Introduction to UML

Team 6

201011311 김도희

201111353 박수민

목차

1. UML이란?

통합 모델링 언어(Unified Modeling Language, UML)는 업무, 소프트웨어 애플리케이션, 시스템 아키텍처를 모델링 하는데 사용되는 표준 비주얼 모델링 언어이다. UML이 객체 관리 그룹(Object Management Group, OMG)의 표준이기는 하지만, 객체지향 소프트웨어 애플리케이션만을 위한 것은 아니다. UML은 매우 융통성 있고 사용자 정의할 수 있도록 설계된 그래픽한 언어이다. 따라서 업무 프로세스, 작업 흐름, 쿼리 순서, 애플리케이션, 데이터베이스, 아키텍처 등을 이해하기 위한 모델을 포함하여, 많은 다른 종류의 모델들을 만들 수 있도록 해준다.

UML은 여러 가지 다이어그램들을 제시함으로서 소프트웨어 개발 과정의 산출물들을 비주얼하게 제공하고, 개발자들과 고객 또는 개발자들 간의 의사소통을 원할하게 할 수 있도록 하고 있다. UML은 시스템을 모델링 할 수 있는 다양한 도구들을 제공하기 때문에, 도메인을 모델링하기가 훨씬 용이할 뿐만

아니라 모델링한 결과를 쉽게 파악할 수 있게 된다. 또한 산업계 표준으로 채택되었기 때문에 UML을 적용한 시스템은 신뢰성 있는 시스템으로 평가 받을 수 있다.

일반적으로 가장 큰 장점을 이야기 한다면 개발자의 부재 시 특정 부분에 대한 설명만으로도 쉽게 위치를 찾아 소스 코딩이 가능하고 업무 분할에 매우 유용하며, 이를 통한 공동 개발의 이점이 매우 크다고 할 수 있다.

2. 객체 지향 모델링

1. 모델링의 목적

- 시스템을 현재 또는 원하는 모습으로 가시화
- 시스템의 구조와 행동을 명세화
- 시스템을 구축하는 기본 형태를 제공
- 시스템 분석/설계의 문서화

2. 소프트웨어의 모델과 관점

※ 알고리즘 관점 : S/W의 주요 구성 요소인 Procedure 와 Function을 제어 관점에서 분할하여 시스템을 모형화

- 요구사항 변화에 적응력이 없음
- 대규모 시스템에서는 유지보수를 포함한 관리의 어려움

※ 객체 지향 관점 : S/W 시스템의 기본 요소를 객체 또는 클래스로 파악하여 문제 영역과 해결 영역을 모형화

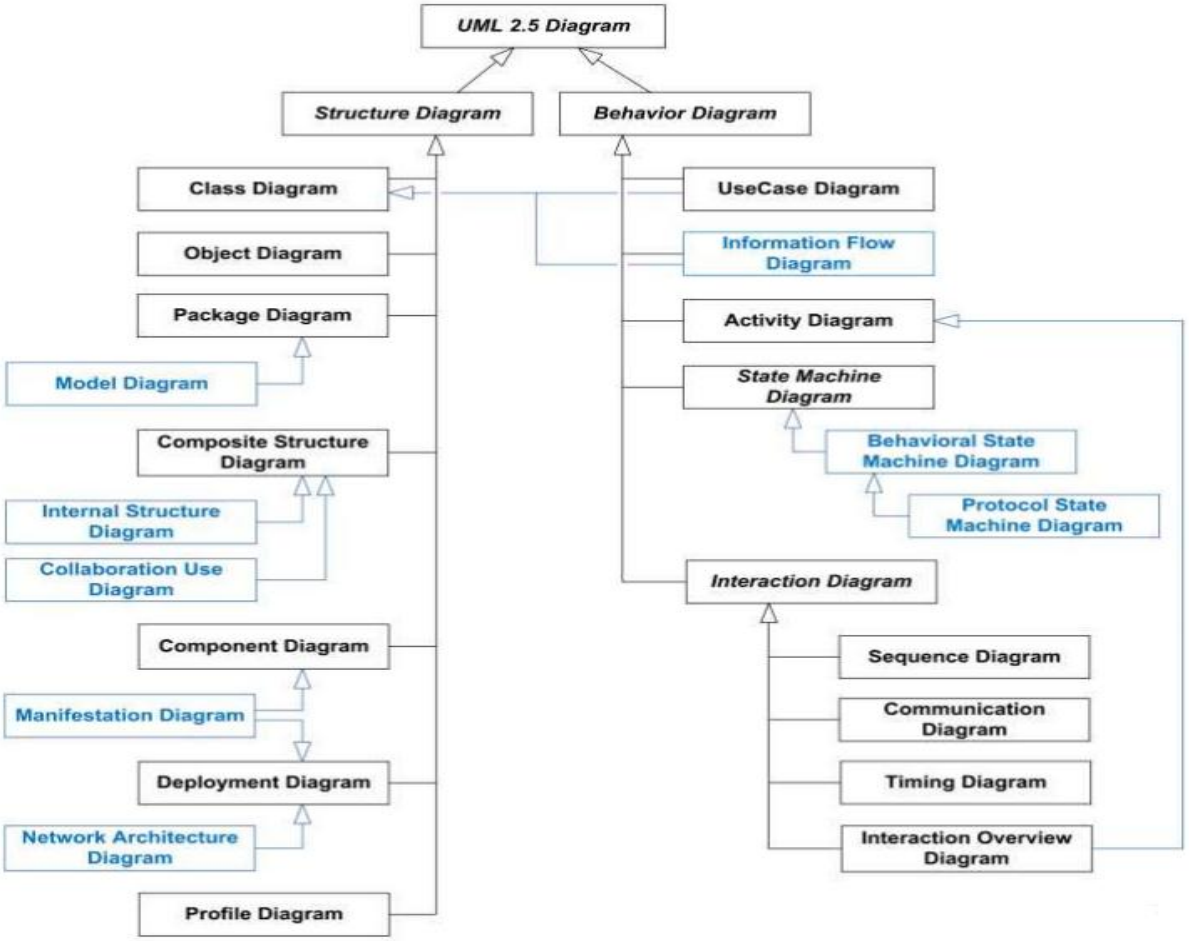
- 객체(Object) : 사물(Thing)을 말하며 고유성과 상태, 행동을 갖음
- 클래스(Class) : 공통적인 객체들의 집합

3. UML은 객체지향 개발만을 위한 것인가?

"이를 위해서 어떻게UML을 사용할 수 있지요? UML은 객체지향 개발만 할 수 있는 것이 아닌가요?" 이것이 가장 큰 오해중 하나이다. 이 오해는UML이 객체지향 시스템을 모델링하기 위해 필요한 것을 충족시키고 컴포넌트 기반의 개발이 가능하도록 고안 되었다는 사실로부터 온다. OO 시스템에서는 일반적으로 '인터페이스'라고 하는 것을 사용하여 여러 컴포넌트들이 서로 연결된다. 서로 다른 컴포넌트들이 어떻게 교류하는지를 이해하기 위해 모델을 구축하는 것이 아주 유용하다. 비록 UML이 원래 이런 이유로 만들어지기는 했지만, 다른 필요성도 염두에 두고 만들어졌다. Grady Booch는 그와 동료들이 UML을 설계할 때, 업계에서 이미 사용 중이던 여러 가지 데이터베이스 모델링 기법들을 포함시켰다고 말한 적이 있다. 비슷하게, Jacobson의 객체 모델링 방법의 위력 중 하나는 업무 모델링 능력이다. 따라서 Jacobson의 객체모델링 방법을 UML에 혼합했을 때, 업무 모델링을 UML에 추가한 것이다.

오늘날, 내장된 확장과 사용자 정의 기능을 사용하여 UML 내에서 여러분이 원하는 거의 모든 것을 모델링 할 수 있다. UML 은 필요한 것을 무엇이든 할 수 있도록 충분히 융통성 있게 해주는 메타모델 기능을 내장하고 있다.

3. UML의 구성요소 - 사물(Thing), 관계(Relationship), 다이어그램(Diagram)



1. 사물(Things)

- 모델의 구성 기본 요소로서 모델을 구성하는 추상적 개념이나 물리적 개체를 나타냄.

