노동부 Ministry of Labor 중소기업직업훈련컨소시엄

Korea Software Technology Association

UML2.x 기초 다루기

훈련기간: 2010.01.25 ~ 02.05

강사명: 손재현 *-넥스트리소프트*
-jhsohn@nextree.co.kr

Copyright © 2009 Nextree Soft, All rights reserved

과정 개요

❑ 교육 목표 & 특징

- UML2.x의 이해
- 유스케이스 작성
- 객체모델링 이해
- UML2.x의 다양한 다이어그램 이해 및 활용
- 모델링 도구 사용법 습득

Copyright © 2009 Nextree Soft, All rights reserved

강의 요구 기술

- **본 강의는 아래 기술에 대한 이해를 필요로 합니다.**
  - 객체지향 언어(Java) 기초
  - 개발프로세스 이해

Copyright © 2009 Nextree Soft, All rights reserved

교육 일정표

❑ 교육은 매 회 4 시간씩 총 5회에 걸쳐 진행합니다.

1 일차	2 일차	3 일차	4 일차	5 일차
- UML 개요 UML 소개 UML 역사 UML 다이어그램분류	- 구조 다이어그램 클래스 객체 컴포넌트 배치	- 행위 다이어그램 유스케이스 액티비티 상태기계	- 상호작용 다이어그램 상호작용 Overview 시퀀스 커뮤니케이션 타이밍	- 유스케이스 I 유스케이스 개요 유스케이스 내용 유스케이스 다이어그램
6 일차	**7 일차**	**8 일차**	**9 일차**	**10 일차**
- 유스케이스 II 유스케이스 목표수준 유스케이스 명세 유스케이스 패턴	- 유스케이스 III 유스케이스 분석기법 분석클래스 제어클래스 실체클래스	- 요구사항 모델실습 I 유스케이스 사용자 시나리오 핵심개념 모델	- 요구사항 모델실습 II 인터페이스 추출 유스케이스 분석 컴포넌트 식별	- 설계모델 실습 컴포넌트 설계 유스케이스 설계 도메인 모델

Copyright © 2009 Nextree Soft, All rights reserved

4일차 – 상호작용 다이어그램

ONE STEP AHEAD

Copyright © 2009 Nextree Soft, All rights reserved

1. 시퀀스 다이어그램

ONE STEP AHEAD

- ❑ 개요
- ❑ 구성요소
- ❑ 작성 및 주의사항

Copyright © 2009 Nextree Soft, All rights reserved

❑ 메타포*Metaphor*

- ATM 현금인출 흐름
  - ATM 기계에 카드 삽입
  - 현금인출 서비스 버튼 클릭
  - 현금인출 가능한 금액을 입력 후 확인
  - 인출 계좌 비밀번호 입력 후 확인
  - 현금과 영수증을 확인한 후 카드를 수령
- 시간의 순서를 중시하고 객체간의 메시지를 도식화

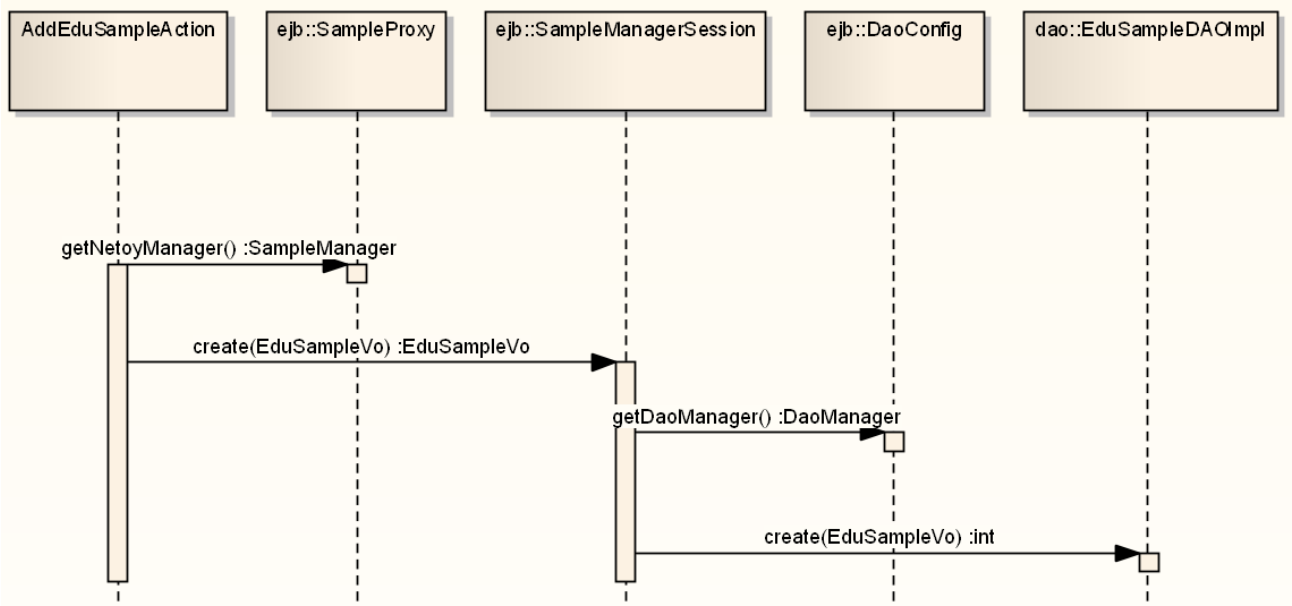
Copyright © 2009 Nextree Soft, All rights reserved

❑ 동적 모델링*Dynamic Modeling* 기법

- 객체와 객체 사이의 동적 상호관계를 정의한 모델

❑ 시간 개념

- 종축을 시간 축으로 하여 시간의 흐름을 나타내며 메시지의 순서에 초점을 두어 객체들 간에 주고 받는 메시지를 보여줌
- 시스템 실행 시 생성되고 소멸되는 객체를 표기



Copyright © 2009 Nextree Soft, All rights reserved

개요

❑ 작성목적

- 객체간의 동적 상호작용을 시간적 개념으로 모델링
- 객체의 오퍼레이션과 속성을 상세히 정의
- 유스케이스를 실현*Realization*
- 프로그램밍 사양을 정의

