UML 개요

일반 목표	UML의 개요를 이해할 수 있다.
수업 목표	• UML의 개요 • UML 다이어그램
수행 과제	• UML의 기본 개념 이해

모델링

- 시스템을 구축할 때 개발자가 고민하고 결정하는 모든 활동
- 구현 단계 이전의 요구사항 정의, 분석, 설계에서 수행하는 모든 활동들

모델

- 모델링 활동의 결과
- 요구사항 정의 활동의 결과는 요구사항 모델, 분석 활동의 결과는 분석 모델, 설계 활동의 결과는 설계 모델이라고 한다.

모델링 언어

- 모델을 표현할 때 사용되는 언어이다.
- UML은 모델링 언어의 일종이다.
- 모델링 언어를 이용한 모델링을 전문적으로 지원하는 도구를 CASE Tool이라고 한다.(Rose, Together, Visio등)

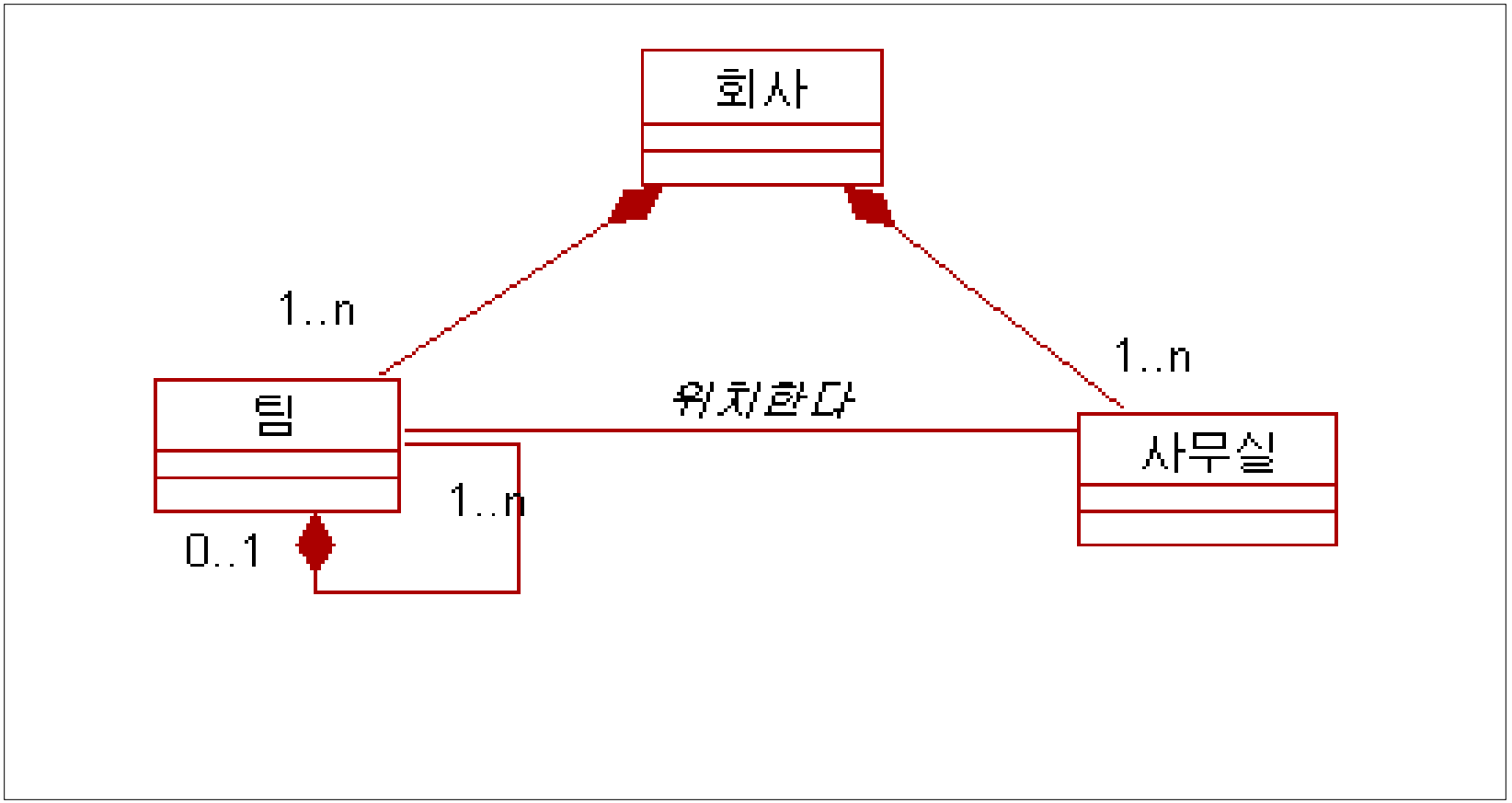
모델링과 프로그래밍의 비교

	모델링	프로그래밍
목적	구축할 시스템의 모습 정의	시스템의 실제 구현
세부 수행 활동	요구사항 정의, 분석, 설계	소스 코드 편집, 컴파일, 디버깅
결과물	모델	소스코드를 포함한 구현된 시스템
표기법	모델링 언어 (UML,ERD,DFD)	프로그래밍 언어 (Java, C#)
지원 도구	CASE 도구 (Rose, Together)	개발 도구 (Jbuilder, Visual Studio.NET)

UML의 특징

- 객체 지향 시스템을 모델링 하는데 사용되는 표준 언어
- 시각적 모델링 언어

한 회사에는 여러 팀이 있다. 각 팀에 대해서는 해당 팀을 관리하는 하나의 팀이 있으며, 한 팀은 다른 여러 팀을 관리할 수 있다. 한 회사는 여러 사무실을 가지고 있으며, 한 팀은 하나 이상의 사무실에 위치 할 수 있고, 반 대로 한 사무실에서 여러 팀이 작업 할 수 있다.