2. 관계(Relationship)

- **연관관계(Association)**

일회성이 아닌 지속적으로 유지되는 구조적 관계로서 사물 객체들간의 연결관계를 나타냄

- **포함관계**

연관관계의 특수형태로 사물 사이의 포함관계를 표현

- **일반화 관계(Generalization)**

'is-a kind-of' 관계로써 특수화, 일반화된 사물간의 관계 표현

클래스와 클래스, 유스케이스와 유스케이스 사이의 상속 관계를 표현

- **의존 관계(Dependency)**

두 사물간의 의미적 관계로서, 한쪽 사물의 명세의 변화가 다른 사물에 영향을 주는 관계를 표현

- **실체화 관계(Realization)**

분류자간의 의미적 관계이다. 한 분류자가 다른 분류자가 수행하기로 한 계약을 명세.

인터페이스-클래스, 인터페이스-컴포넌트 간의 관계에 나타낼 수 있음.

3. 다이어그램(Diagram)

※ 어떤 것이 어떻게 작동하는지 시연하고 설명하거나 전체부품들 사이의 관계를 명확히 하기 위한 계획, 스케치, 도면 또는 아우트라인.

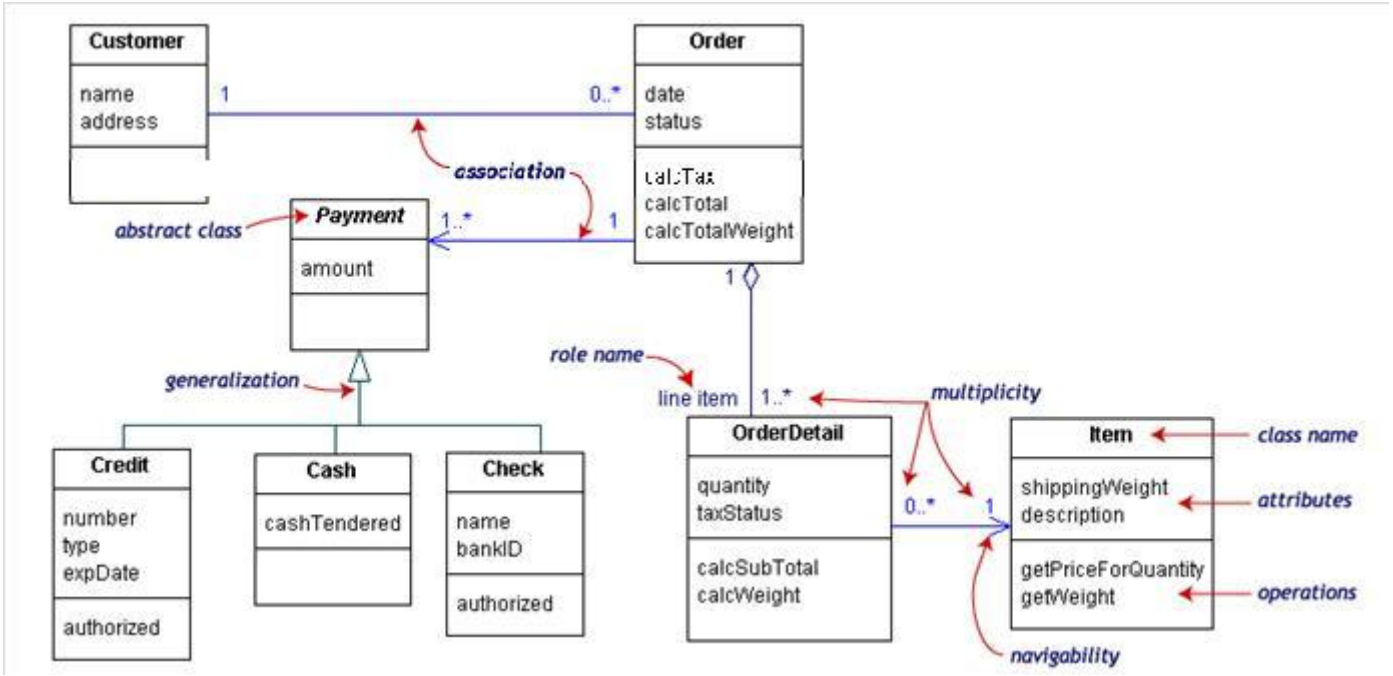
※ 수학. 대수 또는 기하학적 관계의 그래픽적인 표현.

※ 차트 또는 그래프.

여러 모델링 요소들의 레이아웃 또는 시각화이다. 각 UML 다이어그램은 특별한 목적을 위해(특히 시스템의 어떤 측면을 시각화하기 위해) 사용된다. 각 다이어그램은 목적을 달성하기 위해 고유한 UML 기호들을 사용한다.

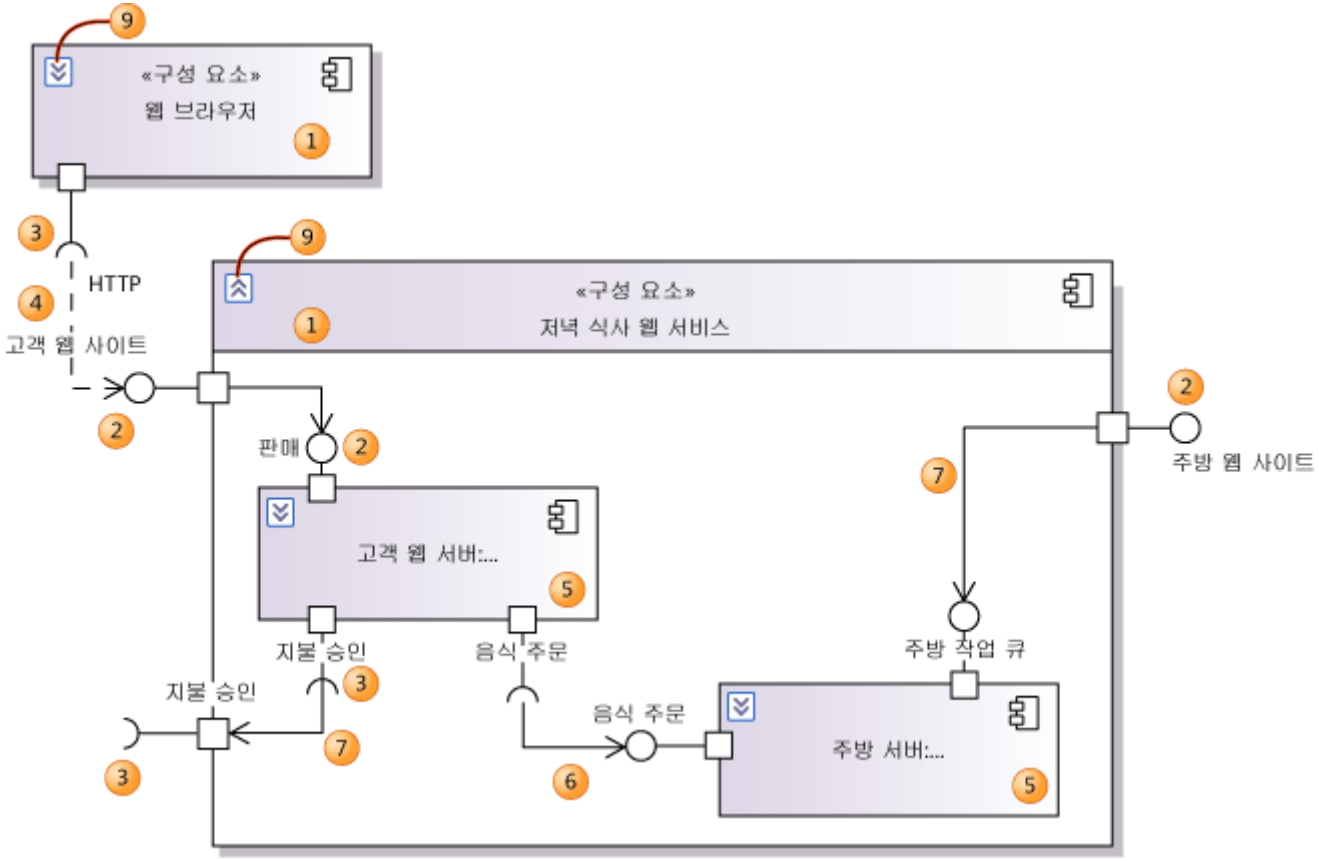
	Diagram		Definition
S T R U C T U R E	클래스	Class	시스템을 구성하는 클래스와 인터페이스 사이의 유지하는 지속적인 정적인 관계를 나타낸 다이어그램
	컴포넌트	Component	소프트웨어의 물리적 단위의 구성과 의존 관계를 표현한 다이어그램
	개체	Object	클래스 다이어그램에 포함된 사물의 인스턴스를 모델링하는데 특정 시점의 객체와 객체 사이의 관계로 표현
	배치	Deployment	노드와 노드에 존재하는 컴포넌트의 물리적 구성을 표현한 다이어그램
	복합체 구조	Composite Structure	각 구성 요소들과 그 요소들이 어떻게 분리/연결 되는지 표현
	패키지	Package	· 패키지란 일종의 관련된 모델링 요소들을 묶는 장치
B E H A V I O R	활동	Activity	시스템 내부의 활동 흐름을 표현한 다이어그램
	유스 케이스	Use Case	요구사항들 가운데서 기능적인 요구사항을 유스케이스란 단위로 표현하고 액터와 이들 사이의 관계를 표현
	상태	Statechart	사건에 따라 순차적으로 발생하는 한 객체의 상태변화를 표현
	협력	Collaboration	시퀀스 다이어그램과 마찬가지의 객체들 사이의 메시지 교환의 순서를 표현
	순차	Sequence	시스템 외부 이벤트를 처리하기 위해 시스템 내부 객체 간에 주고 받는 동적 메시지를
	타이밍	Timing	장치의 동작이나 회로 동작에서 타이밍과의 상호 관계를 나타낸 그림.
	교류 개요	Interaction Overview	제어흐름의 이해를 돕기 위한 방법으로 액티비티 다이어그램의 변형(variant)을 통해 인터렉션을 묘사하는 다이어그램