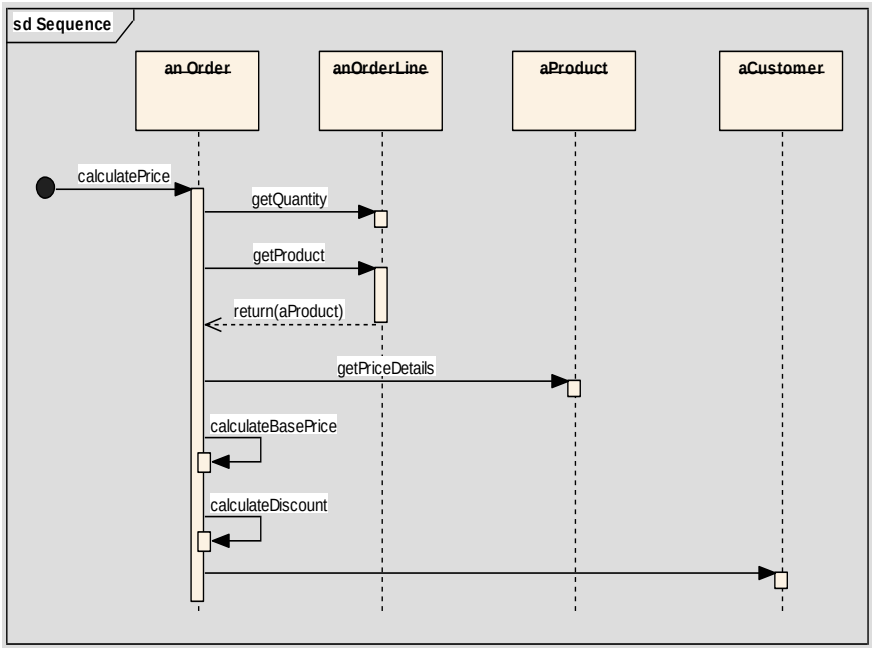
❑ 작성시기

- 유스케이스 다이어그램이 정의된 후 부터 프로그램 코딩 전 까지 수행
- 시퀀스 다이어그램은 다른 여러 다이어그램처럼 여러 번의 정제과정을 거침
- 분석단계에서는 비즈니스 관점에서의 객체
- 설계단계에서는 구현관점의 객체

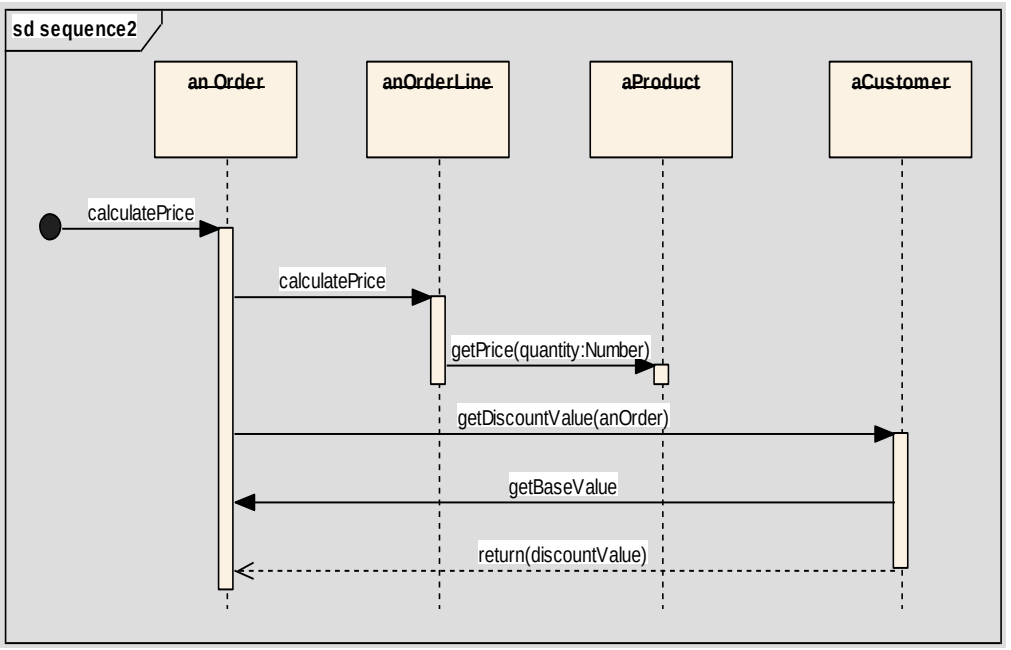
Copyright © 2009 Nextree Soft, All rights reserved

❑ 집중제어와 분산제어

- 집중 제어가 단순하고, 객체를 추적하는 감각이 보다 수월
- 좋은 설계는 변경에 대한 영향을 지역화 해야 한다는 목표달성을 위하여, 객체 전문가는 분산 제어를 선호



[집중 제어의 예]

