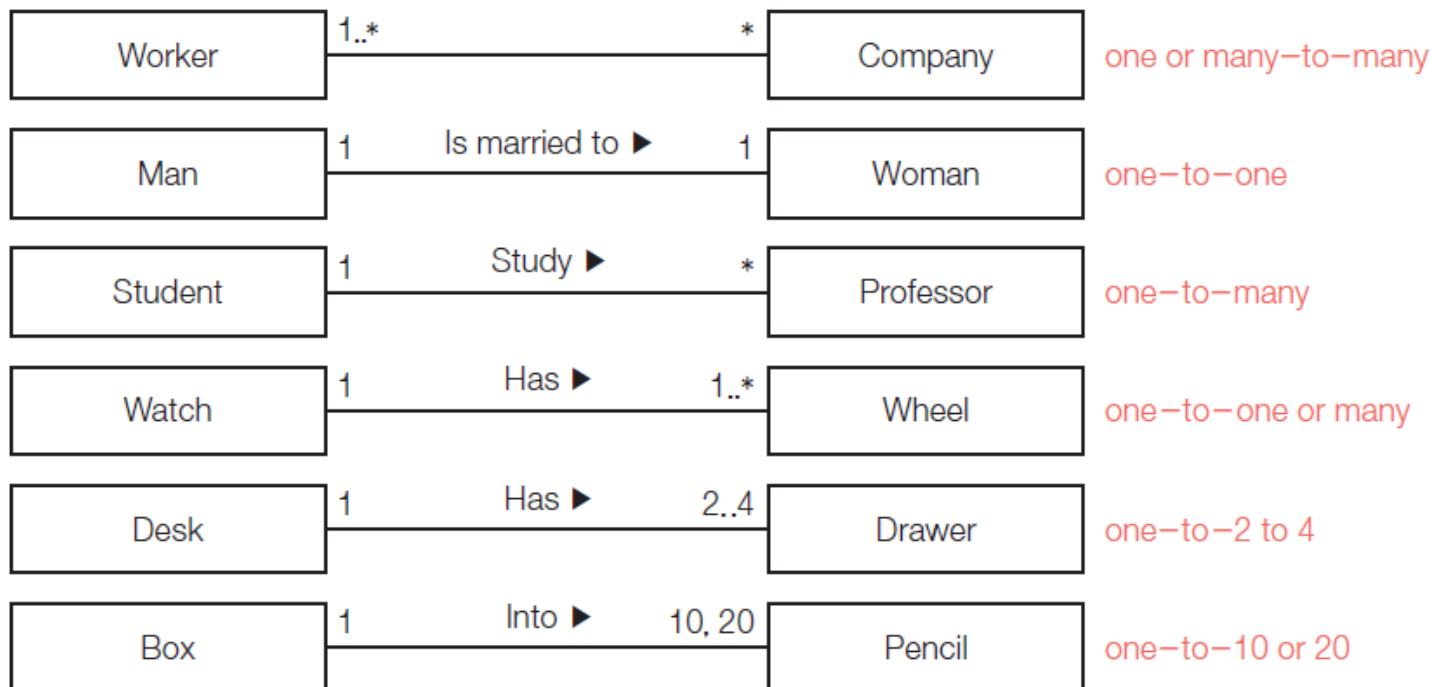


그림 2-15 다중성 관계

■ 관계

■ 연관 Association 관계

• 집합 연관

- 전체 쪽 객체 하나가 부분 쪽 객체들을 소유
- has-a 관계라고도 함

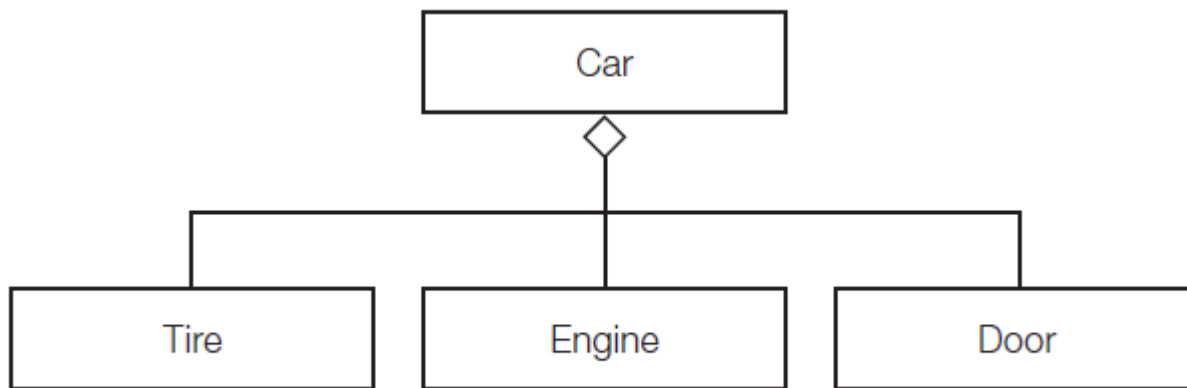


그림 2-16 집합 연관 관계

■ 관계

■ 일반화(Generalization) 관계

- 일반화된 사물과 좀 더 특수화된 사물 사이의 관계
- 자식 객체는 부모 객체의 속성과 오퍼레이션을 상속함
- 부모에게 없는 속성과 오퍼레이션을 가지기도 함