※ 클래스 다이어그램(class diagram)



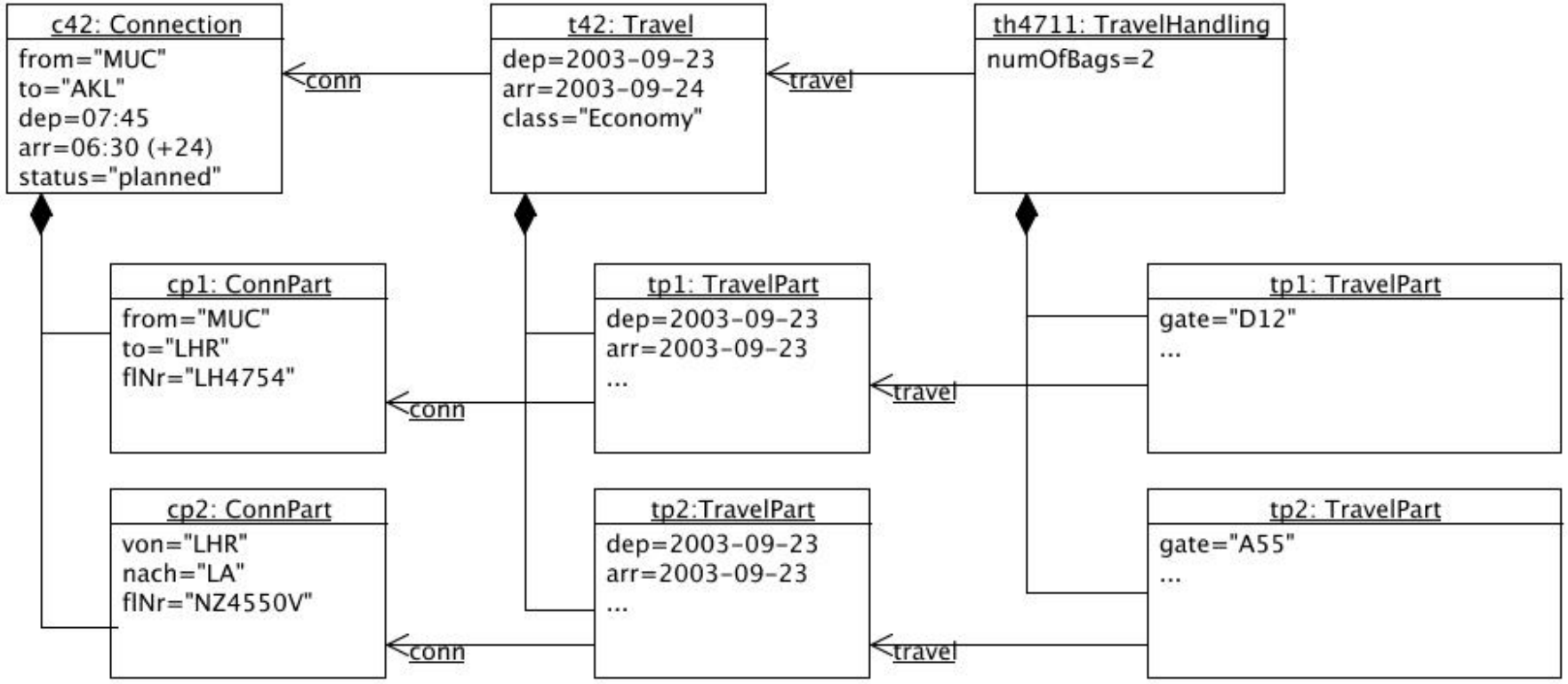
클래스 다이어그램은 UML 모델링에서 사용되는 가장 일반적인 다이어그램으로 시스템, 그 구조 그리고 그들의 상호 관계 내에 존재하는 정적인 것을 나타낸다. 시스템의 논리적 및 물리적 설계를 나타내는 데 주로 사용된다.

※ 컴포넌트 다이어그램(component diagram)



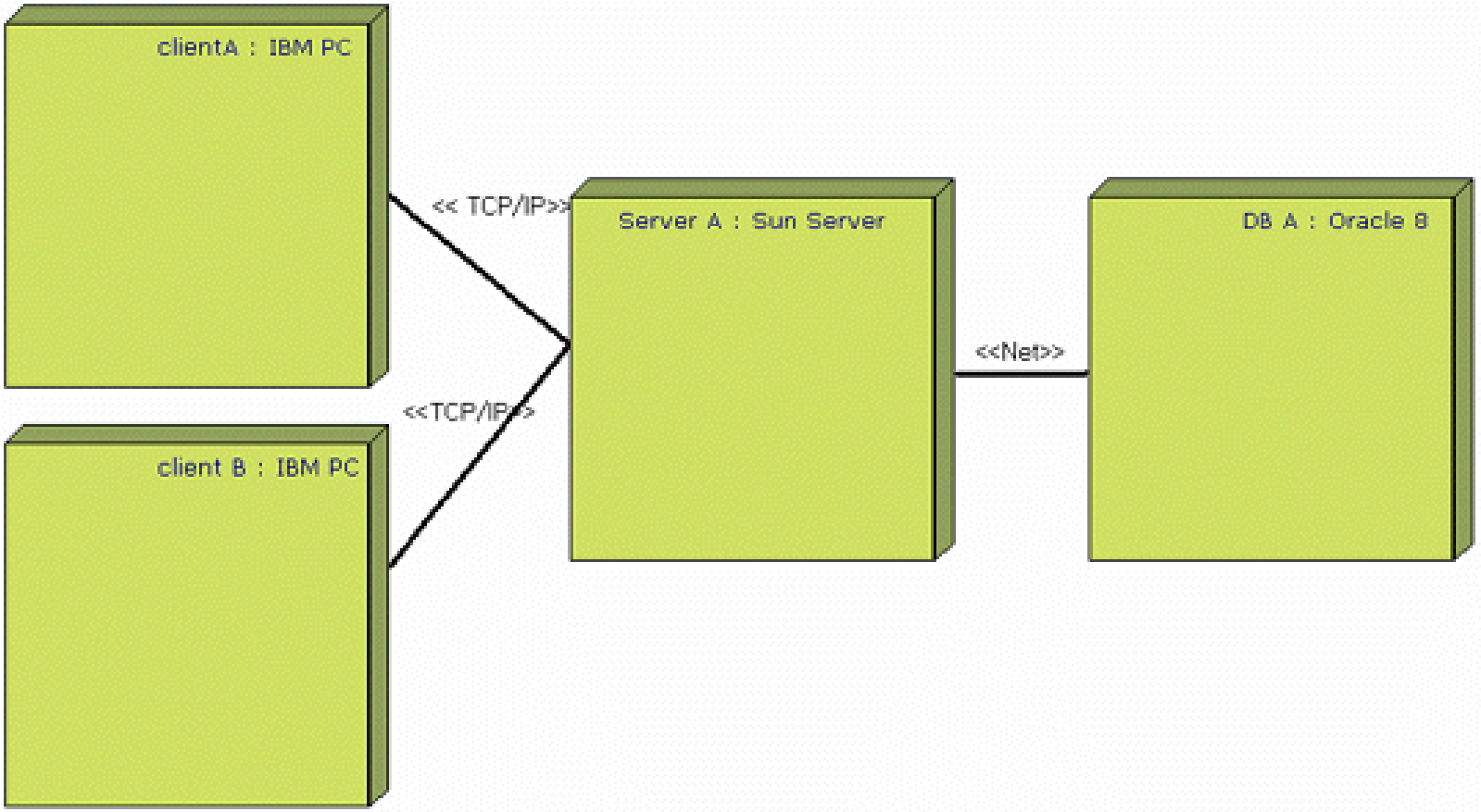
컴포넌트 다이어그램은 컴포넌트 집합 내의 구성과 의존관계를 보여준다. 구현되는 시스템과 시스템 내에서 부품들이 어떻게 상호 작용하는지를 보여준다.