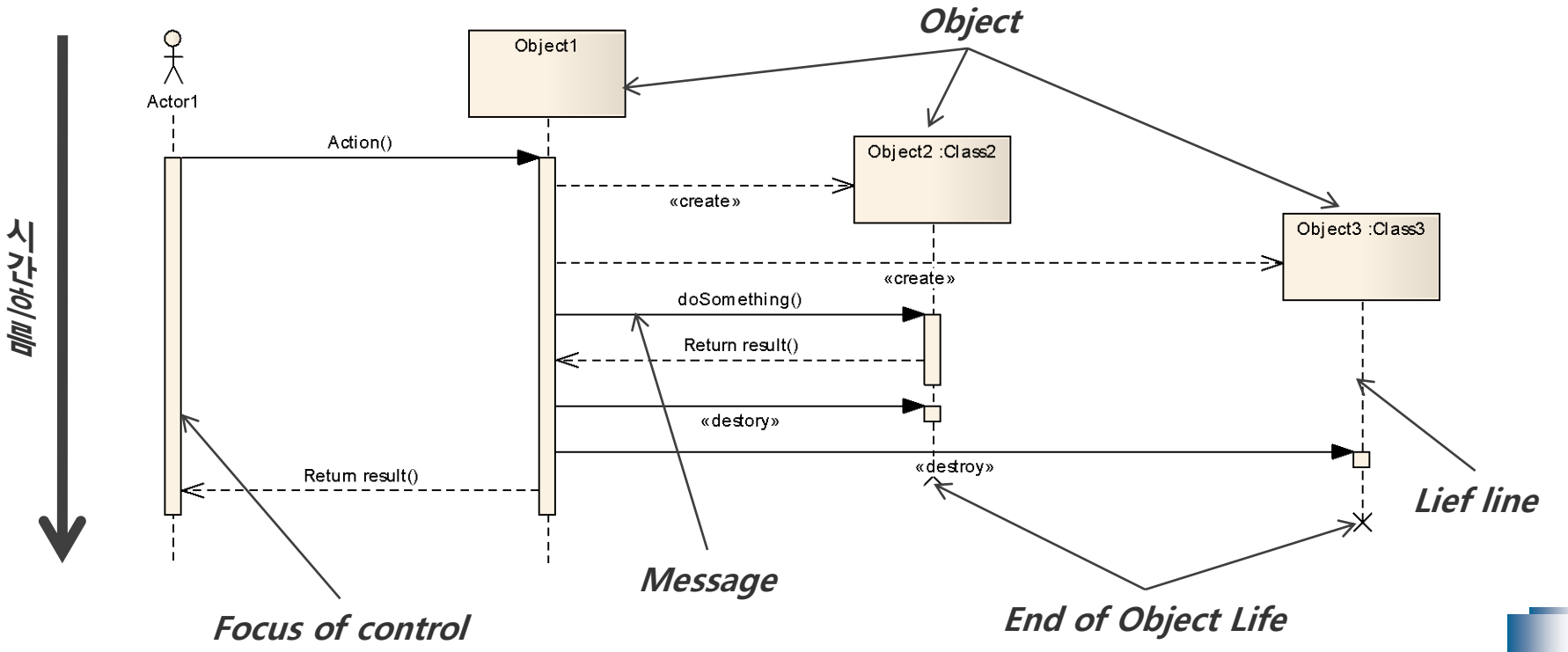


[분산 제어의 예]

Copyright © 2009 Nextree Soft, All rights reserved

❑ 시퀀스 다이어그램의 구성요소

- 요소: 액터*Actor*, 객체*Object*
- 관계: 메시지*Message*
- 기타: Life Line, Focus of control



Copyright © 2009 Nextree Soft, All rights reserved

❑ 액터*Actor*

- 시스템의 외부에 존재하면서 시스템과 교류 혹은 상호작용하는 것
- 시스템이 서비스를 해 주기를 요청하는 존재
- 시스템에게 정보를 제공하는 대상

액터의 예)

보험시스템	고객, 사원, 관리자, 결재자, 환자 등
오픈 마켓 시스템	회원, 구매자, 배달자, 관리자, 배송시스템 등
병원관리 시스템	의사, 간호사, 환자, 수납책임자 등

Copyright © 2009 Nextree Soft, All rights reserved

❑ 객체 *Object*

- 클래스의 인스턴스
- 시스템에서는 해당 클래스 타입으로 선언된 변수의 형태로 존재
- 생성되고 소멸되기까지의 생명주기 동안 다양한 상태의 변화가 존재

표기의 예)

홍길동 : 사원	: 사원 클래스의 객체 중 홍길동 객체를 의미
홍길동	: 홍길동 객체로 클래스는 정의가 없음
: 사원	: 사원 클래스의 일반 객체 (일반적인 객체를 지칭)
홍길동 : 사원 사번=20391 이름=홍길동 급여=4000 성별=남 직위=과장	: 사원 클래스의 홍길동 객체로서 객체 속성을 모두 표현

Copyright © 2009 Nextree Soft, All rights reserved

구성요소

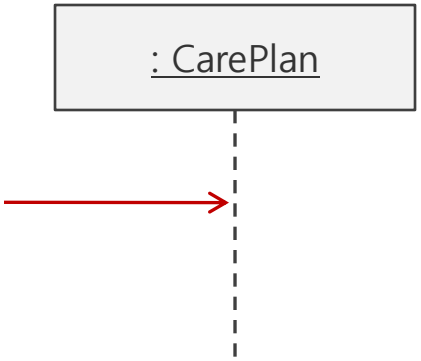
❑ 메시지 *Message*

- 객체지향 패러다임에서 객체와 객체가 통신하는 유일한 수단
- 객체 간의 협력작업과 상호작용
- 향후 클래스의 오퍼레이션으로 구현
- 메시지 유형
  - Flat flow of control
  - Nested flow of control
  - Asynchronous of control
  - Return flow

Copyright © 2009 Nextree Soft, All rights reserved

❑ Flat flow of control

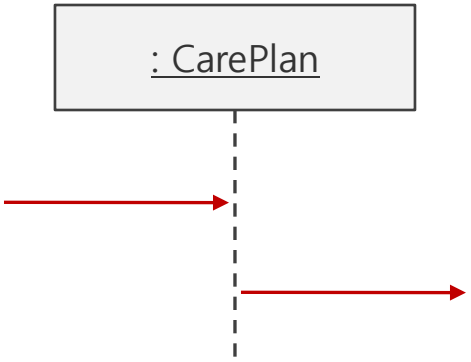
- 가장 일반적인 메시지 형태
- 객체에 메시지를 연결할 때 사용



Flat flow of control

❑ Nested flow of control

- Procedural call과 같은 의미로 사용
- 중첩*Nested*되는 경우 내부 메시지의 결과가 모두 돌아와야 가장 처음 시작한 메시지의 결과가 되돌려 보내짐
- 메시지 결과가 돌려지게 될 때까지 다음 처리를 진행하지 않는 동기화 메시지

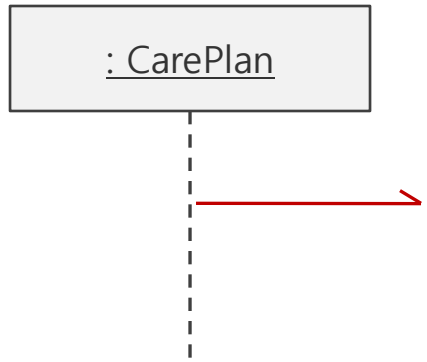


Nested flow of control

Copyright © 2009 Nextree Soft, All rights reserved

❑ Asynchronous of control

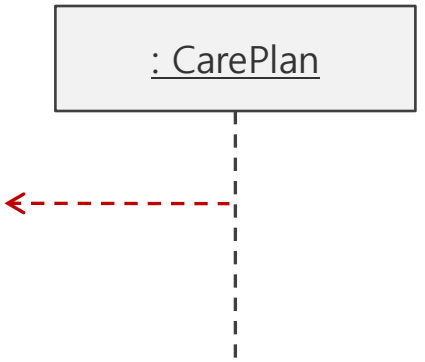
- 객체가 보낸 메시지의 결과를 기다리지 않고 다음 처리를 진행할 경우 사용