- 자식 객체는 부모 객체를 대신 할 수 있으나 그 반대는 불가

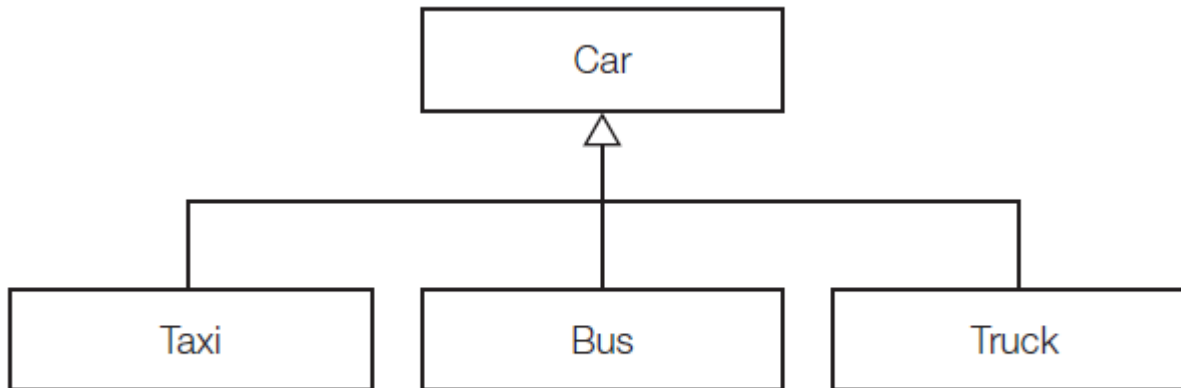


그림 2-17 일반화 관계

'자동차의 종류에는 택시,버스,트럭이 있다.'

■ 관계

■ 실체화Realization 관계

- 한 객체가 다른 객체에게 오퍼레이션을 수행하도록 지정하는 의미적 관계
- 인터페이스와 인터페이스에 오퍼레이션이나 서비스를 제공하는 클래스나 컴포넌트 사이의 관계를 지정하기 위해 사용
- 클래스는 2개 이상의 인터페이스를 실체화할 수 있고, 인터페이스는 2개 이상의 클래스로부터 실체화될 수 있음