※ 개체 다이어그램(object diagram)



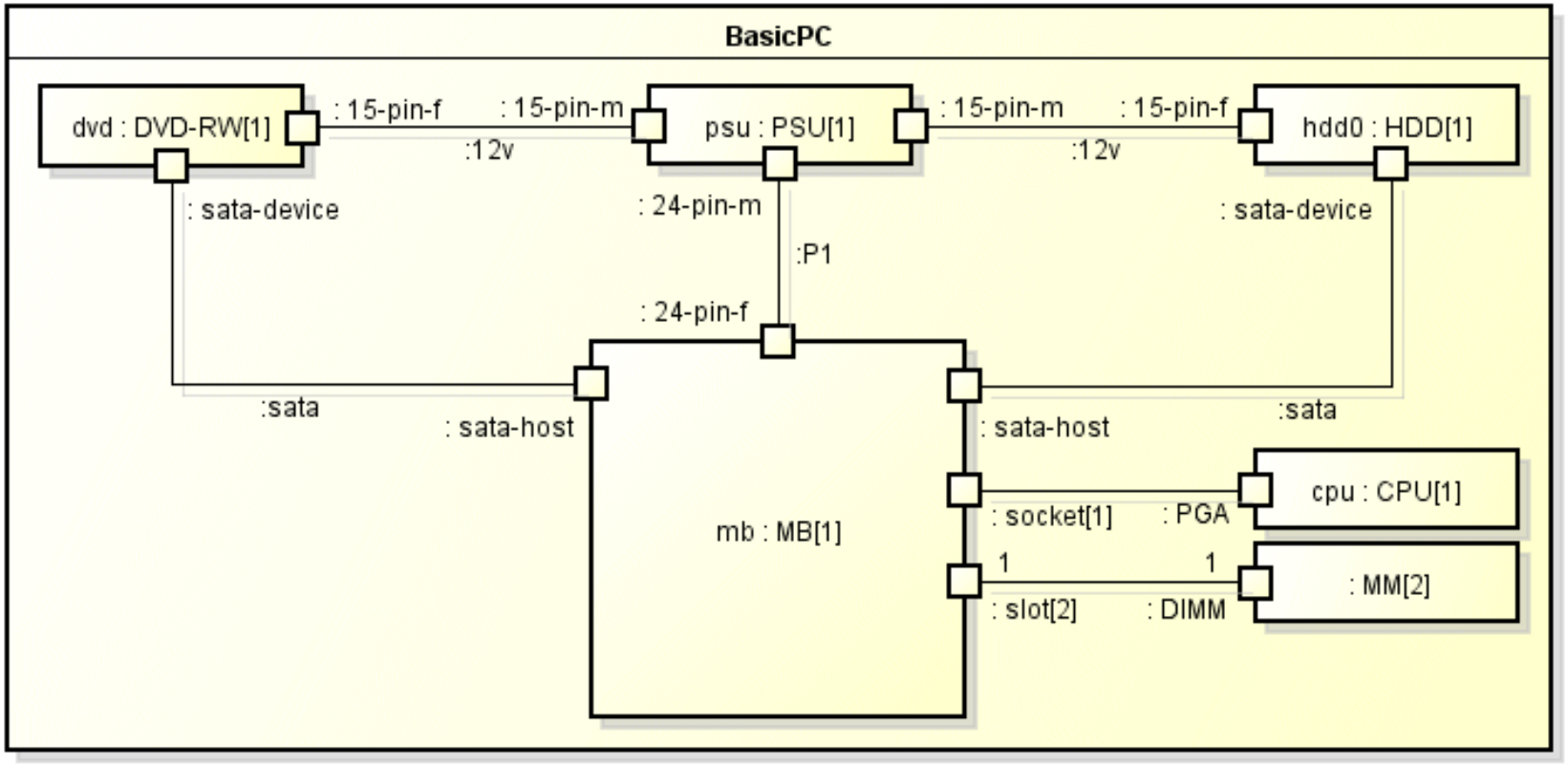
개체 다이어그램은 시스템 내의일련의 개체들 사이의 관계를 보여주는데, 특정시점에 시스템의 스냅샷을 보여준다.

※ 배치 다이어그램(deployment diagram)



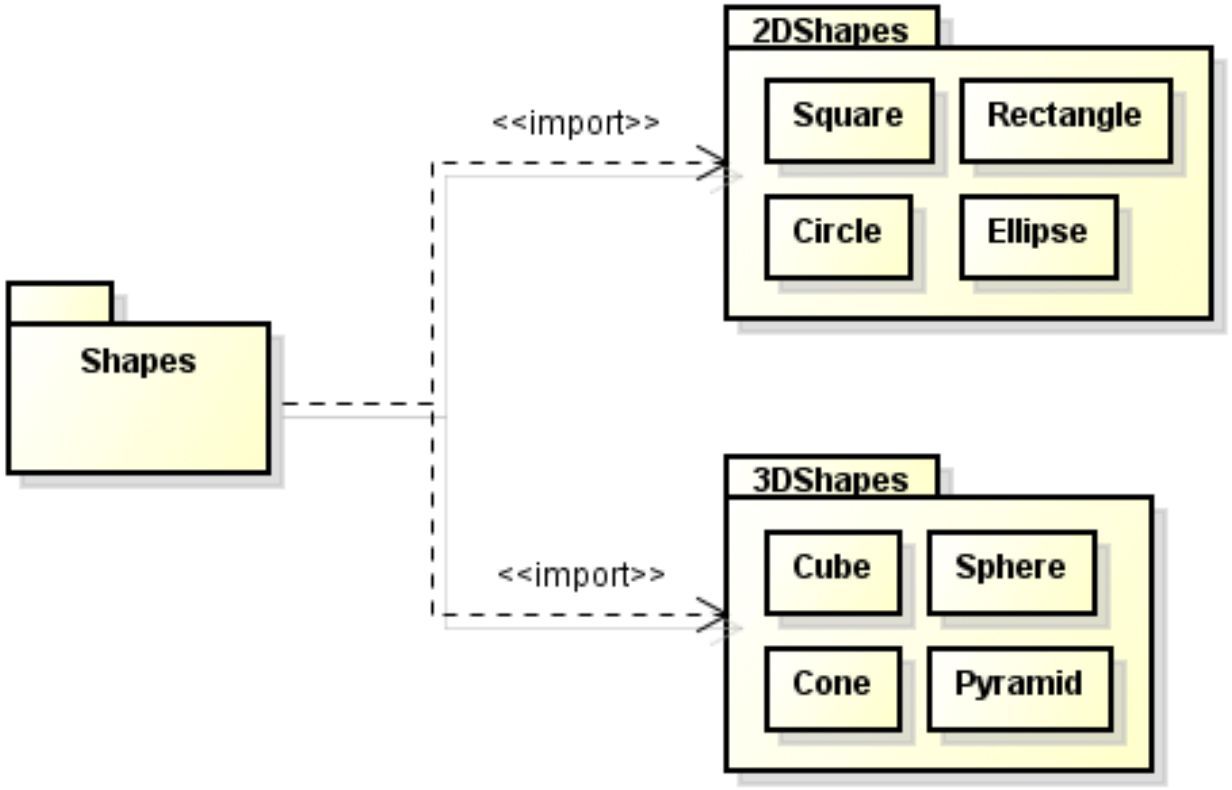
배치 다이어그램은 물리적 시스템의 실행시점의 아키텍처를 보여준다. 배치 다이어그램은 하드웨어와 이 하드웨어 내에 있는 소프트웨어의 설명을 포함할 수 있다.