Asynchronous flow of control

❑ Return flow

- 메시지를 처리한 결과 리턴을 의미
- 필요한 경우에만 표현

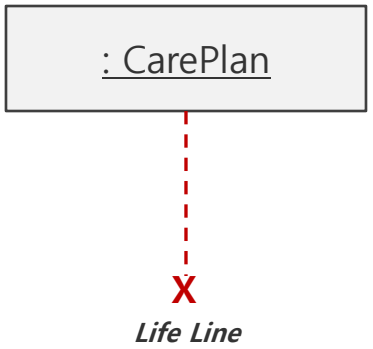


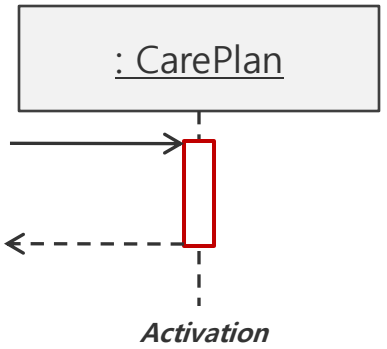
Return flow

Copyright © 2009 Nextree Soft, All rights reserved

구성요소

- **Life line**
  - 객체의 생존 기간을 의미
  - 객체에 붙은 수직방향의 점선으로 표기
  - 점선 X 표시가 있을 경우 그 시점이 객체의 소멸시점
- **Activation**
  - 객체가 활성화 되어 있는 기간
  - 객체가 외부 메시지를 받고 다른 객체에 보낸 메시지에 대한 Return Flow를 기다리는 시간을 의미
  - Life line 위에 긴 사각형으로 표시





Copyright © 2009 Nextree Soft, All rights reserved

작성 및 주의사항

❑ 시퀀스 다이어그램 작성 단계

- 작성 대상 선정
- 유스케이스의 액터를 파악
- 유스케이스를 실현하기 위해 참여할 클래스(객체)들을 선정
- 시간 순서에 따른 객체 간 메시지 정의
- 필요한 객체를 추가로 정의

단계	내용
대상 선정	유스케이스 다이어그램을 이용하여 시퀀스 다이어그램의 작성 대상을 선정한 후 하나의 유스케이스를 선택하고 유스케이스 정의서를 분석한다.
액터 파악	액터가 둘 이상일 경우라도 모두 좌측부터 액터를 위치시켜야 한다. 순서는 중요하지 않고 메시지 선이 적게 교차하도록 배치하는 것이 좋다.
클래스(객체) 선정	정의된 클래스 중에 유스케이스의 처리에 참여하는 것들을 식별하고 시퀀스 다이어그램에 위치시켜야 하지만 순서는 중요하지 않다.
메시지 정의	유스케이스를 실현하기 위해 필요한 객체들 간의 메시지를 정의하되, 시간 순서에 유의하도록 한다. (시간흐름은 위에서 아래로 흐름)
객체 추가	정의되지 않은 객체가 있으면 시퀀스 다이어그램에 추가하고 객체 사이의 메시지도 정의하여 추가해야 한다.

Copyright © 2009 Nextree Soft, All rights reserved

작성 및 주의사항

❑ 시퀀스 다이어그램 작성 시 주의사항

- 동일한 상호작용을 여러 시퀀스 다이어그램에 중복되게 작성하는 것을 피함
- 중복을 최소화하기 위해 UI별 시퀀스 다이어그램을 작성해도 됨
- 시퀀스 다이어그램의 가독성이 좋도록 적당한 코멘트를 사용
- 메시지 흐름은 액터로부터 시작되게 작성
- 클래스 다이어그램에 표기된 클래스명과 매팅 가능하도록 객체이름을 표기

시퀀스 다이어그램은 유스케이스 별로 하나씩 작성해야 한다.
하지만 경우에 따라서 하나의 유스케이스에 여러 개의 시퀀스 다이어그램으로
또는 여러 유스케이스에서 공통으로 사용하는 상호작용을 하나의
시퀀스 다이어그램으로 작성할 수 있다.

Copyright © 2009 Nextree Soft, All rights reserved

2. 커뮤니케이션 다이어그램

- ❑ 개요
- ❑ 구성요소
- ❑ 작성 및 주의사항

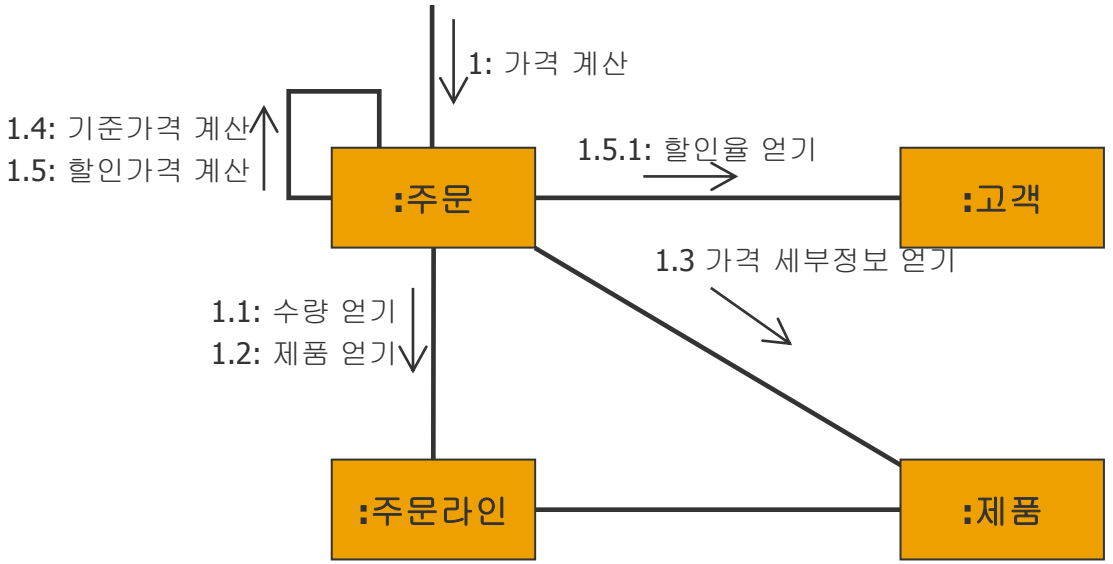
Copyright © 2009 Nextree Soft, All rights reserved

❑ 동적 모델링*Dynamic Modeling* 기법

- 객체와 객체 사이의 동적 상호관계를 정의한 모델

❑ 구조적인 측면 중시

- 해결해야 할 문제가 주어진 상황에서 그 문제를 해결하기 위해 필요한 객체를 정의하고 동적인 상호관계를 순서에 따라 정의
- 모델링 공간이 확보되어 같은 유형의 객체를 모아 놓아 모델링이 가능



