UML의 구성과 도구

- UML(Unified Modeling Language)
- UML의 구성 요소
- UML의 관계
- UML의 다이어그램
- UML 도구

UML(Unified Modeling Language)

- 모델링 과정(modeling process)과 모델링 언어(modeling language)를 제안
  - 모델링 과정 : 객체지향으로 분석하고 설계하는 프로세스
  - 모델링 언어 : 설계를 표현할 때 사용하는 그래픽 심볼
- UML탄생
  - OMT(Object Modeling Technique와, Booch, OOSE(Object-Oriented Software Engineering) 방법의 통합
  - Shlare/Mellor, Coad/Yourdon, Wirfs-Brock, Martin/Odell 방법의 영향
- 광범위 응용 분야에서 사용
  - 데이터베이스 설계 표현, 실시간 시스템 등

UML의 구성 요소

- **UML의 기본 구성요소**
  - 사물(Things) : UML은 기본 요소를 구성
  - 관계(Relationship) : 사물 간의 관계를 나타냄
  - 다이어그램(Diagram) : 사물과 관계를 도형으로 표현

- **사물(Things)**
  - **사물은 추상적 개념으로서 모델에서 가장 중요**
    1. Structural Things : 시스템의 구조를 표현하는 사물
    2. Behavioral Things : 시스템의 행위를 표현하는 사물
    3. Grouping Things : 개념을 그룹화하는 사물
    4. Annotation Things : 부가적으로 개념을 설명하는 사물

구조사물

- UML 모델의 명사형
- 모델의 정적인 부분이며, 개념적·물리적 요소 표현

클래스(Class)

- 동일한 속성, 오퍼레이션, 관계, 그리고 의미를 공유하는 객체를 기술한 것
- 클래스는 직사각형으로 표현
- 사각형 안에 이름, 속성, 오퍼레이션을 넣는다.

- 인터페이스(Interface)
  - 클래스 또는 컴포넌트의 서비스를 명세화하는 오퍼레이션을 모아놓은 것
  - 외부적으로 가시화되는 요소의 행동을 표현
  - 인터페이스는 특정 클래스나 컴포넌트의 전체 또는 일부분만의 행동을 나타냄
  - 인터페이스는 원으로 표현하고 인터페이스명을 아래에 표시하거나 클래스 형식으로 표현하고 스테레오 타입으로 <<interface>>를 사용
  - 인터페이스는 단독으로 나타나는 경우가 거의 없고, 인터페이스를 구현하는 클래스나 컴포넌트와 함께 나타냄

- 통신(Communication)
  - 교류(Interaction)를 정의하며, 서로 다른 요소와 역할들이 모여 있는 것
  - 행동적이고 구조적인 중요성을 가지며 하나의 클래스는 다수의 통신에 참여
  - 실선으로 된 사각형으로 표현하고 보통 이름을 안에 넣는다.

- 유스케이스(Use Case)
  - 유스케이스는 시스템이 수행하는 활동들을 순차적으로 기술
  - 액터(Actor, 행위자)에게 의미 있는 결과를 제공
  - 유스케이스는 모델에서 행동사물을 구조화하기 위해 사용되고 통신으로 실현
  - 유스케이스는 실선으로 된 타원으로 표현하고 보통 이름을 안에 넣는다.

- 활성 클래스(Active Class)
  - 활성의 2가지 형태 : 동작상태, 활동상태
  - 동작상태는
    - 객체에 있는 Operation을 호출하거나,
    - 객체에 Signal을 전송하고,
    - 객체가 생성되고 소멸될 때 사용되며 보통 순간적으로 작업이 실행
  - 동작상태의 경우 더 이상 분할되지 않는다.
  - 활동상태는
    - 하나의 Activity에 다른 제어 흐름을 가지는 활동 또는 동작상태로서 더 작은 활동상태나 동작상태로 분해 가능하다.

- 컴포넌트(Component)
  - 컴포넌트는 시스템의 물리적(눈에 보이는)이고 대체 가능한 부분
  - 컴포넌트는 일반적으로 클래스, 인터페이스, 그리고 통신과 같이 서로 다른 논리 요소를 물리적으로 패키지화한 것
  - 컴포넌트는 탭이 달린 직사각형으로 표시하며 이름을 안에 넣는다.

- 노드(Node)
  - 노드는 실행할 때에 존재하는 물리적 요소이다.
  - 컴포넌트가 노드에 존재할 수 있으며 노드에서 노드로 이동
  - 노드는 육면체로 표시하고 이름을 안에 넣는다.

행동사물

- 행동사물은 UML 모델의 동적 부분
- 모델의 동사로서, 시간과 공간에 따른 행동을 나타낸다.