※ 복합체 구조 다이어그램(composite structure diagram)



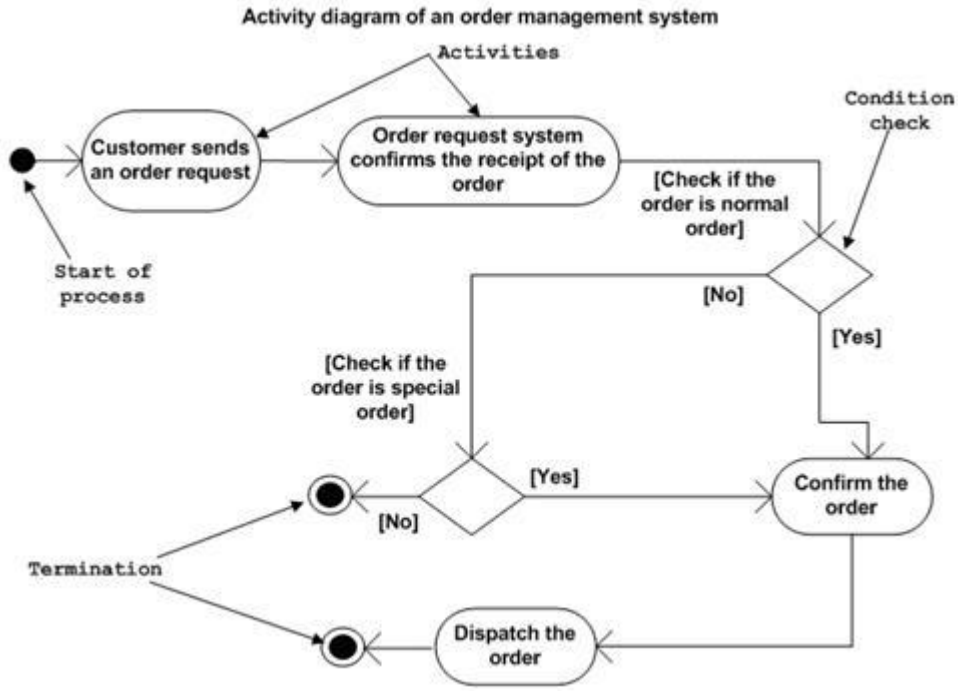
복합체 구조 다이어그램은 모델링 요소들의 내부적 구조를 보여준다.

※ 패키지 다이어그램(package diagram)



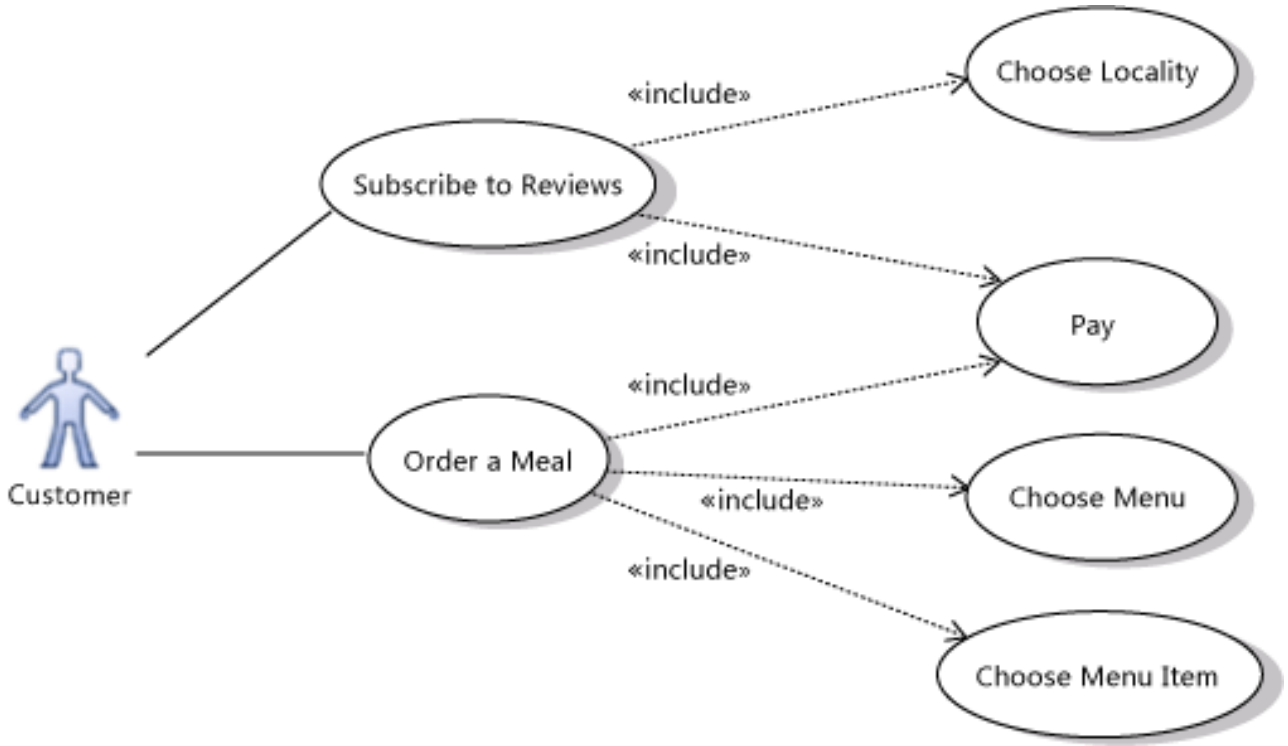
패키지 다이어그램은 패키지 사이의 의존 관계를 나타낸다(패키지는 다른 모델 요소들을 그룹화하는데 사용되는모델 요소이다).

※ 활동 다이어그램(activity diagram)



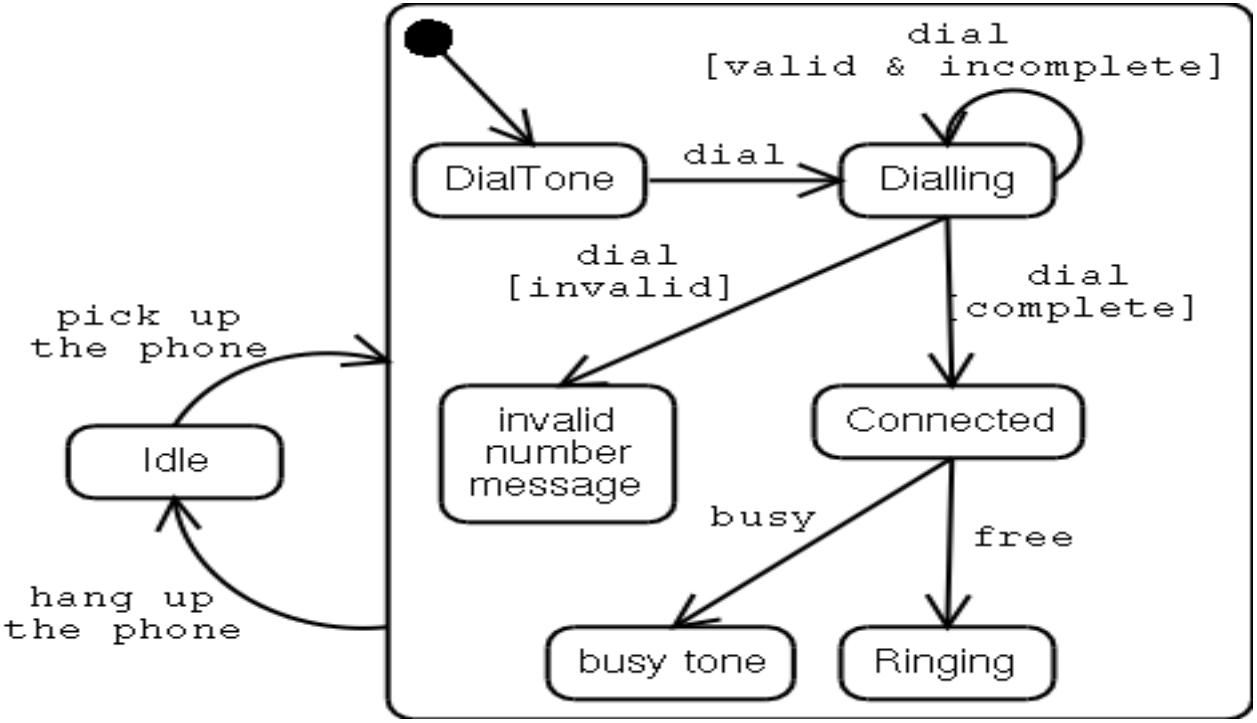
활동 다이어그램 은 시스템 내의 활동들의 흐름을 보여준다. 여러 업무 프로세스들을 설명하는데 자주 사용된다.

