Lecture 5

Introduction to UML

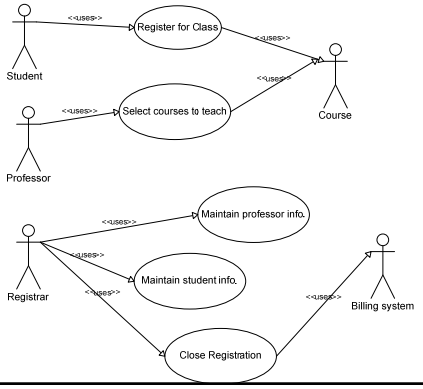
Today's Topic

- What is UML
- Why use UML?
- UML Diagrams
  - Static
    - Use case, Class, Object
    - Deployment, Component (Physical Diagrams)
  - Dynamic
    - Sequence, Collaboration (Interaction Diagrams)
    - Activity, State

What is UML ?

- UML: Universal Modeling Language
- A standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.
- UML uses mostly graphical notations.
- Current version: 2.3 (2.4 beta2)
- Complete spec can be freely downloaded at www.uml.org

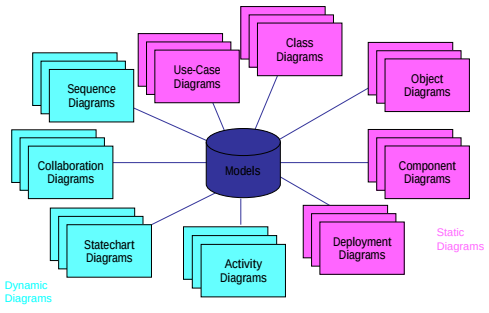
Example: University Course Registration System



Why Use UML

- UML is a technique to manage the complexity of object-oriented systems as they increase in scope and scale.
- Programming language independent
- Communication: clearer than natural language, provides a level of precision, but avoids details
- Supports iterative development (i.e., spiral model)
  - Supports both high level requirements/design in early spirals and detailed requirements/design later
- Provides a blueprint for programmers to implement

UML Diagrams



Source: Rational Software

Using multiple views for a single model.

Reference: UML Distilled: A Brief Guide to the Standard Object Modeling Language, Third Edition, by Martin Fowler (2003)

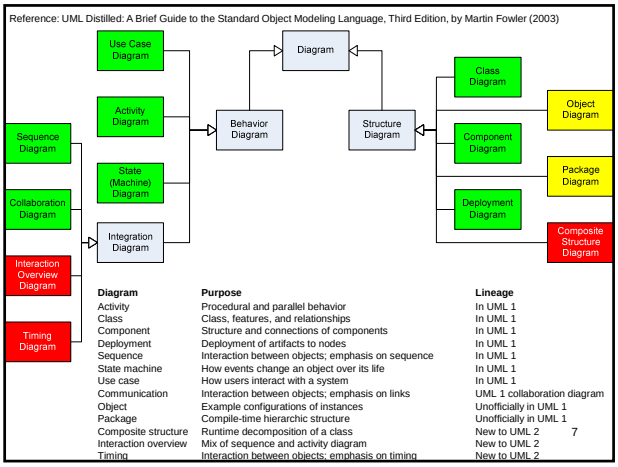


Diagram	Purpose	Lineage
Activity	Procedural and parallel behavior	In UML 1
Class	Class, features, and relationships	In UML 1
Component	Structure and connections of components	In UML 1
Deployment	Deployment of artifacts to nodes	In UML 1
Sequence	Interaction between objects; emphasis on sequence	In UML 1
State machine	How events change an object over its life	In UML 1
Use case	How users interact with a system	In UML 1
Communication	Interaction between objects; emphasis on links	UML 1 collaboration diagram
Object	Example configurations of instances	Unofficially in UML 1
Package	Compile-time hierarchic structure	Unofficially in UML 1
Composite structure	Runtime decomposition of a class	New to UML 2
Interaction overview	Mix of sequence and activity diagram	New to UML 2
Timing	Interaction between objects; emphasis on timing	New to UML 2

7

Static Model Diagrams

- Use case diagram
  - Drawings and structured descriptions of steps in workflows
- Class diagram
  - Internal structure of system, extension of entity-relationship diagram (ER diagram for database, to be covered).
  - Three elements in each entity: name, attributes, methods
- Deployment diagram
  - Physical components: processors, workstations, network
- Component diagram
  - High level model of physical software architecture
  - Consists of modules, which are grouped in packages
  - Packages contains definition of group of classes (entities, methods)

8

Source: MIT 1.264 lecture notes

Dynamic Model Diagrams

While static models are done for the system as a whole, dynamic models are done only for key components

- Activity Diagram
- State diagram
  - Specifies behavior of an object (entity)
- Sequence diagram (or ladder diagram)
  - Shows details of scenario and messages that flow between objects over time
  - Heavily used in standards
- Collaboration diagram
  - Shows flow of messages as a graph.