그림 2-18 실체화 관계 예 : 텔레비전과 리모콘

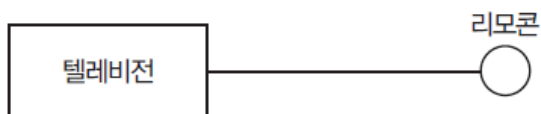


그림 2-19 실체화의 간단한 표현



그림 2-20 실체화와 의존

■ 다이어그램

■ 클래스 다이어그램

- 클래스, 인터페이스, 통신과 함께 이들의 관계를 나타냄

■ 컴포넌트 다이어그램

- 컴포넌트 사이의 구성과 의존을 표현

■ 배치 다이어그램

- 실행 시 처리하는 노드와 그 노드에 있는 컴포넌트들의 구성을 표현

■ 패키지 다이어그램

- 여러 모델 요소를 그룹화하여 패키지를 구성하고, 이들 패키지 사이를 관계로 표현

■ 유스케이스 다이어그램

- 유스케이스와 액터의 관계를 구조적으로 표현

■ 순차 다이어그램과 통신 다이어그램

- 교류 다이어그램의 한 종류

■ 활동 다이어그램

- 시스템 내부에 있는 활동의 흐름을 표현한 것

■ 상태 다이어그램

- 시스템의 동적부를 나타냄

■ 다이어그램

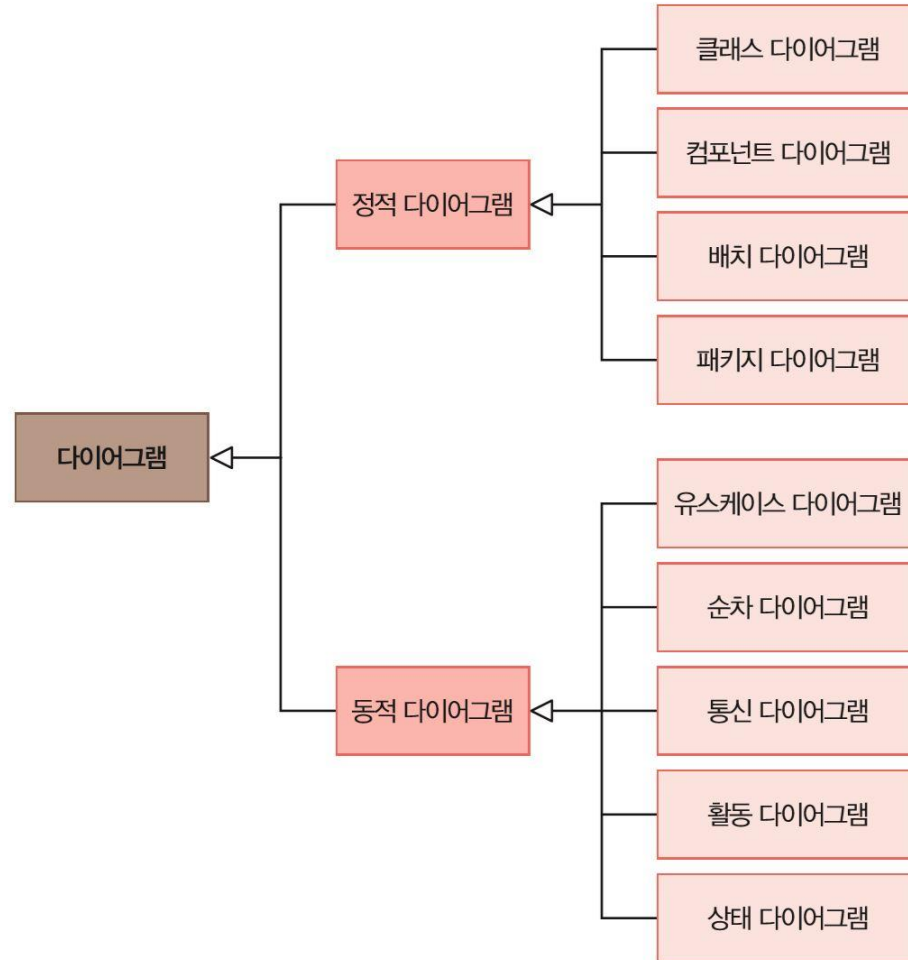


그림 2-21 정적 다이어그램과 동적 다이어그램

■ UML뷰의 개념과 종류

■ 유스케이스 뷰 (요구사항 뷰)

- 외부 액터에 의해 인식되는 시스템의 기능 요구 사항을 보여주는 관점
- 사용자가 시스템으로부터 원하는 기능이 '무엇'인지를 정의
- 다른 뷰를 유도하는 중심 역할
- 시스템을 하나의 블랙박스 Black Box로 바라봄