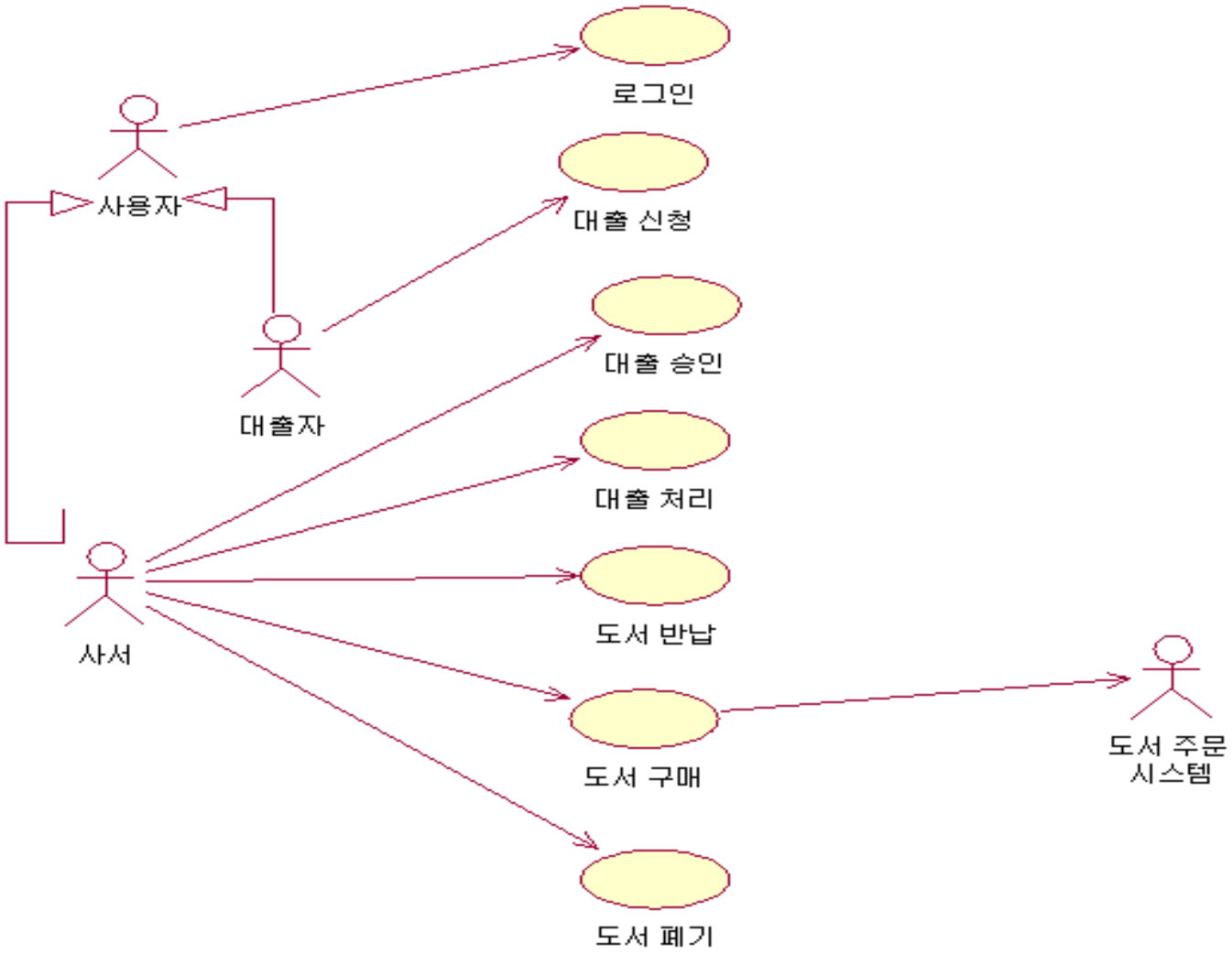


UML 다이어그램

- Use case 다이어그램
- Class 다이어그램
- Sequence 다이어그램
- Activity 다이어그램
- Statecharts 다이어그램
- Collaboration 다이어그램
- Component 다이어그램
- Deployment 다이어그램

유즈케이스 다이어그램

- 시스템에 대한 요구사항을 정의하기 위하여 시스템과 상호 작용을 하는 외부 환경와 시스템이 제공해야 하는 기능을 표현할 때 사용한다.