Copyright © 2009 Nextree Soft, All rights reserved

개요

❑ 작성목적

- 객체 간의 동적 상호작용을 구조적 측면을 중시하여 모델링
- 객체 간 상호작용을 정의하는 과정에서 객체를 더욱 상세하게 정의
- 유스케이스를 실현*Realization*
- 프로그래밍 사양을 정의

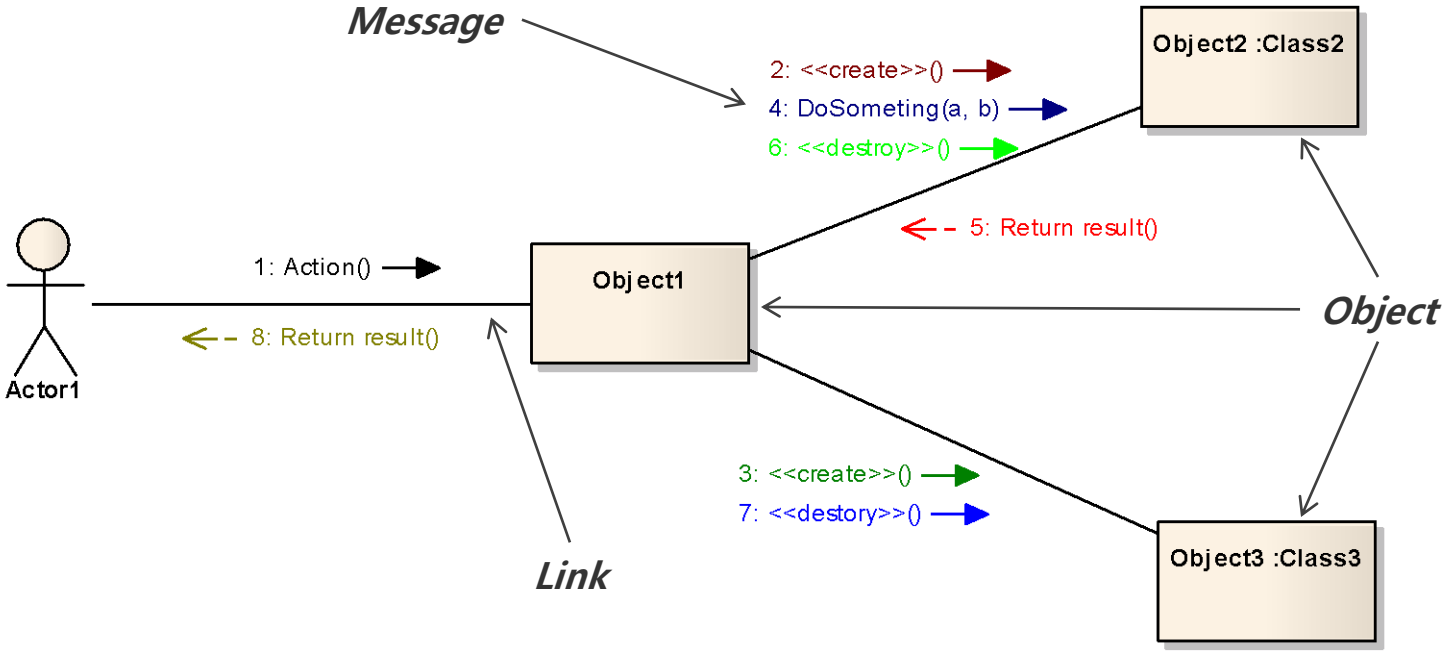
❑ 작성시기

- 시퀀스 다이어그램의 작성시기와 동일
- 분석단계에서는 비즈니스 관점에서 객체가 등장
- 설계 단계에서는 구현 관점의 객체들이 등장

Copyright © 2009 Nextree Soft, All rights reserved

❑ 커뮤니케이션 다이어그램 구성요소

- 요소: 액터*Actor*, 객체*Object*
- 관계: 메시지*Message*, 링크*Link*



Copyright © 2009 Nextree Soft, All rights reserved

구성요소 (2/7)

❑ 액터*Actor*

- 시스템의 외부에 존재하면서 시스템과 교류 혹은 상호작용하는 것
- 시스템이 서비스를 해 주기를 요청하는 존재
- 시스템에게 정보를 제공하는 대상

액터의 예)

보험시스템	고객, 사원, 관리자, 결재자, 환자 등
오픈 마켓 시스템	회원, 구매자, 배달자, 관리자, 배송시스템 등
병원관리 시스템	의사, 간호사, 환자, 수납책임자 등

Copyright © 2009 Nextree Soft, All rights reserved

❑ 객체 *Object*

- 클래스의 인스턴스
- 시스템에서는 해당 클래스 타입으로 선언된 변수의 형태로 존재
- 생성되고 소멸되기까지의 생명주기 동안 다양한 상태의 변화가 존재

표기의 예)

홍길동 : 사원	: 사원 클래스의 객체 중 홍길동 객체를 의미
홍길동	: 홍길동 객체로 클래스는 정의가 없음
: 사원	: 사원 클래스의 일반 객체 (일반적인 객체를 지칭)
홍길동 : 사원 사번=20391 이름=홍길동 급여=4000 성별=남 직위=과장	: 사원 클래스의 홍길동 객체로서 객체 속성을 모두 표현

Copyright © 2009 Nextree Soft, All rights reserved

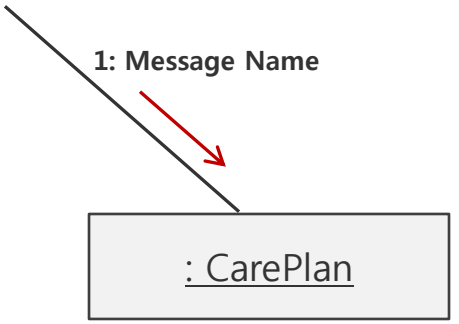
❑ 메시지 *Message*

- 객체지향 패러다임에서 객체와 객체가 통신하는 유일한 수단
- 객체 간의 협력작업과 상호작용
- 향후 클래스의 오퍼레이션으로 구현
- 메시지 유형
  - Flat flow of control
  - Nested flow of control
  - Asynchronous of control
  - Return flow