교류(Interaction)

- 교류는 행동이며, 목적을 위해 객체들간에 주고받는 메시지로 구성
- 객체로 이루어진 공동체의 행동, 또는 개별 오퍼레이션의 행동을 교류로 명세화
- 교류의 다양한 요소 : 메시지, 활동 순서(메시지에 의해 호출되는 행동), 링크
- 메시지는 직선으로 나타내고, 항상 오퍼레이션 이름을 포함

- 상태 머신(State Machine)
  - 상태 머신은 상태의 순서를 지정하는 행동
  - 개별 클래스의 행동이나 여러 클래스들로 된 특정 행동을 하나의 상태로 지정
  - 상태 머신의 서로 다른 요소 : 상태 전이(상태에서 다른 상태로의 흐름), 사건(전이를 유발시키는 것), 활동(전이에 따른 응답)
  - 상태는 둥근 직사각형으로 표현하며 안에 이름을 넣고, 필요시 하위 상태를 포함

그룹사물

- 그룹사물 : UML 모델을 조직화 하며, 모델을 분해하고 담을 수 있는 상자를 의미
- 그룹사물 : 패키지

패키지(Package)

- 패키지는 요소를 그룹으로 묶는다.
- 구조사물, 행동사물, 그리고 다른 그룹사물까지도 하나의 패키지 내에 들어감
- 컴포넌트가 물리적인 것에 반해 패키지는 개념적(개발시에만 존재한다는 의미)
- 패키지는 탭이 달린 폴더로 표현하며 보통 이름만 쓰는데, 때때로 그 안에 있는 내용을 표현하기도 한다.

주해사물

- 주해사물은 UML 모델을 설명하는 부분
- 주석으로서 모델에 있는 어떠한 요소에 대해서 설명하고, 비고를 표시할 때 사용

노트(Note)

- 노트는 하나의 요소 또는 공동체에 첨부되는 제약과 주석을 나타내기 위해 사용
- 노트는 접힌 직사각형으로 표현되며, 문자와 그래픽을 함께 나타낼 수 있다.

UML의 관계

- **관계(Relationship)**
  - 의존(Dependency)
  - 연관(Association)
  - 일반화(Generalization)
  - 실체화(Realization)

- 의존
  - 의존은 두 사물간의 의미적 관계
  - 한 사물의 명세서가 바뀌면 그것을 사용하는 다른 사물에게 영향을 끼치는 것 (그 반대도 반드시 성립하는 것은 아니다.)
  - 의존은 점선으로 된 직선을 사용하고, 의존하고 있는 사물을 향하고 있다.
  - 의존은 한 클래스가 다른 클래스를 오퍼레이션의 매개변수로 사용하는 경우에 주로 나타난다.

- 연관
  - 연관은 구조적 관계로서 어느 한 사물 객체가 다른 사물 객체와 연결을 의미
  - 두 클래스가 서로 연결되어 연관이 있다면, 한 쪽 객체에서 다른 객체로 옮겨갈 수 있으며 그 반대도 가능
  - 한 연관의 양쪽 끝이 같은 클래스를 향해 원을 그리며 순환하는 것도 가능
    - 한 클래스의 객체가 있을 때 같은 클래스의 다른 객체에게 연결할 수 있다는 의미
  - 쌍방연관 : 정확하게 두 클래스를 연결하는 연관
  - 다수연관 : 2개 이상의 클래스를 연결하는 연관
  - 연관은 클래스들을 연결하는 실선으로 표현
  - 기본 형태에 연관의 이름, 역할, 다중성, 그리고 집합연관 등을 추가

① 이름

- 연관은 이름을 가질 수 있다. 관계의 의미를 설명하기 위해 이름을 사용
- 의미의 모호함을 없애기 위해 이름에 방향성을 추가할 수 있는데,
  - 이름이 읽히기를 원하는 방향으로 방향 삼각형을 표기

② 역할

- 클래스가 연관에 참여하면 그것이 수행해야 하는 특별한 역할을 가진다.
- 클래스가 수행하는 역할을 명시적으로 이름 지을 수 있다.
- 클래스 옆에 원하는 역할을 써줌으로써 연관관계 내에서의 역할을 표시

③ 다중성

- 연관은 객체간 구조적 관계를 표현
- 한 연관에 참여하는 하나의 객체에 대해 상대 쪽에는 몇 개의 객체가 연결되어 있는지를 밝히는 것이 중요한 때가 있다.
- ‘몇 개’를 일컬어 연관 역할에 대한 다중성이라 한다.
- 다중성은 하나(1), 제로 혹은 하나(0..1), 다수(0..*), 하나이상(1..*) 등으로 표현

④ 집합연관

- 순수한 연관은 두 동료 클래스 사이의 구조적 관계를 표현하는데, 두 클래스는 개념적으로 같은 수준에 위치
- ‘전체/부분’관계를 모델링하려고 할 때가 있다.
  - 한 클래스는 더 큰 것(‘전체’)을 대표하고 그것은 더 작은 것들(‘부분’)로 구성
  - 이러한 관계를 집합연관 (‘has-a’관계)
- 전체 쪽 한 객체가 부분 쪽 객체들을 소유
- 연관에 속이 빈 다이아몬드를 전체 쪽에 추가하여 표현