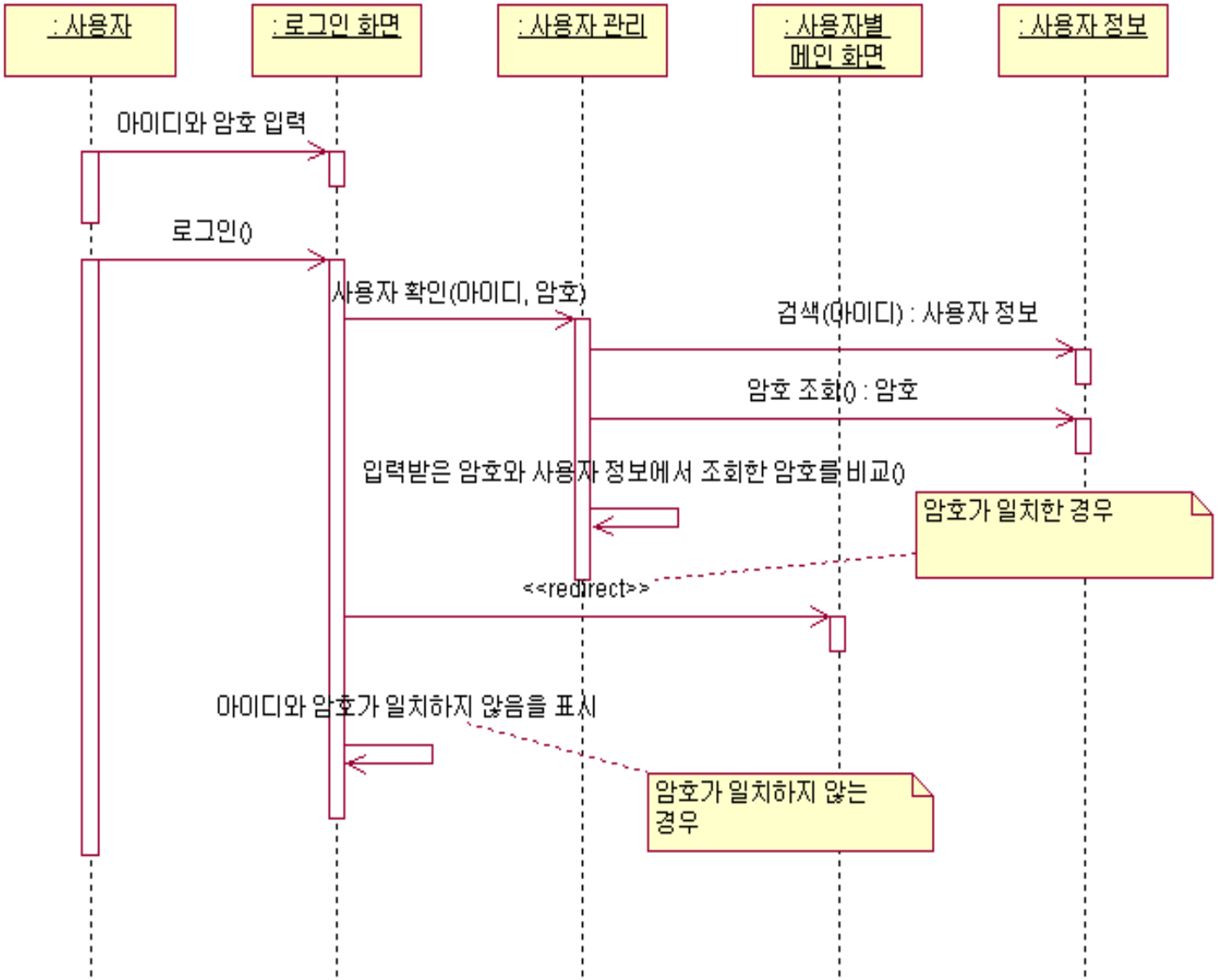

로그인
사용자
대출 신청
대출 승인
대출자
대출 처리
사서
도서 반납
도서 주문
시스템
도서 구매
도서 폐기

클래스 다이어그램

- 시스템을 구성하는 클래스 및 각 클래스의 속성과 연산 그리고 클래스 사이의 관계를 표현하는 데 사용된다.
- 객체지향 시스템 개발 시 가장 중요한 역할을 한다.

시퀀스 다이어그램

- 객체 지향 시스템은 객체들 간의 메시지에 의한 상호작용을 통하여 구현된다.
- 객체들 사이의 상호작용을 통하여 구현된다.


: 사용자
: 로그인 화면
: 사용자 관리
: 사용자별 메인 화면
: 사용자 정보
아이디와 암호 입력
로그인()
사용자 확인(아이디, 암호)
검색(아이디) : 사용자 정보
암호 조회() : 암호
입력받은 암호와 사용자 정보에서 조회한 암호를 비교()
암호가 일치한 경우
<<redirect>>
아이디와 암호가 일치하지 않음을 표시
암호가 일치하지 않는 경우

협력 다이어그램

- 객체들 사이에 주고 받는 메시지를 기술할 때 사용된다.
- 시퀀스 다이어그램과 동일한 목적으로 사용된다.