9

Source: MIT 1.264 lecture notes

UML Modeling Tools

- There are several free and commercial software for creating UML diagrams
  - Microsoft Visio
    - http://office.microsoft.com/en-us/FX010857981033.aspx
  - Rational Rose (now with IBM)
    - http://www-306.ibm.com/software/rational/
  - Poseidon for UML
    - http://www.gentleware.com/
    - The user guide is a good reference !

10

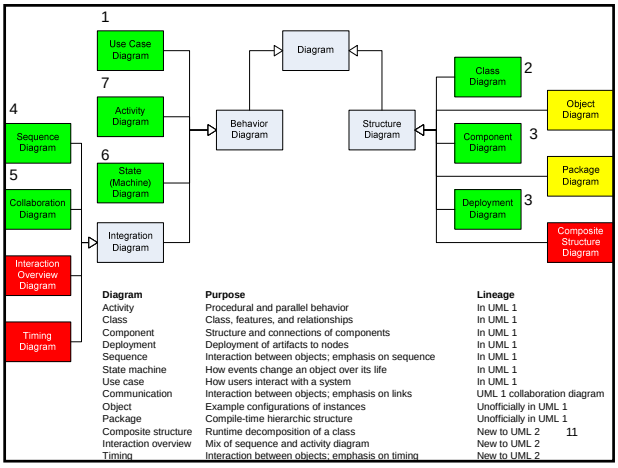


Diagram	Purpose	Lineage
Activity	Procedural and parallel behavior	In UML 1
Class	Class, features, and relationships	In UML 1
Component	Structure and connections of components	In UML 1
Deployment	Deployment of artifacts to nodes	In UML 1
Sequence	Interaction between objects; emphasis on sequence	In UML 1
State machine	How events change an object over its life	In UML 1
Use case	How users interact with a system	In UML 1
Communication	Interaction between objects; emphasis on links	UML 1 collaboration diagram
Object	Example configurations of instances	Unofficially in UML 1
Package	Compile-time hierarchic structure	Unofficially in UML 1
Composite structure	Runtime decomposition of a class	New to UML 2
Interaction overview	Mix of sequence and activity diagram	New to UML 2
Timing	Interaction between objects; emphasis on timing	New to UML 2

11

Use Case Diagram

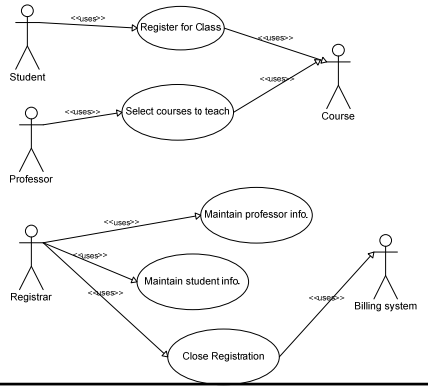
12

Use Case Diagram

- Describes the requirements of the system, especially functional requirements
- Use case diagram contains
  - Actors and their descriptions
  - Relationships between different entities

13

Example: University Course Registration System



14

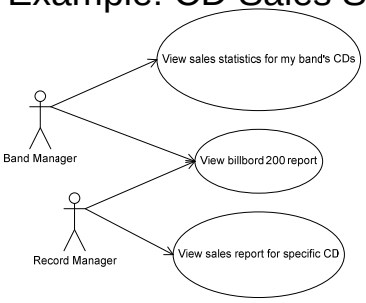
Elements of Use Case Diagram

An actor represents anything that interacts with the system.
- Actors are not part of the system, they represent roles a user of the system can play.
- An actor can actively interchange information with the system.
- An actor can be a passive recipient of information.
- An actor can be a giver of information.
- An actor can represent a human, a machine or another system.

A use case (instance) defines a sequence of actions a system performs that yields a result of observable value to an actor.
- A use case models a dialogue between an actor and the system.
- A use case is initiated by an actor to invoke a certain functionality in the system.
- A use case is a complete and meaningful flow of events.
- Taken together, all use cases constitute all possible ways of using the system.

15

Example: CD Sales System

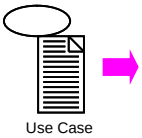


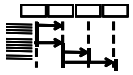
DEMO

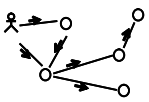
http://bronze.rational.com:8169/content/jun_03/f_UMLintro_db.jsp

16

- Once the use case diagram is completed (in other words, the requirement specification is done), analysis and design take places.