- 일반화
  - 일반화는 일반화된 사물과 좀 더 특수화된 사물 사이의 관계(‘is-a-kind-of’관계)
  - 일반화는 자식 객체가 부모 객체가 사용되는 어느 곳에서나 사용될 수 있다는 것을 의미(그 반대는 성립하지 않는다.)
    - 자식 객체는 부모 객체를 대신할 수 있다는 것
  - 일반화는 실선으로서 속이 빈 큰 화살표로 부모를 향해 그린다.
  - 부모/자식 관계를 표현하려면 일반화를 사용
  - 일반화는 주로 클래스와 인터페이스 사이에서 상속관계를 보여주기 위해 사용
  - UML에서는 다양한 사물들 사이에서도 일반화를 사용 (패키지)

실체화

- 실체화
  - 객체들 사이의 의미적 관계로서 한 객체가 다른 객체의 계약을 지정
  - 의존과 일반화의 혼합이며, 그 표기법도 의존과 일반화에서 가져왔다.
  - 인터페이스와 인터페이스에 오퍼레이션이나 서비스를 제공하는 클래스나 컴포넌트
  - 사이의 관계를 지정하기 위해서 사용

- 인터페이스
  - 오퍼레이션의 모음으로서 클래스나 컴포넌트의 서비스를 명세화하기 위해 사용
  - 속성을 갖지 않음
  - 인터페이스의 모델링 방법은 클래스의 모델링 방법과 비슷
  - 인터페이스는 오퍼레이션의 집합이기 때문에 속성을 가지지 않는다

- 클래스를 나타낼 때 속성 생략 가능

- 속성을 생략한 클래스와 인터페이스는 모양과 형태가 같다.
  - 구분하는 방법
    - » 키워드를 사용 : 인터페이스의 이름 위에다가 <<interface>>라고 써주면 된다.
    - » 인터페이스의 이름 첫머리에 'I'를 붙여서 클래스와 구분

UML의 다이어그램

- 다이어그램(Diagram)
  1. Class Diagram
  2. Object Diagram
  3. Component Diagram
  4. Deployment Diagram
  5. Usecase Diagram
  6. Sequence Diagram
  7. Communication Diagram
  8. State Diagram
  9. Activity Diagram
  10. Robustness Diagram
  11. Composite Structure Diagram

- 클래스 다이어그램
  - 클래스, 인터페이스, 통신, 그리고 이들의 관계들을 나타내며, 객체지향 시스템 모델링에서 가장 공통적으로 쓰이는 다이어그램
  - 시스템의 정적 설계 뷰를 다루며, 활성 클래스를 갖는 클래스 다이어그램은 시스템의 정적 프로세스 뷰를 다룬다.

- 객체 다이어그램
  - 객체와 객체들 사이의 관계를 나타낸다.
  - 특정 시점의 객체들의 구조적 상태를 표현
  - 클래스 다이어그램처럼 시스템의 정적 설계 뷰와 정적 프로세스 뷰를 다루지만 실제 사례나 프로토타입 사례의 시각에서 표현

- 컴포넌트 다이어그램
  - 컴포넌트 사이의 구성과 의존을 나타내며, 시스템의 정적 구현 뷰를 다룬다.
  - 클래스 다이어그램과 관련이 있는데, 일반적으로 하나의 컴포넌트는 클래스 다이어그램에 있는 하나 또는 그 이상의 클래스, 인터페이스, 통신들과 대응

- 배치 다이어그램
  - 실행 시 처리하는 노드와 그 노드에 있는 컴포넌트들의 구성을 나타내며, 시스템의 정적 배치 뷰를 나타낸다.
  - 컴포넌트 다이어그램과 관련이 있는데, 일반적으로 하나의 노드는 컴포넌트 다이어그램 안에 있는 하나 또는 그 이상의 컴포넌트를 수용하기 때문
- 유스케이스 다이어그램
  - 유스케이스(시스템을 사용하는 다양한 경우)와 행위자(외부 행위자)의 관계를 구조적으로 나타낸다.
  - 시스템의 정적 유스케이스 뷰를 다룬다.
  - 시스템 행동을 조직화하고 모델링하는 데 특히 중요
- 상태 다이어그램
  - 상태 머신을 나타내며, 시스템의 내부 전이를 표현
  - 상태 머신은 상태(State), 전이(Transition), 이벤트(Event), 활동(Activity)으로 구성
  - 시스템의 동적 뷰를 다룬다.
  - 인터페이스, 클래스, 또는 통신의 행동을 모델링하는 데 중요
  - 이벤트에 따라 순차적으로 일어나는 객체 행동에 중점을 두기 때문에 빠른 반응이 필요한 시스템을 모델링할 때 유용