※ 유스 케이스 다이어그램(use case diagram)



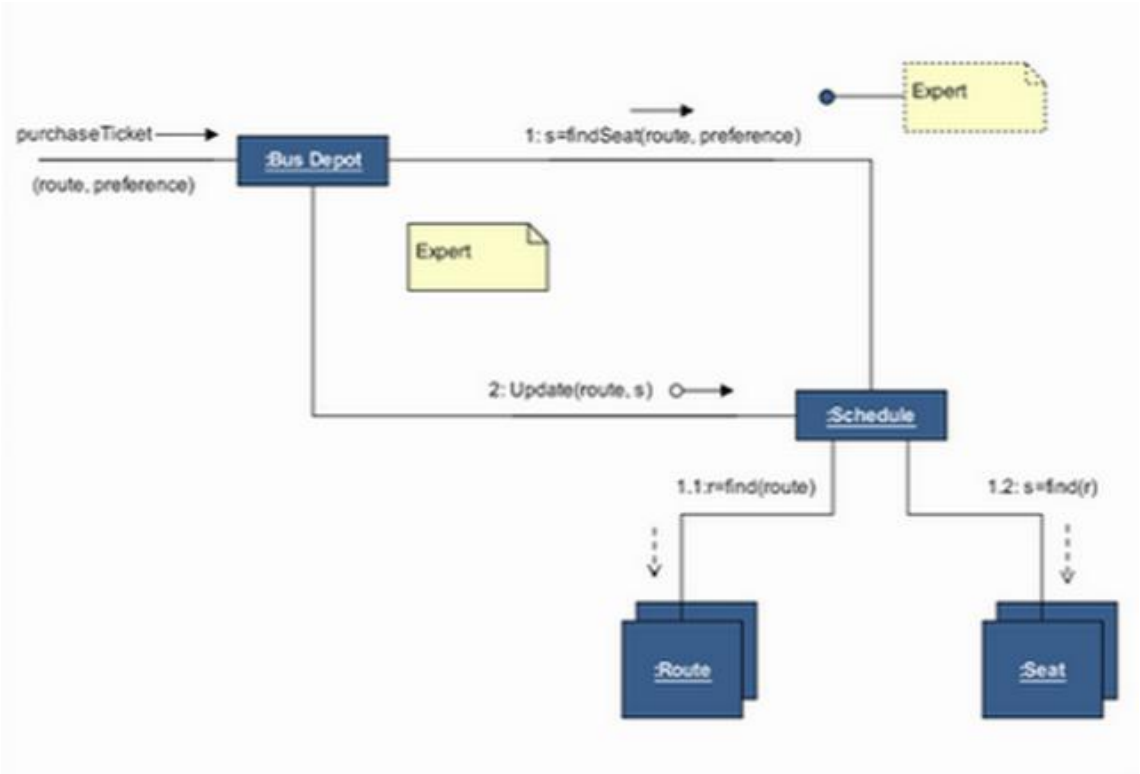
유스 케이스 다이어그램은 시스템이 구현할 업무프로세스를 다룬다. 유스 케이스는 시스템이 작동하는 방법과 시스템과 교류하는 사람들을 나타낸다.

※ 상태 다이어그램(statechart diagram)



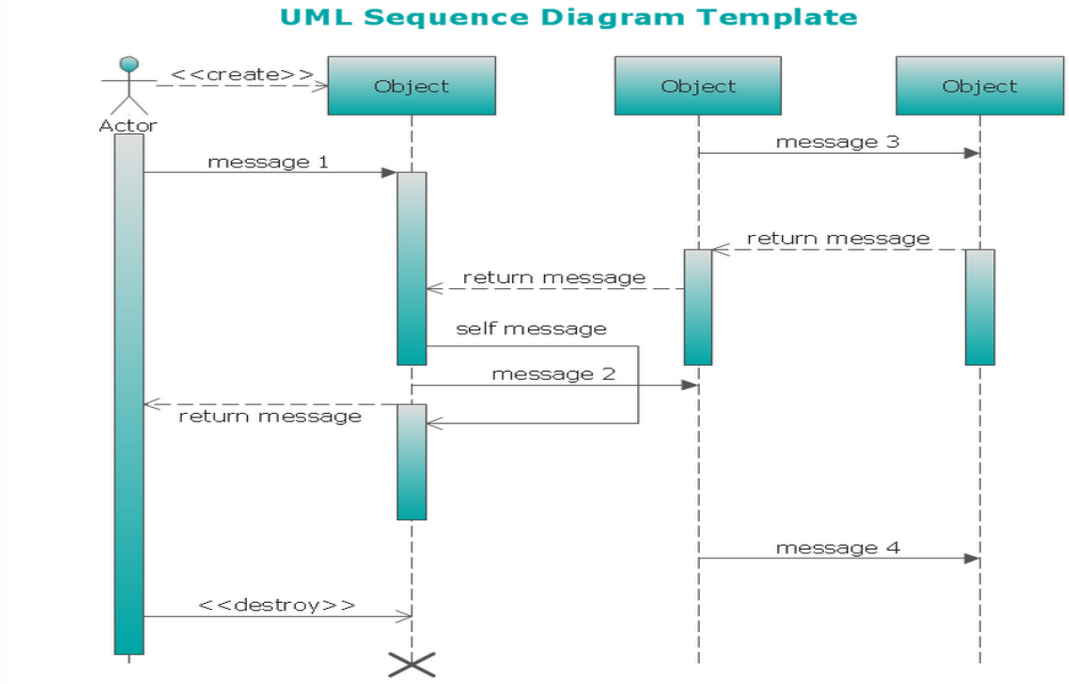
상태 다이어그램은 개체의 상태와, 이 개체가 상태들 사이를 어떻게 전이하는지 보여준다. 상태 다이어그램은 상태, 전이, 이벤트, 그리고 활동을 포함할 수 있다. 상태 다이어그램은 동적 뷰를 제공하며, 이벤트 중심의 행동을 모델링할 때 매우 중요하다. 예를 들어, 전화 교환 시스템 내의 스위치를 나타내기 위해 상태 다이어그램을 사용할 수 있다. 이 스위치는 여러 이벤트들에 기초하여 상태를 바꾸며, 어떻게 이 스위치가 작동하는지를 이해하기 위해 상태 다이어그램에서 이 이벤트들을 모델링할 수 있다. UML 2.0에서는 상태 기계 다이어그램(state machine diagram)이라고 한다.

※ 협력 다이어그램(collaboration diagram)