Copyright © 2009 Nextree Soft, All rights reserved

❑ Flat flow of control

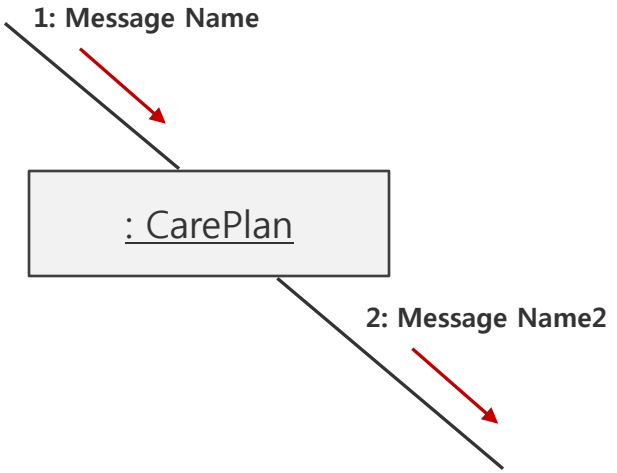
- 가장 일반적인 메시지 형태
- 객체에 메시지를 연결할 때 사용



Flat flow of control

❑ Nested flow of control

- Procedural call과 같은 의미로 사용
- 중첩*Nested*되는 경우 내부 메시지의 결과가 모두 돌아와야 가장 처음 시작한 메시지의 결과가 되돌려 보내짐
- 메시지 결과가 돌려지게 될 때까지 다음 처리를 진행하지 않는 동기화 메시지

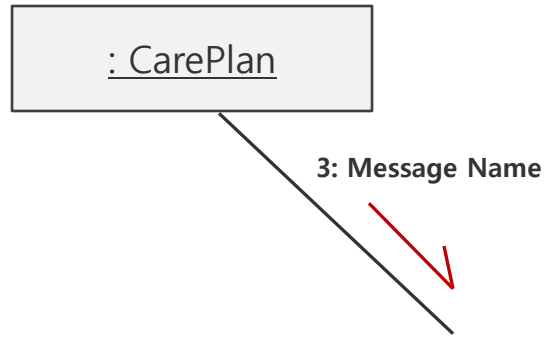


Nested flow of control

Copyright © 2009 Nextree Soft, All rights reserved

❑ Asynchronous of control

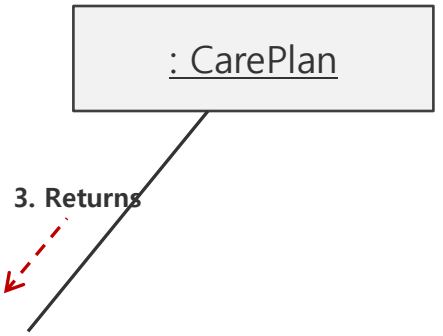
- 객체가 보낸 메시지의 결과를 기다리지 않고 다음 처리를 진행할 경우 사용



Asynchronous flow of control

❑ Return flow

- 메시지를 처리한 결과 리턴을 의미
- 필요한 경우에만 표현



Return flow

Copyright © 2009 Nextree Soft, All rights reserved

❑ 링크*Link*

- 객체와 객체 간의 연관관계를 표현
- 객체 간에 서로 통신을 하기 위해서는 링크로 연결
- 메시지는 링크를 따라 움직인다고 표현
- 링크에는 명명이 가능

Copyright © 2009 Nextree Soft, All rights reserved

작성 및 주의사항

❑ 커뮤니케이션 다이어그램 작성 단계

- 작성 대상을 선정
- 유스케이스의 액터를 파악
- 유스케이스를 실현하기 위해 참여할 클래스(객체)를 정의
- 시간 순서에 따른 객체 간 메시지를 정의
- 필요한 객체를 추가로 정의

단계	내용
작성 대상 선정	유스케이스 다이어그램을 이용하여 작성대상을 선정한 후 하나의 유스케이스를 선택하고, 유스케이스 정의서를 분석한다.
액터 파악	액터가 둘 이상일 경우라도 모두 좌측부터 액터를 위치시켜야 한다.
클래스(객체) 정의	정의된 클래스 중 유스케이스의 처리에 참여하는 것들을 식별한다.
메시지 정의	유스케이스 실현을 하기 위해 필요한 객체들 간의 메시지를 정의하되 시간 순서에 유의하도록 한다.
객체 추가	요구된 처리를 위해 필요하지만 아직 정의되지 않은 객체를 새로 정의하여 추가한다. 이때 추가된 객체 사이의 메시지도 정의하여 추가한다.

Copyright © 2009 Nextree Soft, All rights reserved

작성 및 주의사항

❑ 커뮤니케이션 다이어그램 작성 시 주의사항

- 유스케이스 별로 하나씩 작성
- 가독성이 좋도록 적당한 코멘트를 사용
- 메시지의 흐름은 액터로부터 시작되게 작성
- 클래스 다이어그램에 표기된 클래스 명과 매핑 가능하도록 객체 이름을 표기

동일한 상호작용을 여러 커뮤니케이션 다이어그램에 중복되게 작성하는 것을 피하고 중복을 최소화 시키기 위해서, UI별 커뮤니케이션 다이어그램을 작성해도 된다.