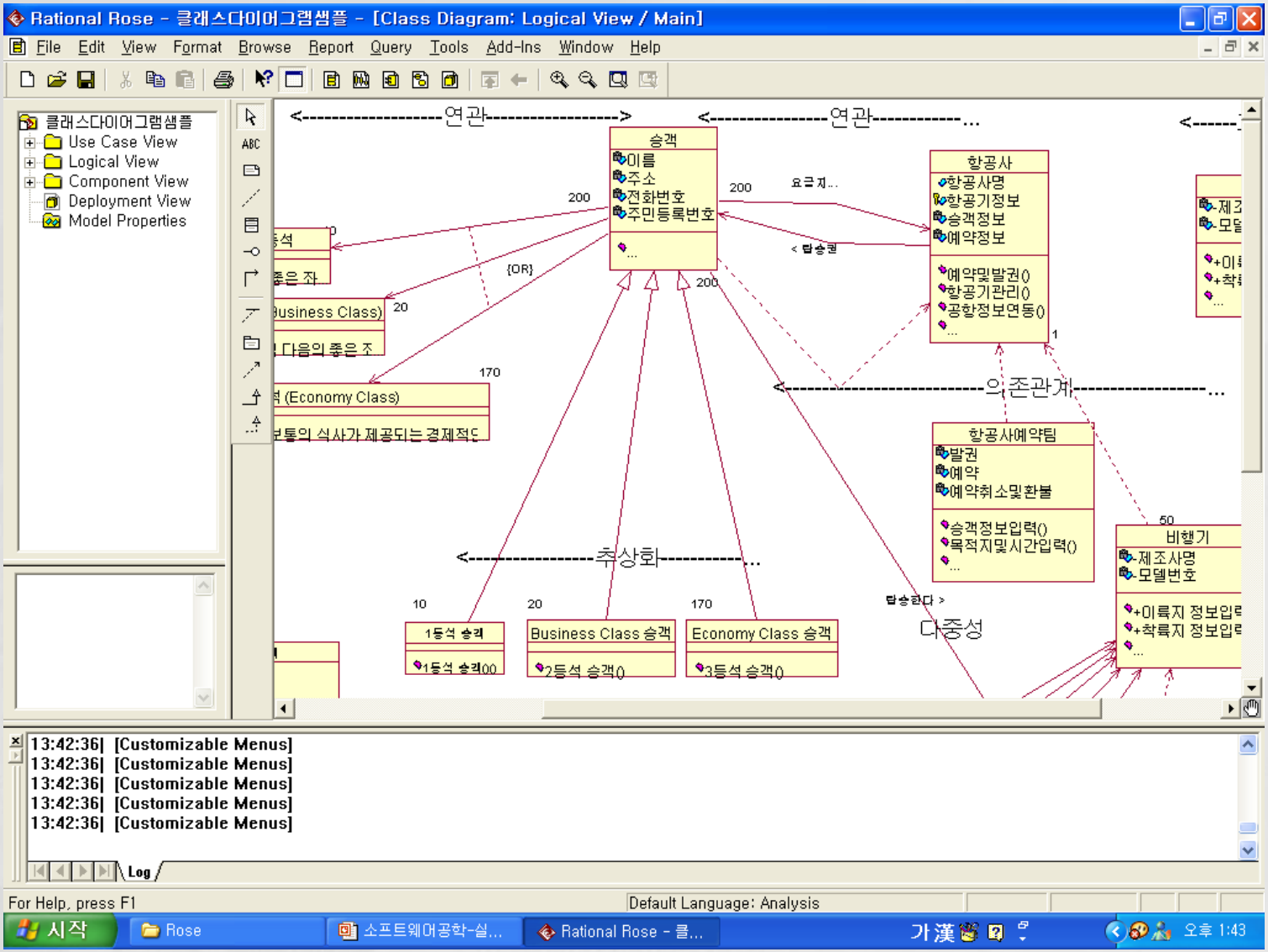
- 순차 다이어그램과 통신 다이어그램
  - 순차 다이어그램과 통신 다이어그램은 교류도(Interaction Diagram)의 한 종류
  - 교류도는
    - 객체와 관계, 그리고 이들 사이에 보낼 수 있는 메시지로 구성되는 교류
    - 교류도는 시스템의 동적 뷰를 다룬다.
  - 순차 다이어그램은
    - 메시지의 시간적 순서를 강조하는 교류도이며, 통신 다이어그램은 메시지를 주고받는 객체의 구조적 구성을 강조하는 교류도
    - 시스템 외부 이벤트를 처리하기 위하여 시스템 내부 객체간에 주고받는 동적 메시지를 시간의 흐름에 따라 표현한 것
  - 통신 다이어그램은
    - 순차 다이어그램과 동일한 내용을 객체 상호관계의 관점에서 표현한 것
  - 순차와 통신 다이어그램은 동일한 구조로 되어 있어서 서로 변환이 가능
- 활동 다이어그램
  - 시스템 내부에 있는 활동의 흐름
  - 시스템의 동적 뷰를 다루며, 시스템 기능을 모델링 할 때 중요하고, 객체간의 제어흐름에 중점을 둔다.