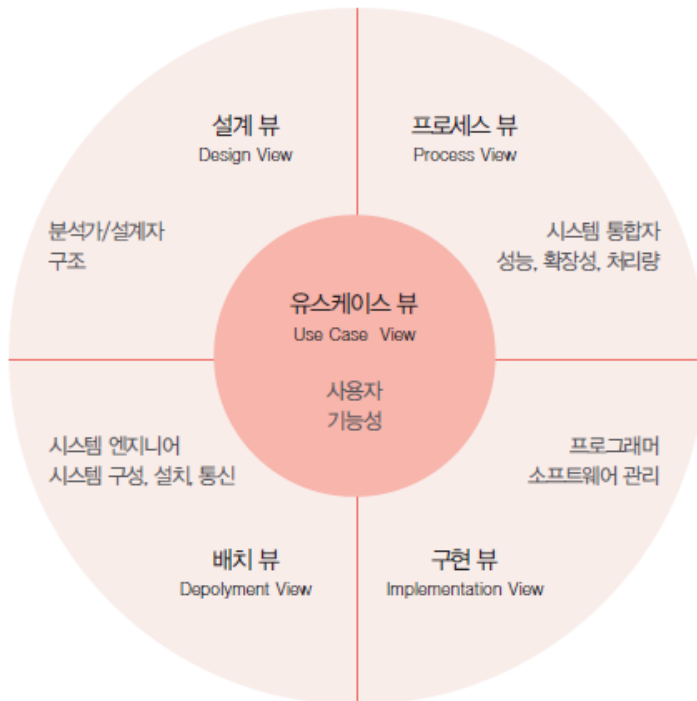


그림 2-22 소프트웨어 아키텍처의 5개 뷰

표 2-1 유스케이스 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	유스케이스 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램 통신 다이어그램, 활동 다이어그램

■ UML뷰의 개념과 종류

■ 설계 뷰

- 시스템 내부의 클래스와 컴포넌트를 파악해 기술

표 2-2 설계 뷰에 이용되는 UML 다이어그램

구분	관심 사항	이용되는 UML 다이어그램
정적 측면	클래스 및 클래스 사이의 관계	클래스 다이어그램, 객체 다이어그램
동적 측면	클래스 내의 동작	상태 다이어그램
	클래스 간의 상호작용	순차 다이어그램, 통신 다이어그램
	클래스의 연산 동작	활동 다이어그램

■ 프로세스 뷰

- 설계 뷰와 마찬가지로 시스템 내부의 구조에 중점을 두고 기술
- 그러나 독자적인 제어 스레드를 가질 수 있는 클래스를 중점으로 함

표 2-3 프로세스 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	클래스 다이어그램, 활성 클래스 다이어그램
동적 측면	상태 다이어그램, 통신 다이어그램, 활동 다이어그램

■ UML뷰의 개념과 종류

■ 구현 뷰

- 시스템 구현 형태를 나타내기 위해 구현 모듈과 그들의 관계 각종 파일의 의존 관계 등을 보여줌
- 컴포넌트 다이어그램으로 표현

표 2-4 구현 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	컴포넌트 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램, 통신 다이어그램, 활동 다이어그램

■ 배치 뷰

- 컴퓨터와 컴퓨터 간의 통신 방법에 중점을 둠
- 배치 다이어그램으로 표현

표 2-5 배치 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	배치 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램, 통신 다이어그램, 활동 다이어그램