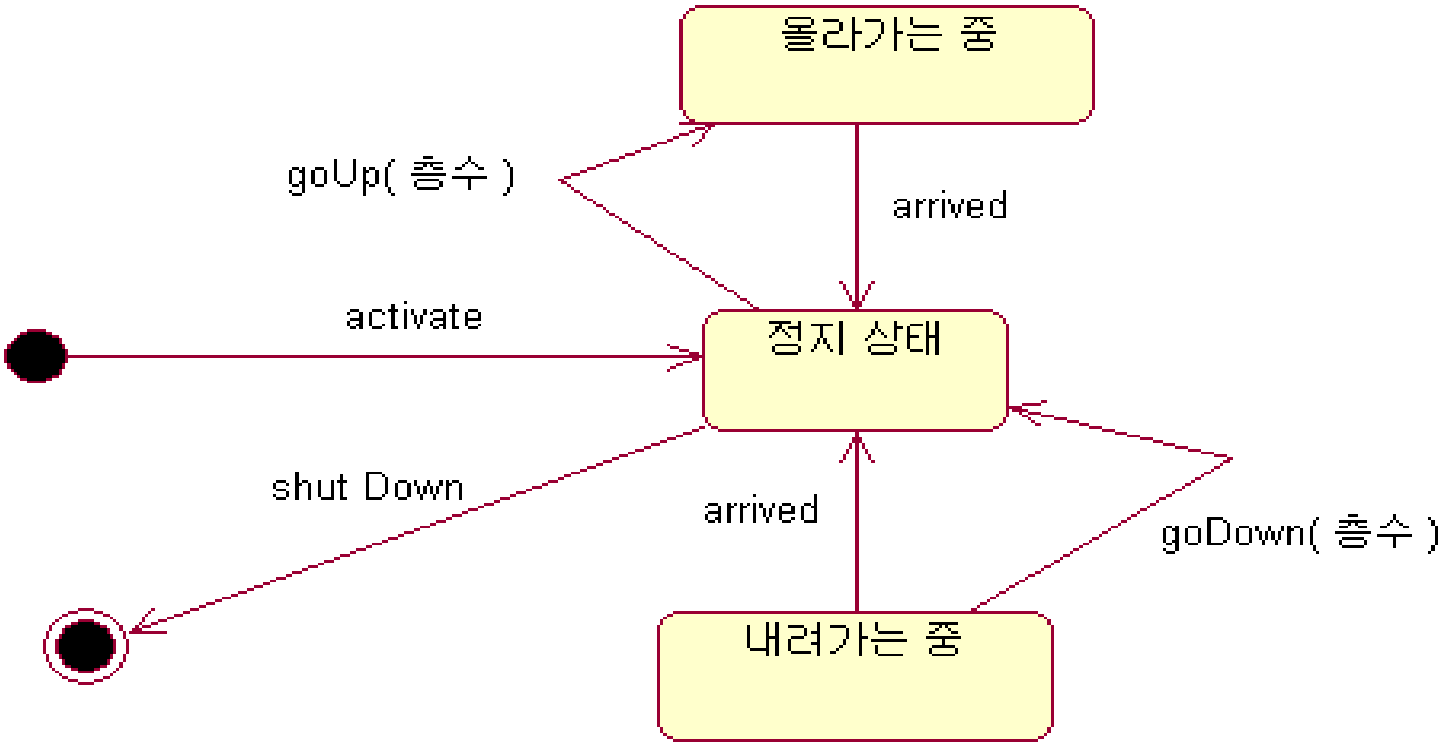
시퀀스, 협력다이어그램 비교

	시퀀스 다이어그램	협력 다이어그램
객체 간의관계 표시	표시 안됨	명시적으로 링크가 표시됨
메시지 전송의 순서	메시지의 상하 위치와 동일	각 메시지에 부여된 순번을 통해서 구분
주요 용도	객체들 사이의 상호작용을 시간적인 면에서 보다 쉽게 파악하고 싶은 경우	시간적인 면보다 객체들 사이의 관계를 보다 쉽게 파악하고 싶은 경우

상태차트 다이어그램

- 하나의 객체 또는 시스템 전체에 대해서 객체 내부 또는 시스템의 자세한 행동을 기술하는데 이용된다.