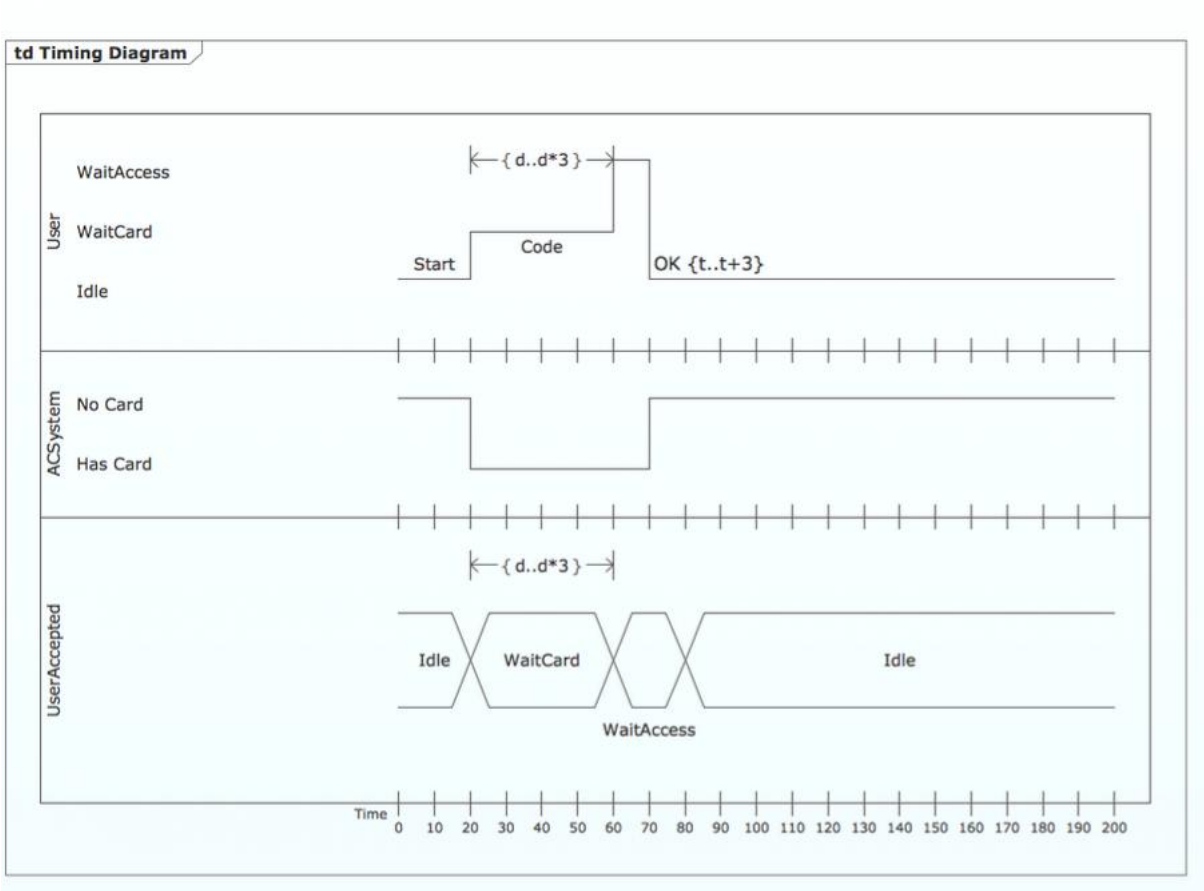
협력 다이어그램(collaboration diagram)은 순차 다이어그램(sequence diagram)처럼 교류 다이어그램(interaction diagram)의 일종이다. 협력 다이어그램은 개체들이 어떻게 협력하고 교류하는지를 강조한다. UML 2.0 에서 협력 다이어그램과 대등한 것은 통신 다이어그램(communication diagram)이다.

※ 순차 다이어그램(sequence diagram)



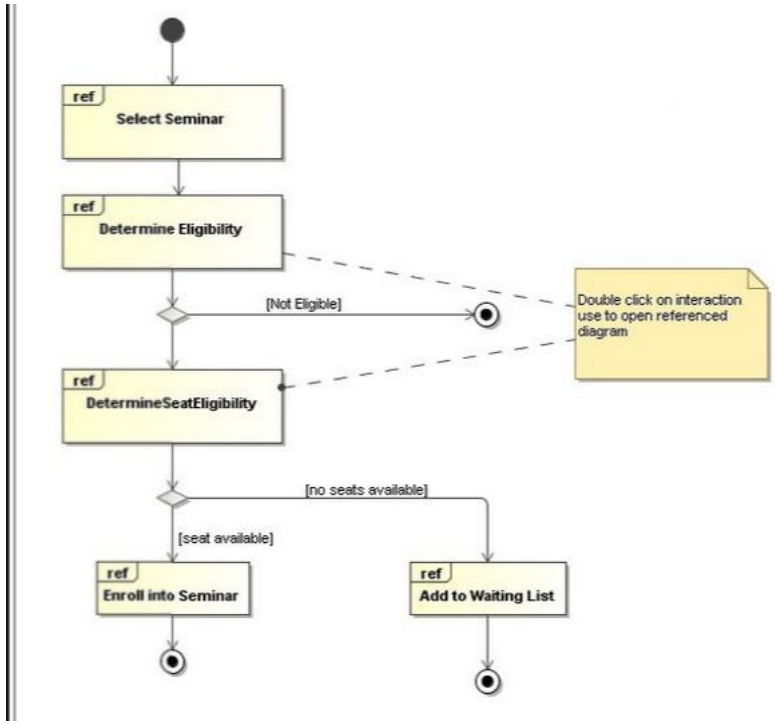
순차 다이어그램(sequence diagram)은 또 다른 종류의 교류 다이어그램이다. 순차 다이어그램은 시스템의 다른 요소들 사이의 메시지들의 시간 순서를 강조한다.

※ 타이밍 다이어그램(timing diagram)



타이밍 다이어그램(timing diagram)은 또 다른 종류의 교류 다이어그램이다. 세부 타이밍 정보와, 교류하는 요소들의 상태 또는 조건 정보에 대한 변경 사항을 나타낸다.

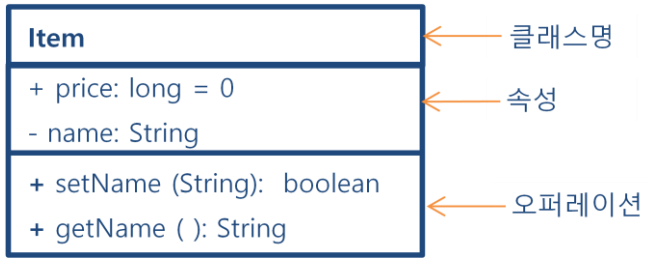
※ 교류 개요 다이어그램(interaction overview diagram)



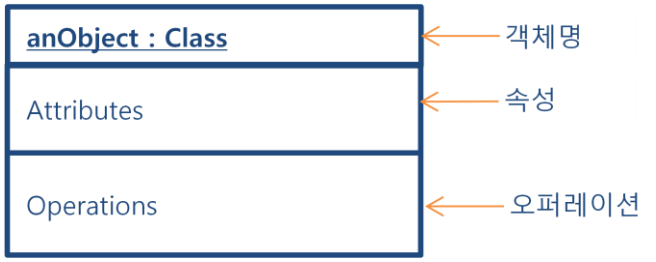
교류 개요 다이어그램(interaction overview diagram)은 교류 순차들(interaction sequences) 사이의 제어 흐름에 대한 개요를 보여주는 데 사용되는 고수준의 다이어그램이다.

4. 표기법

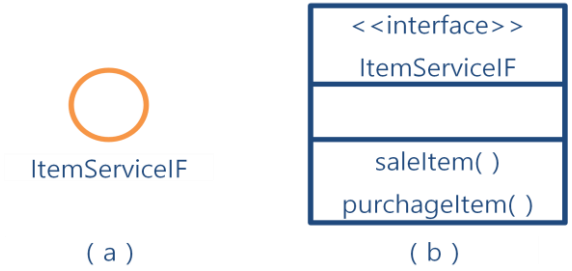
※ 클래스(Class)



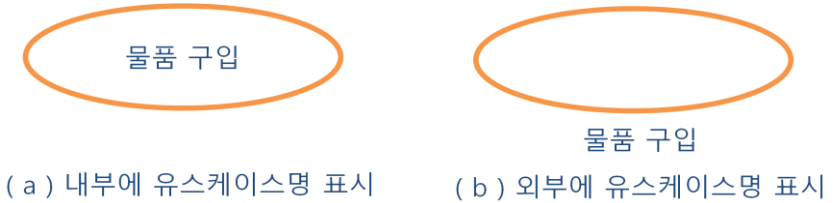
※ 객체(Object)



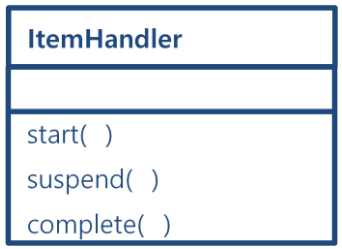
※ 인터페이스(Interface)



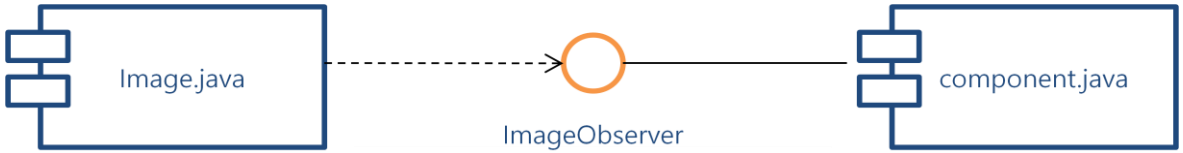
※ 유스 케이스(Use Case)



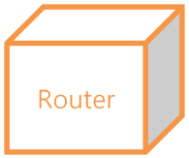
※ 액티스 클래스(Active Class)



※ 컴포넌트(Component)