Copyright © 2009 Nextree Soft, All rights reserved

3. 그 외 다이어그램

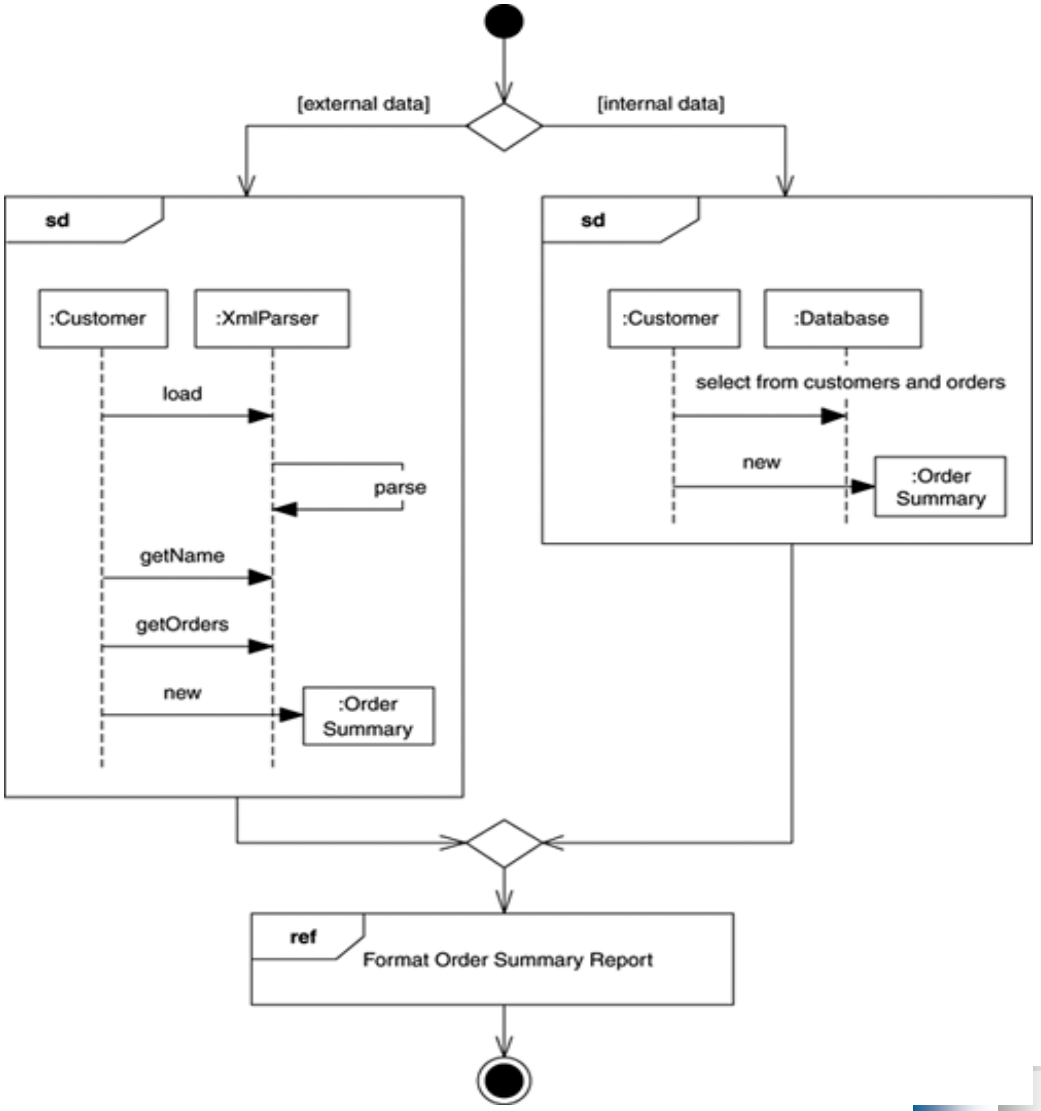
- ❑ 상호작용 개요 다이어그램
- ❑ 복합구조 다이어그램
- ❑ 타이밍 다이어그램

Copyright © 2009 Nextree Soft, All rights reserved

상호작용 개요(Interaction Overview) 다이어그램

❑ 상호작용 개요 다이어그램

- 액티비티 다이어그램과 시퀀스 다이어그램의 접목
- 액티비티 다이어그램에서 액티비티가 작은 시퀀스 다이어그램으로 바뀐 형태
- UML 2.X 에서 새로 추가됨
- 그림과 같이 선택 상황에서 따로 시퀀스 다이어그램을 그리지 않고 상호작용 프레임 내부에 표현

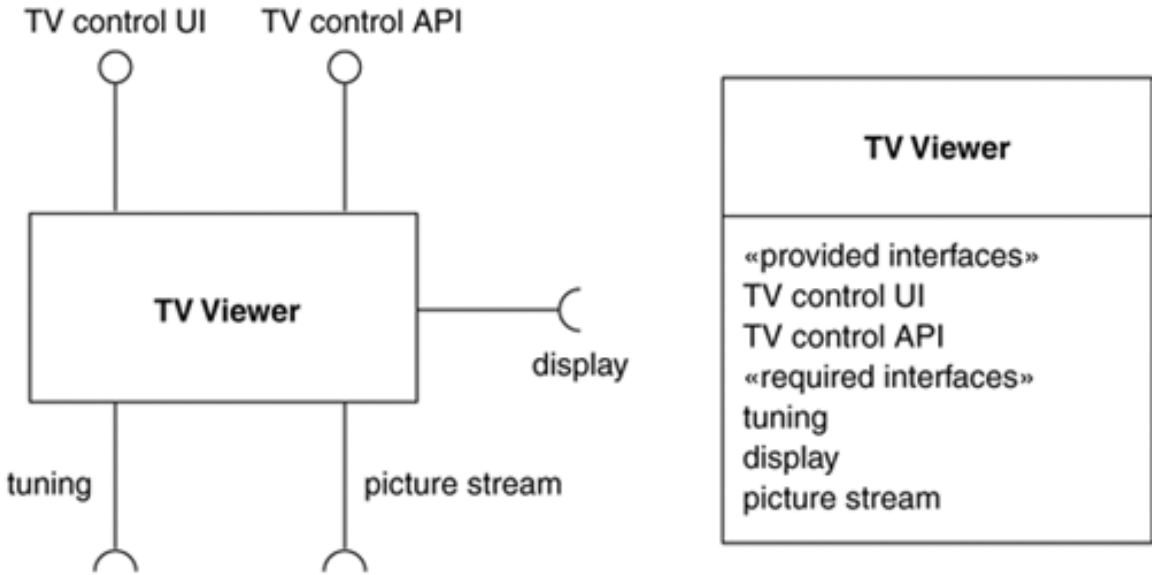


Copyright © 2009 Nextree Soft, All rights reserved

복합 구조(Composite Structures) 다이어그램(1/3)

❑ 복합 구조(1/2)

- UML 2.X에서 새롭게 제시되는 구조
  - 각 구성요소 들과 그 요소들이 어떻게 분리/연결되는지를 표현
  - 복잡한 객체를 부분들로 분해