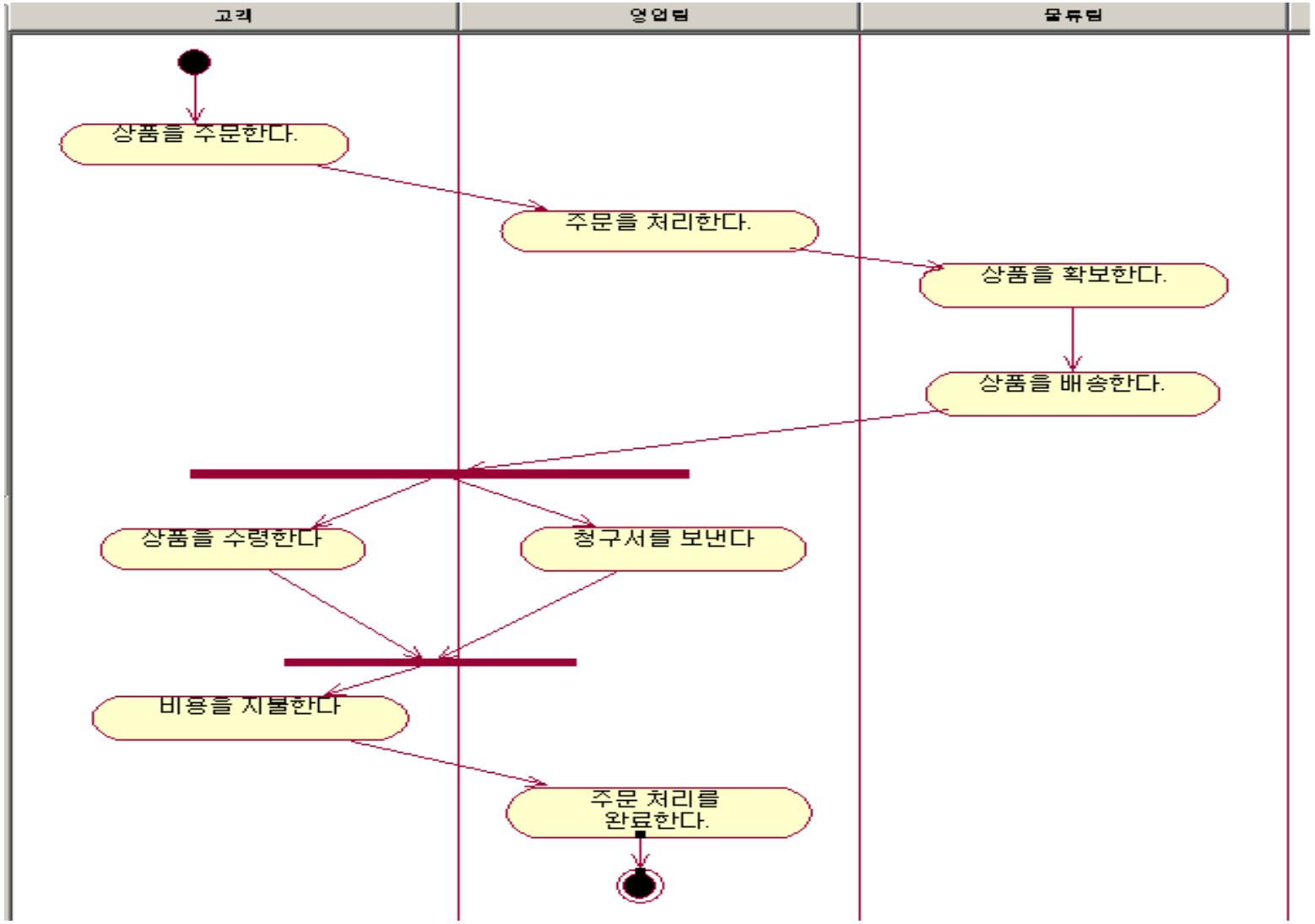
[엘리베이터 객체의 행동 표현]



액티비티 다이어그램

- 수행되는 활동의 순서와 활동 수행의 주체 등을 기술할 때 유용한다.
- **업무 흐름**, 화면 흐름, 연산의 알고리즘을 표현할 때 이용된다.