■ 개발 활동과 UML뷰

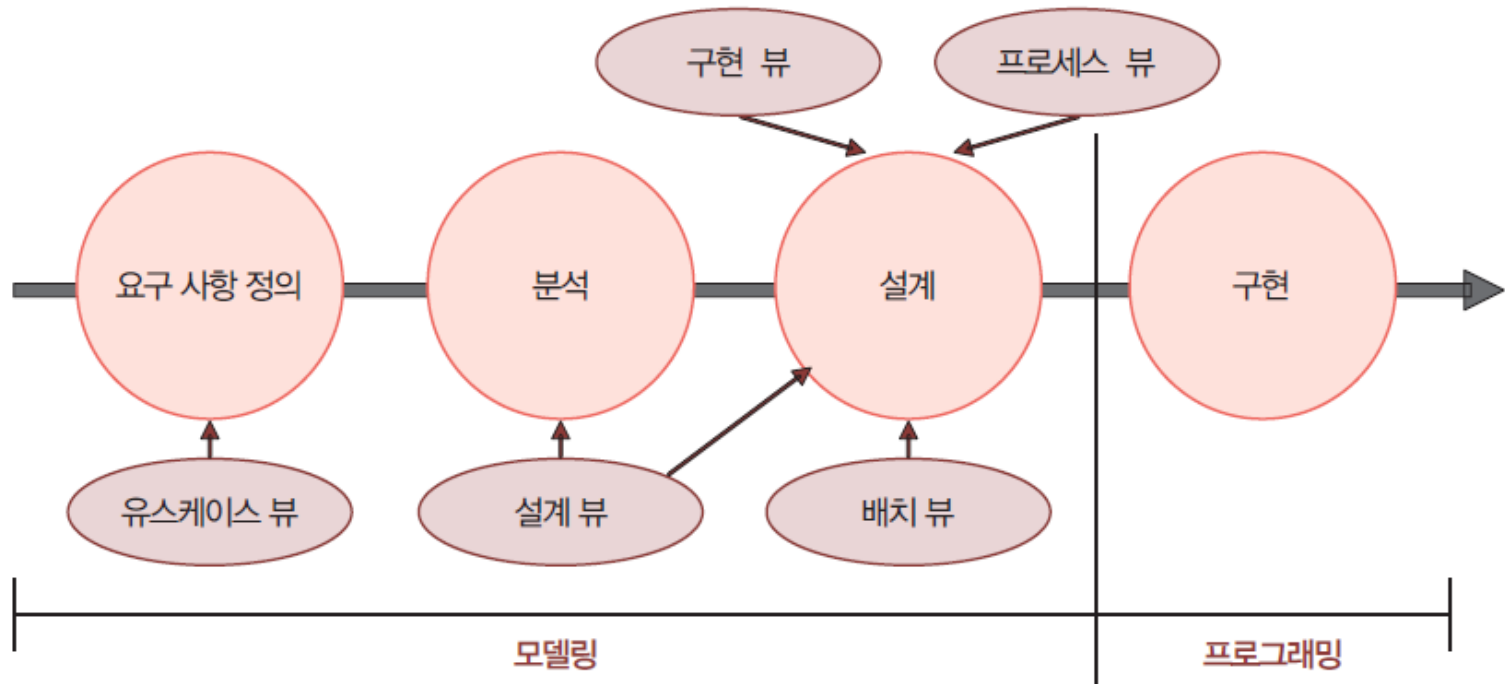


그림 2-23 개발 활동과 UML 뷰

■ 개발 활동과 UML뷰

표 2-6 시스템 유형별 뷰와 UML 다이어그램

뷰	UML 다이어그램	시스템 유형		
		간단한 시스템	반응적 시스템	분산 시스템
유스케이스 뷰	유스케이스 다이어그램	✓	✓	✓
설계 뷰	클래스 다이어그램	✓	✓	✓
	순차 다이어그램	✓	✓	✓
	통신 다이어그램	✓	✓	✓
	상태 다이어그램		✓	✓
프로세스 뷰	클래스 다이어그램			✓
	순차 다이어그램			✓
	통신 다이어그램			✓
구현 뷰	컴포넌트 다이어그램			✓
배치 뷰	배치 다이어그램			✓