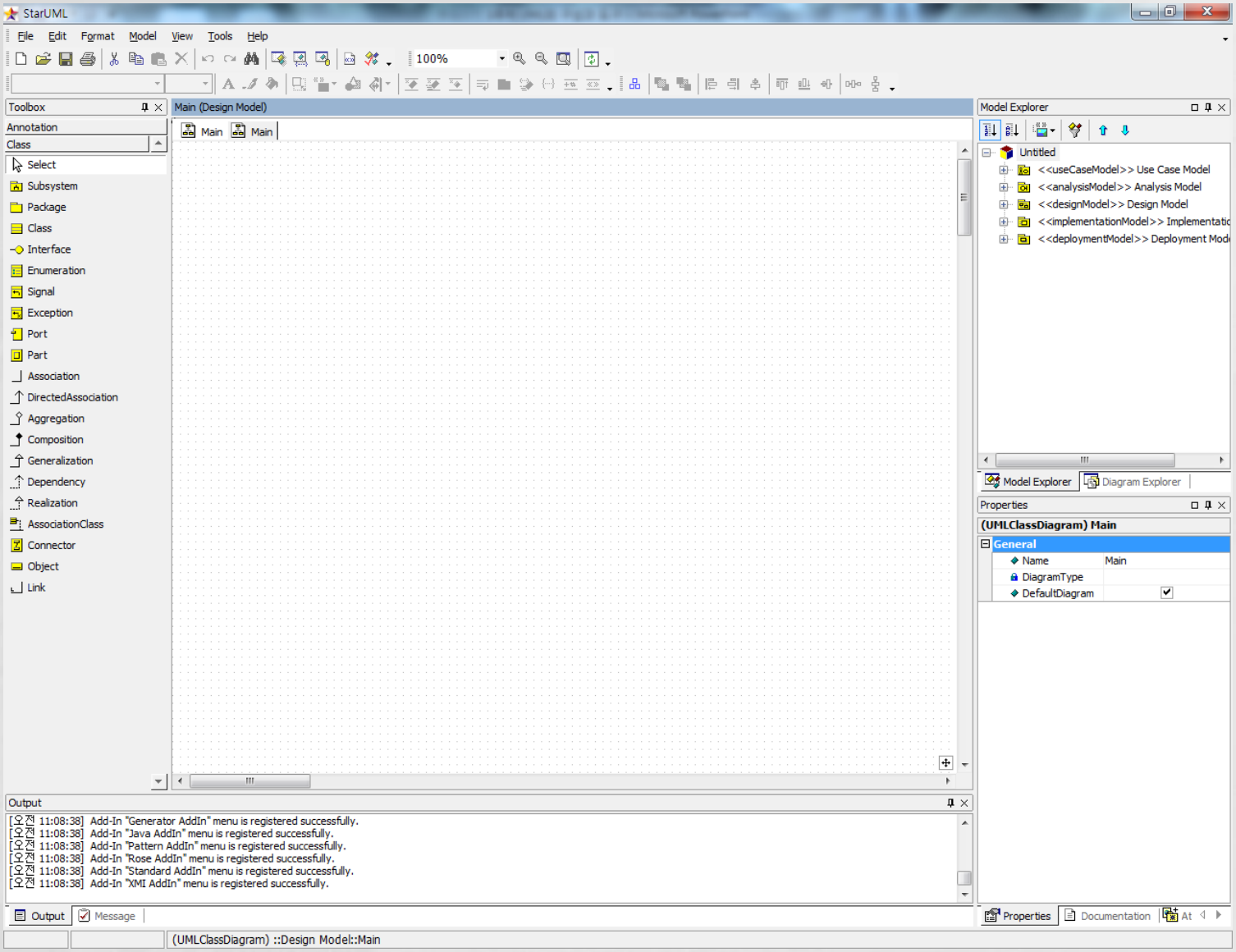
UML 뷰

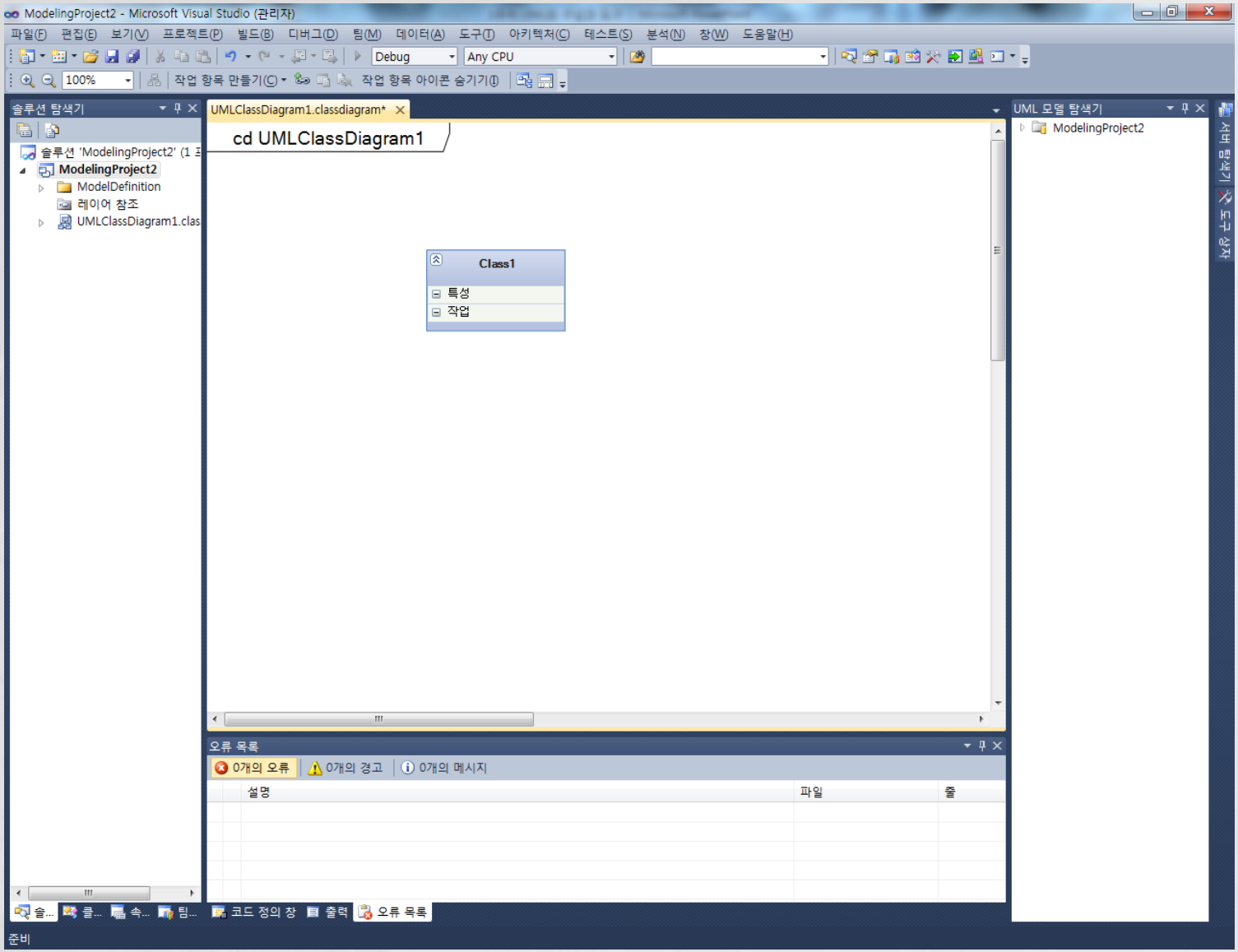
- 큰 규모의 건축물을 만들 때는 설계라는 과정이 반드시 필요
- 소프트웨어도 매우 복잡하기 때문에 몇 가지 관점에서 기술할 필요가 있다.
- 객체지향 전문가들은 5가지 중요한 관점(View)을 정의
- 소프트웨어의 5가지 관점을 표현하기 위하여UML 다이어그램을 이용

① 유스케이스 뷰(Use Case View)
② 설계 뷰(Design View)
③ 프로세스 뷰(Process View)
④ 구현 뷰(Implementation View)
⑤ 배치 뷰(Development View)