[상품 주문에 대한 처리 과정]



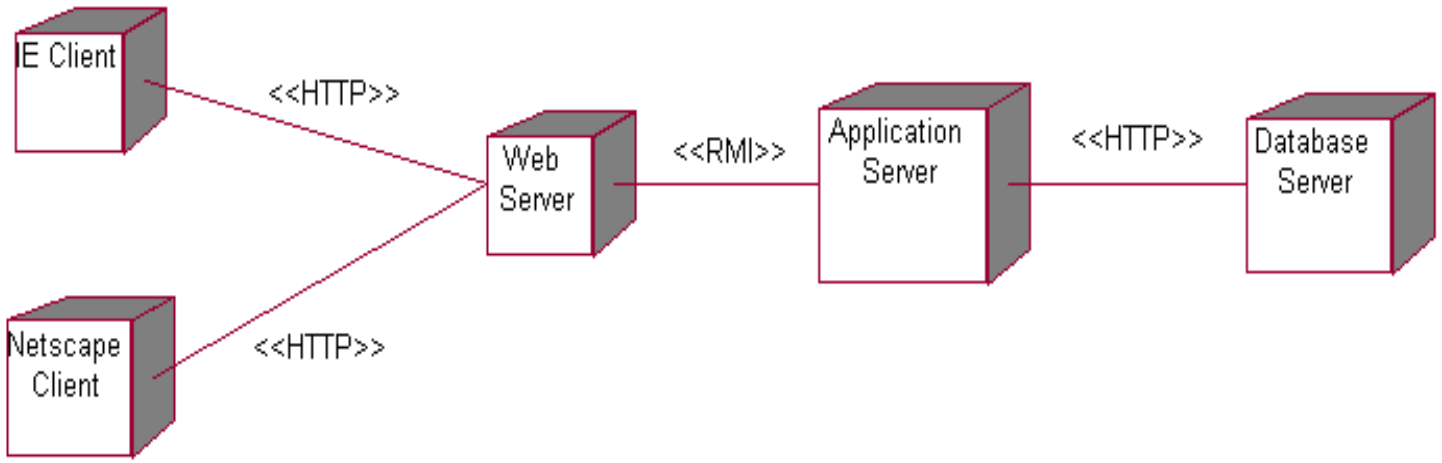
컴포넌트 다이어그램

- 시스템을 구성하는 물리적인 컴포넌트와 그들 사이의 의존 관계를 표시한다.
- UML 에서의 컴포넌트란 실제로 구현되어서 존재하는 것(파일)


A
컴포넌트
B
컴포넌트

배치 다이어그램

- 시스템의 물리적인 요소를 모델링 할 때 사용된다.
- 시스템을 구성하는 처리 장치 즉, 컴퓨터와 그들 사이의 통신 경로를 기술할 때 사용된다.


IE Client
<<HTTP>>
Web Server
<<RMI>>
Application Server
<<HTTP>>
Database Server
Netscape Client
<<HTTP>>

소프트웨어의 관점

- 유즈케이스 관점
- 설계 관점
- 구현 관점
- 배치 관점

유즈케이스 관점

- 시스템을 볼 때 시스템 외부에 있으면서 시스템과 상호작용을 하는 대상과 시스템이 제공해야 할 기능이 무엇인지에 초점을 둔다.
- 개발자 뿐만 아니라 고객에게도 중요한다.
- 시스템을 검증 시 중요한 역할을 한다.
- 시스템이 제공해야 하는 기능이 무엇(What)인가에 초점을 둔다.

설계 관점

- 유즈케이스 관점에서 정의된 기능을 시스템이 제공하기 위해서는 어떤 클래스가 필요하고 이들 클래스들이 서로 어떻게 이용/호출되는 지에 초점을 맞춘다.
- 시스템의 내부 구조 및 내부 기능에 초점을 맞춘다.

설계 관점

구분	관심 사항	이용되는 다이어그램
정적인 측면	클래스 및 클래스간의 관계	클래스
동적인 측면	클래스 내의 동작	상태차트
	클래스 간의 상호작용	시퀀스, 협력
	클래스의 연산의 동작	액티비티

구현 관점

- 시스템을 구성하는 구현된 물리적인 요소(컴포넌트) 즉, 파일과 파일 간의 의존 관계에 초점을 둔다.
- 컴포넌트 다이어그램이 이용된다.

배치 관점

- 시스템을 구성하는 물리적인 처리 장치와 각 처리 장치에 배치되는 컴포넌트에 초점을 둔다.
- 배치 다이어그램