■ 명세서

- 클래스의 명세 표기법은 클래스 이름, 속성, 오퍼레이션 등을 표현하는 방법을 제공

■ 장식

- 중요 특징을 표현하기 위해 고유한 그래픽 표기

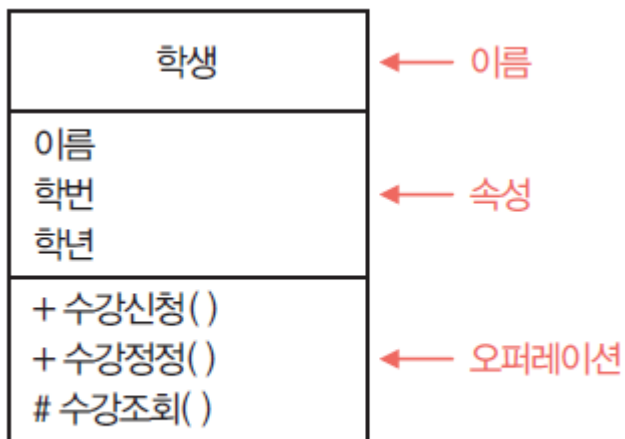
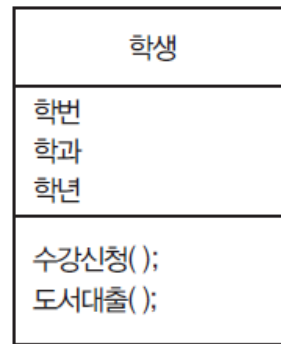


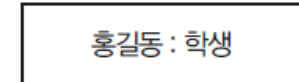
그림 2-24 장식

■ 공통 분할

- 클래스와 객체의 분할
- 추상개념을 구체적으로 명시



(a) 클래스



(b) 객체

그림 2-25 클래스와 객체의 공통 분할

- 인터페이스와 구현의 분할
- 계약과 계약의 구체적인 실현을 명시

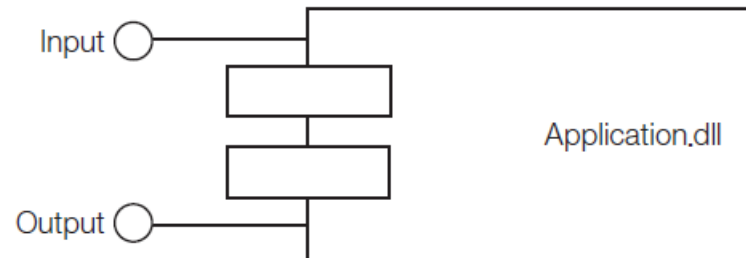


그림 2-26 인터페이스와 구현의 공통 분할

■ 확장

- 스테레오 타입 << >>
 - 기본 요소 외에 새로운 요소를 만들어내기 위한 확장
- 꼬리표 값 {tag=value}
 - UML 구성 요소의 속성을 확장
 - 구성 요소의 명세서에 새로운 정보를 생성을 도움
- 제약 {}
 - 이전 규칙을 수정하거나 새롭게 생성할 수 있음

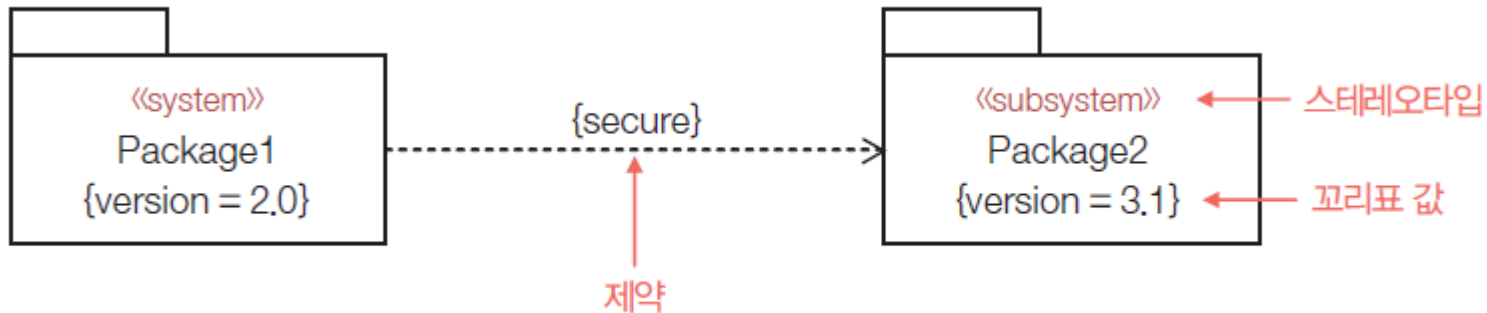


그림 2-27 확장의 예