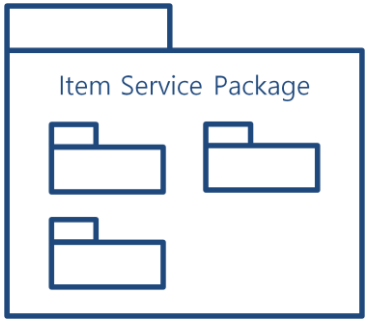
※ 노드(Node)



※ 인터럭션(Interaction)

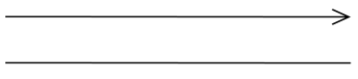
※ 상태 머신(State Machine)

※ 패키지(Package)



※ 노트(Note)

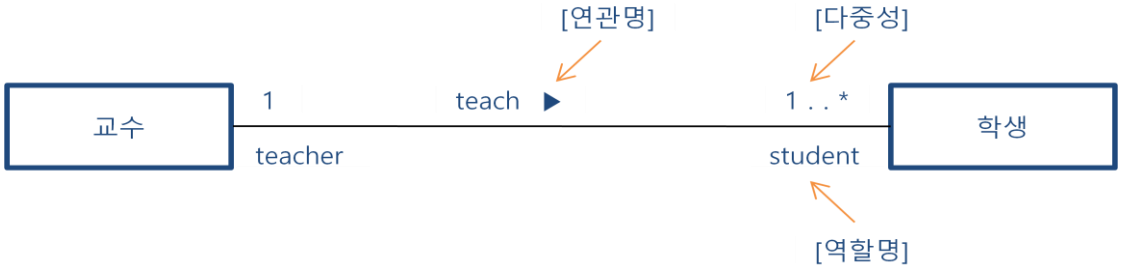
※ 연관관계(Association)



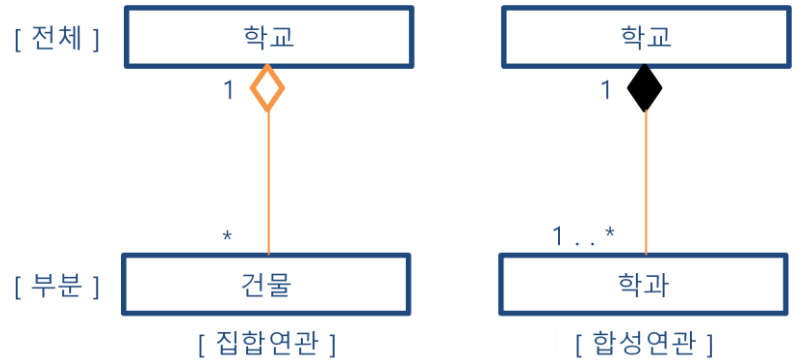
한쪽으로만 연관관계를 가질 때

양쪽으로 연관관계를 가질 때

※ 다중성(Multiplicity)

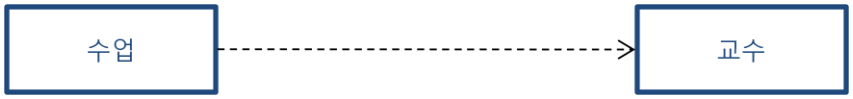


※ 포함관계



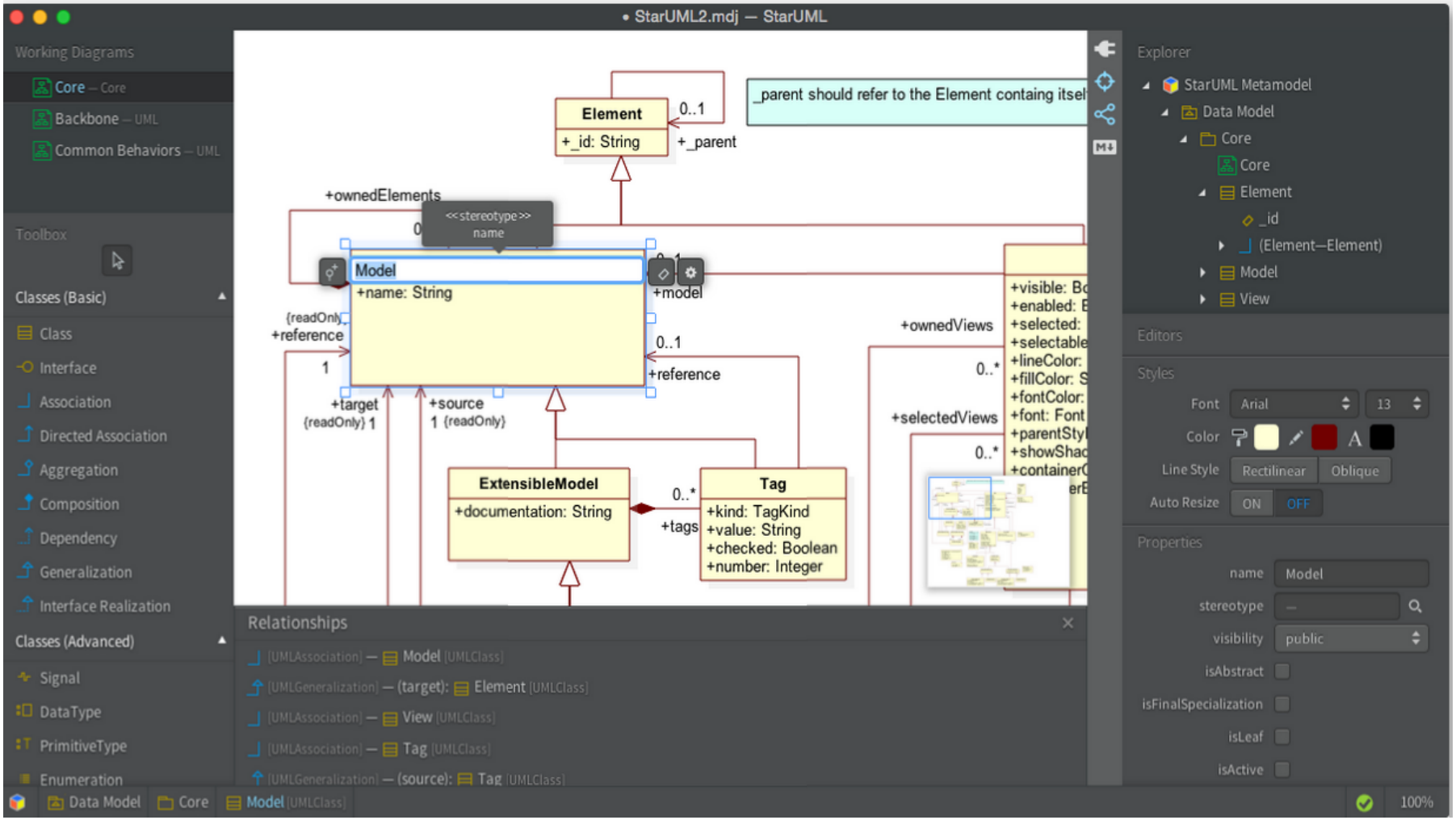
※ 일반화 관계(Generalization)

※ 의존 관계(Dependency)



※ 실체화 관계(Realization)

4. UML Tool - star UML



StarUML™은 UML(Unified Modeling Language)을 지원하는 소프트웨어 모델링 플랫폼입니다. UML 버전 1.4에 기반을 두고 있으며, UML 버전 2.0의 표기법을 적극적으로 지원하고 있습니다. 총 11가지의 다양한 종류의 다이어그램을 제공할 뿐만 아니라 UML 프로파일 개념과 템플릿 기반의 문서 및 코드 생성을 지원하여 MDA(Model Driven Architecture) 접근방법을 적극적으로 지원합니다. 또한 고객의 환경에 대한 맞춤 능력이 우수하고 기능에 대한 확장성이 매우 뛰어난 것이 장점입니다. 가장 선도적인 소프트웨어 모델링 도구 중의 하나인 StarUML™을 사용하면 소프트웨어 프로젝트의 생산성(Productivity), 품질(Quality)이 획기적으로 높아진다는 것을 실감하실 수 있습니다.

5. 참조문헌

● StarUML 5.0 사용자 가이드

http://staruml.sourceforge.net/docs/user-guide(ko)/toc.html

● 위키피디아

http://en.wikipedia.org/wiki/Unified_Modeling_Language

● 운명적 존재를 위한 UML

● UML 제대로 이해하기 - 이민규

그 외 여러 인터넷 사이트!