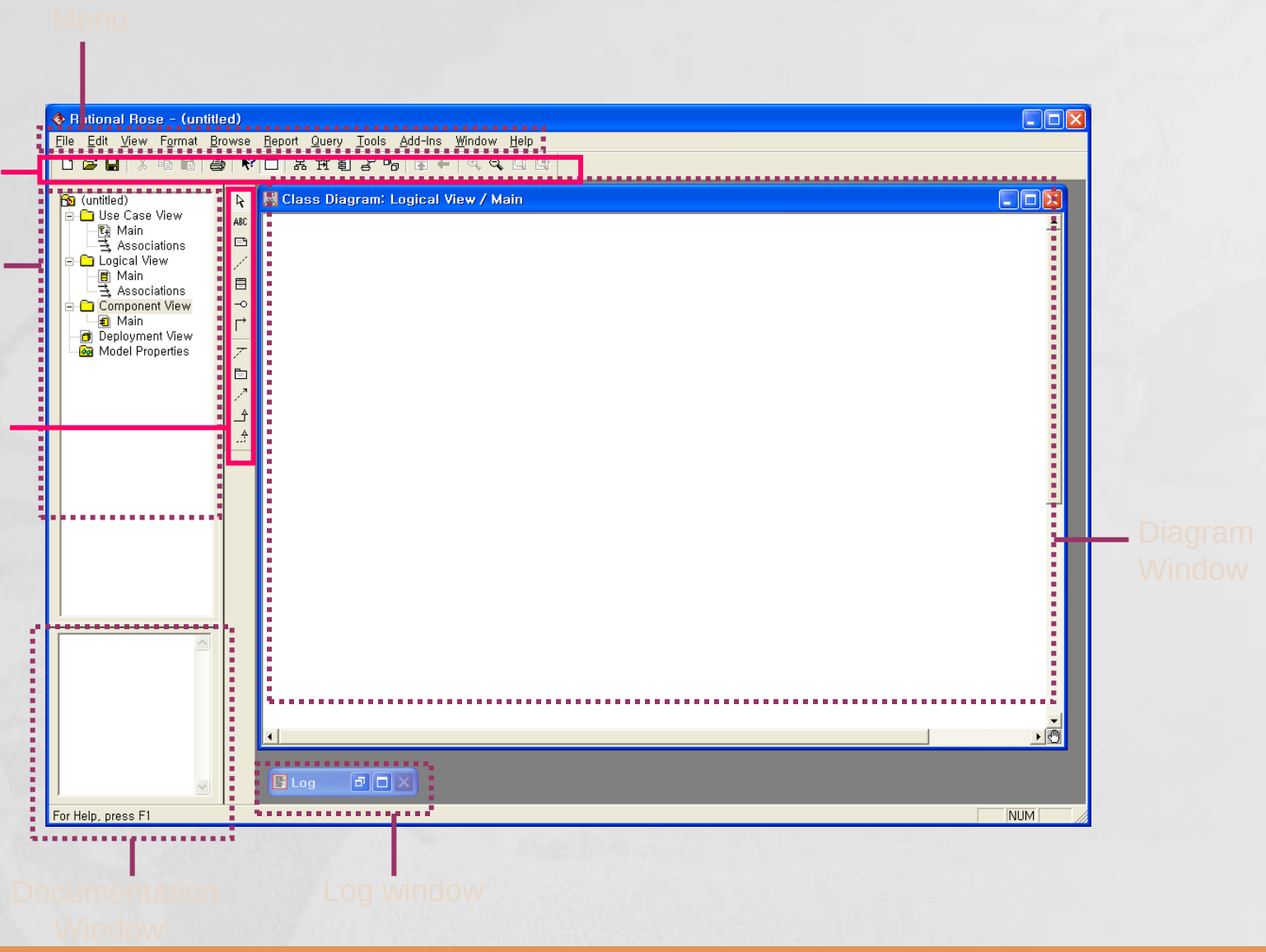
UML 도구


Rational Rose - 클래스다이어그램샘플 - [Class Diagram: Logical View / Main]
File Edit View Format Browse Report Query Tools Add-Ins Window Help
클래스다이어그램샘플
Use Case View
Logical View
Component View
Deployment View
Model Properties
연관
승객
이름
주소
전화번호
주민등록번호
항공사
항공사명
항공기정보
승객정보
예약정보
예약및발권()
항공기관리()
공항정보연동()
의존관계
항공사예약팀
발권
예약
예약취소및환불
승객정보입력()
목적지및시간입력()
비행기
제조사명
모델번호
추상화
1등석 승객
Business Class 승객
Economy Class 승객
다중성
13:42:36| [Customizable Menus]
13:42:36| [Customizable Menus]
13:42:36| [Customizable Menus]
13:42:36| [Customizable Menus]
13:42:36| [Customizable Menus]
Log
For Help, press F1
Default Language: Analysis
시작
Rose
소프트웨어공학-실...
Rational Rose - 클...
오후 1:43