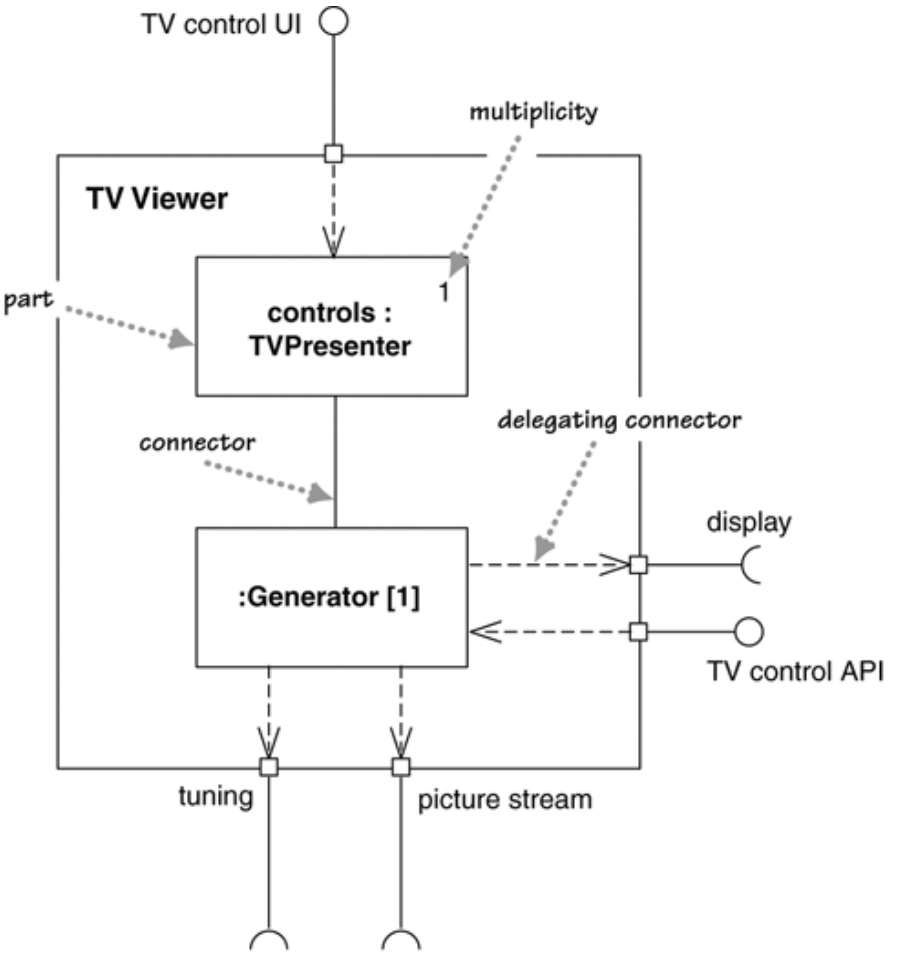


Copyright © 2009 Nextree Soft, All rights reserved

복합 구조(Composite Structures) 다이어그램(2/3)

❑ 복합 구조(2/2)

- 내부적으로 두 부분으로 분석되는 방법과 어느 부분들이 각각 인터페이스를 요청하고 지원하는지 보여줌
- 각 부분들은 name: class 형태로 이름 붙여짐: 밑줄을 그어 표기하지 않고, 굵게 표기됨



컴포넌트의 내부 뷰

Copyright © 2009 Nextree Soft, All rights reserved

복합 구조(Composite Structures) 다이어그램(3/3)

❑ 복합 구조 용도

- 복합 구조와 패키지 의 차이점
  - 패키지 : 컴파일-타임 그룹핑
  - 복합 구조 : 런타임 그룹핑
- 컴포넌트 분해에 사용
  - 컴포넌트 다이어그램에 주로 나타남

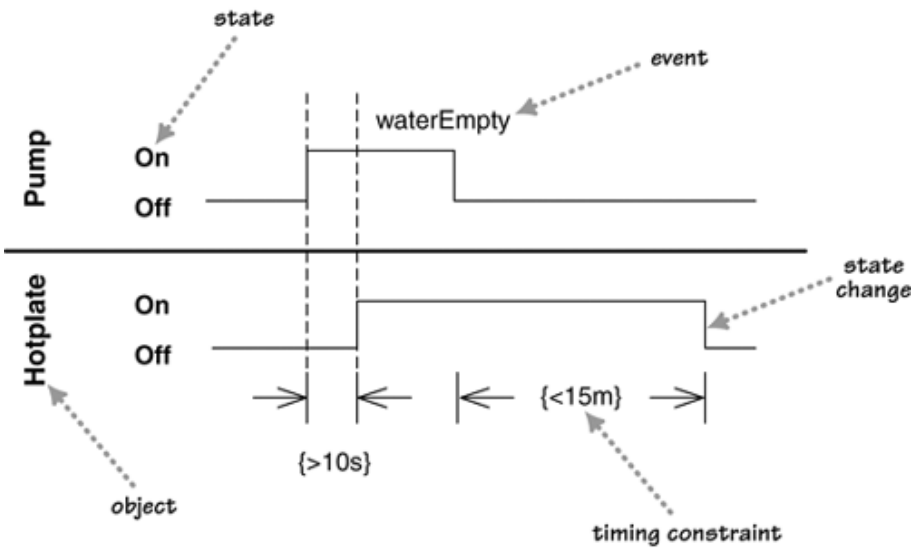
Copyright © 2009 Nextree Soft, All rights reserved

타이밍 다이어그램

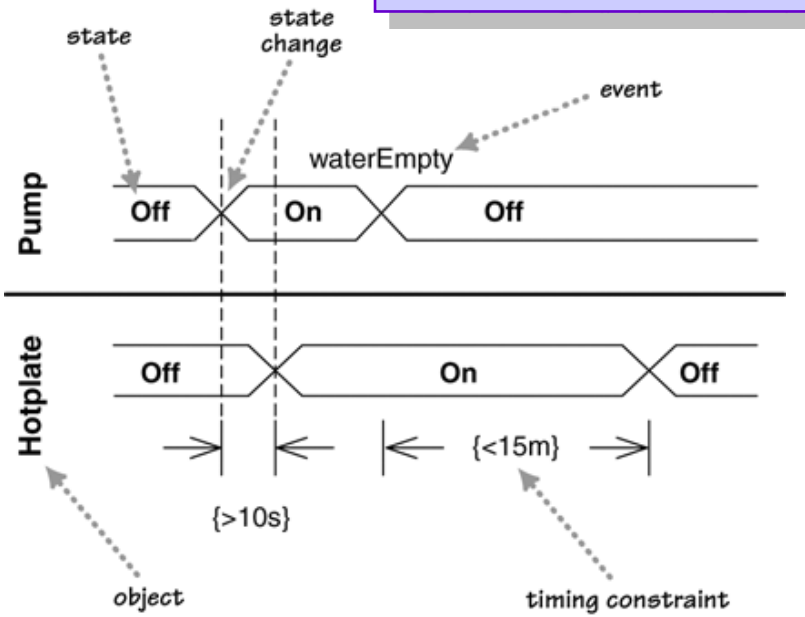
❑ 타이밍 다이어그램 사용

- UML 2.X 에서 신규 추가된 다이어그램
- 타이밍 다이어그램은 상호작용 다이어그램의 표현형태 중 하나로서 오브젝트들의 시간적 제약을 나타냄
- 타이밍 다이어그램은 아래와 같은 2가지 형식으로 표현 가능

선으로 상태의 변화를 나타냄



영역으로 상태의 변화를 나타냄



Copyright © 2009 Nextree Soft, All rights reserved