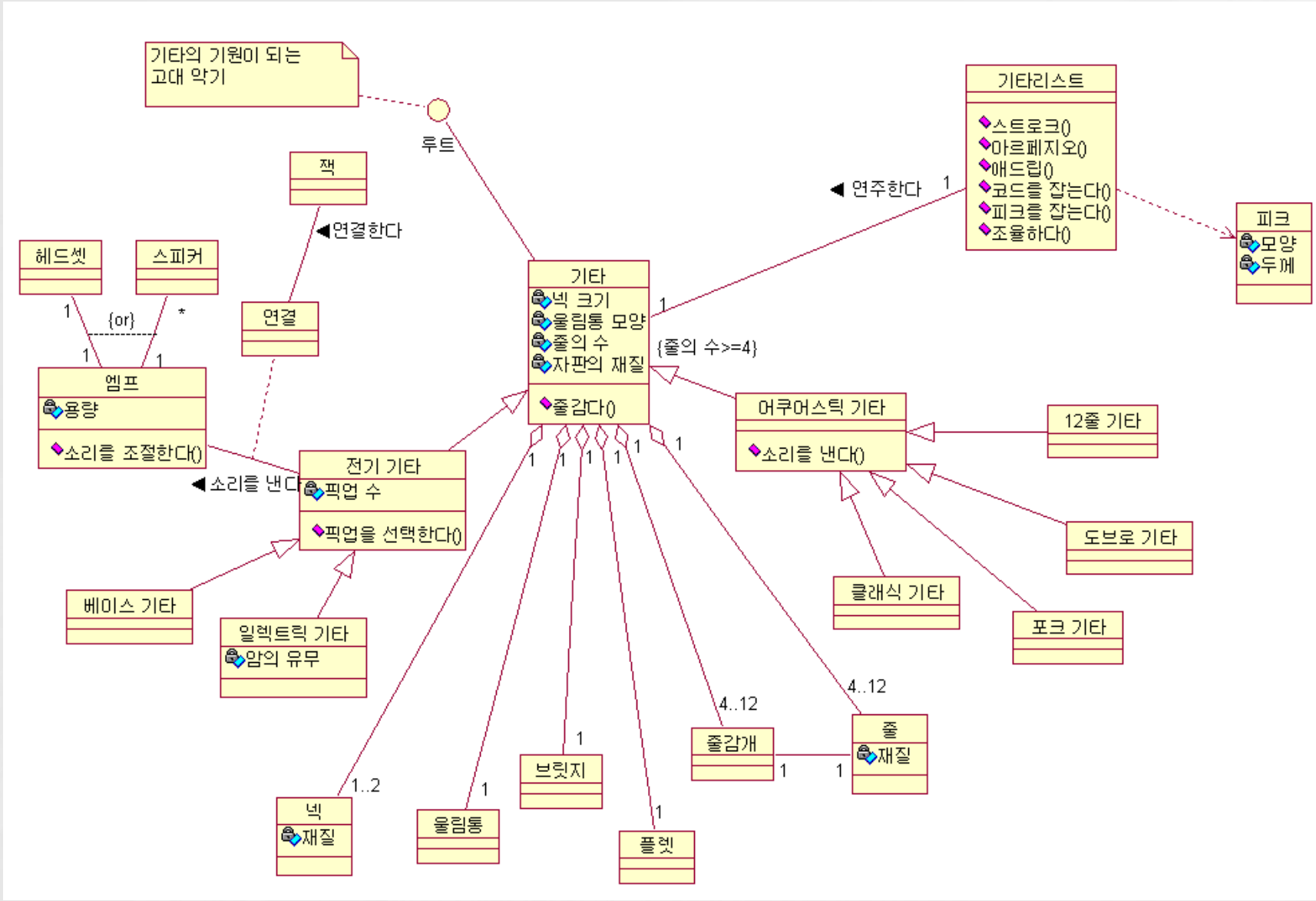

StarUML
File Edit Format Model View Tools Help
100%
Toolbox
Annotation
Class
Select
Subsystem
Package
Class
Interface
Enumeration
Signal
Exception
Port
Part
Association
DirectedAssociation
Aggregation
Composition
Generalization
Dependency
Realization
AssociationClass
Connector
Object
Link
Main (Design Model)
Main
Main
Model Explorer
Untitled
<<useCaseModel>> Use Case Model
<<analysisModel>> Analysis Model
<<designModel>> Design Model
<<implementationModel>> Implementatic
<<deploymentModel>> Deployment Mod
Model Explorer
Diagram Explorer
Properties
(UMLClassDiagram) Main
General
Name
Main
DiagramType
DefaultDiagram
Output
11:08:38 Add-In "Generator AddIn" menu is registered successfully.
11:08:38 Add-In "Java AddIn" menu is registered successfully.
11:08:38 Add-In "Pattern AddIn" menu is registered successfully.
11:08:38 Add-In "Rose AddIn" menu is registered successfully.
11:08:38 Add-In "Standard AddIn" menu is registered successfully.
11:08:38 Add-In "XMI AddIn" menu is registered successfully.
Output
Message
Properties
Documentation
(UMLClassDiagram) ::Design Model::Main


ModelingProject2 - Microsoft Visual Studio (관리자)
파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 데이터(A) 도구(T) 아키텍처(C) 테스트(S) 분석(N) 창(W) 도움말(H)
Debug
Any CPU
100%
작업 항목 만들기(C)
작업 항목 아이콘 숨기기(I)
솔루션 탐색기
솔루션 'ModelingProject2' (1 프
ModelingProject2
ModelDefinition
레이어 참조
UMLClassDiagram1.clas
UMLClassDiagram1.classdiagram*
cd UMLClassDiagram1
Class1
특성
작업
UML 모델 탐색기
ModelingProject2
서버 탐색기
도구 상자
오류 목록
0개의 오류
0개의 경고
0개의 메시지
설명
파일
줄
솔...
클...
속...
팀...
코드 정의 창
출력
오류 목록
준비

Rational Rose 구성



클래스 다이어그램-예제소개



연습문제

- UML의 탄생과정과 배경을 설명하시오.
- UML의 구성 요소별 역할을 설명하시오.
- UML의 다이어그램의 예제를 찾아보시오.
- UML 도구를 조사하여 도구별 장단점을 설명해 보시오.