Use Case


Sequence Diagrams


Collaboration Diagrams


Class Diagrams

17

Class Diagram

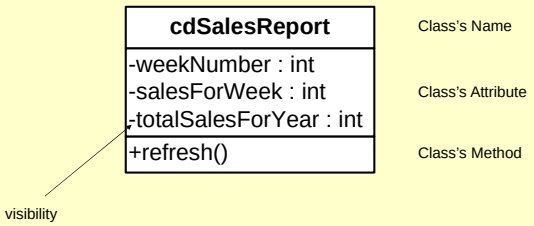
18

Class Diagram

- Used in requirements, design and implementation:
  - Conceptual, to represent general entities in system
  - Specification, where we specify what each entity (class) will do (but not how)
    - List the methods/actions
  - Implementation
    - Detailed class diagram of actual software (Java or C++)
- List attributes, same as data model
- List methods/operations/functions
  - Activities naturally associated with the data in the entity
- We often don't model everything—too hard to read
  - Focus on key parts of system

19

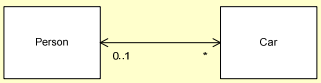
A class element



20

Association

- Associations describe relationships between classes
  - Solid lines are used to describe associations
    - Multiplicity
    - Unidirectional vs. bidirectional

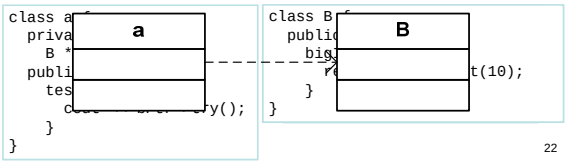


•A person may own 0 to many cars
•A car may have 0 or 1 owner

21

Dependency

- A dependency exists between two elements if changes to the definition of one element (the supplier) may cause changes to the other (the client).
  - one class sends a message to another;
  - one class has another as part of its data;
  - one class mentions another as a parameter to an operation.



22

Relationships

Construct	Description	Syntax
Association	A relationship between two or more classifiers that involves connections among their instances.	———
Aggregation	A special form of association that specifies a whole-part relationship between the aggregate (whole) and the component part.	——◇
Generalization	A taxonomic relationship between a more general and a more specific element	——▷
Dependency	A relationship between two modeling elements, in which a change to one modeling element (the independent element) will affect the other modeling element (the dependent element)	- - - - - ▷
Realization	A relationship between a specification and its implementation	- - - - - ▷
Composition	If a class cannot exist by itself, and instead must be a member of another class, then that class has a Composition relationship with the containing class. A Composition relationship is indicated by a line with a filled diamond.	——◆

Constraint

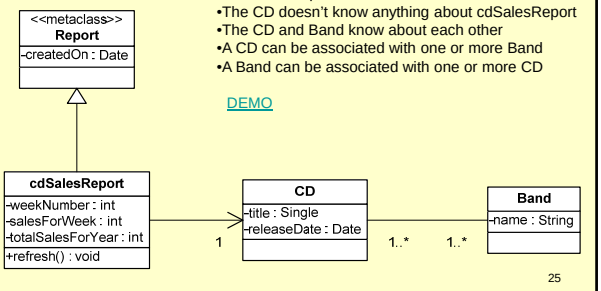
- Constraints can be described by anything, as long as they are inside braces {}.
  - E.g. {debt free: cash in bank should be > 0}
- Conditions
  - Pre-condition: apply to operations. A pre-condition is a statement of how we expect the world to be before we execute an operation.
    - E.g. "square root" operation of input > = 0.
  - Post-condition: applies to operations. A post-condition is a statement of what the world should look like after execution of an operation.
    - E.g. After "square root" operation, input = result * result
  - The conditions suggest the responsibility of checking arguments and results falls on the shoulder of callers
  - Invariant: is an assertion about a class. The invariant is "always" true for all instances of the class.

24

Example

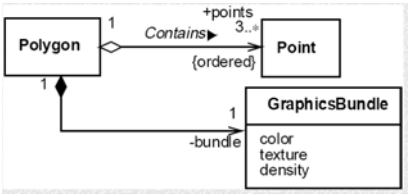
- The cdSalesReport inherits from the Report class
- A cdSalesReport is associated with one CD
- The CD doesn't know anything about cdSalesReport
- The CD and Band know about each other
- A CD can be associated with one or more Band
- A Band can be associated with one or more CD

DEMO



http://bronze.rational.com:8169/content/jun_03/f_UMLintro_db.jsp

Example



- A polygon contains 3 or more points
- 3 or more points are aggregated in a polygon
- One polygon has one GraphicsBundle, and one GraphicsBundle belongs to exactly one polygon

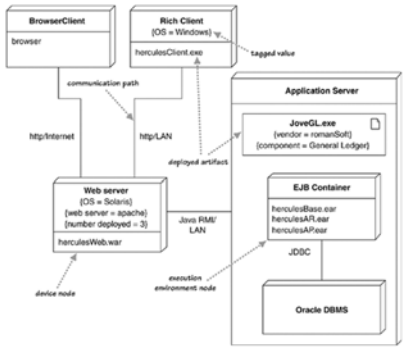
C. Kobryn (1999), "Object Modeling with UML: Introduction to UML"

Deployment Diagram

Deployment Diagrams

- Deployment diagrams show the physical relationship between hardware and software in a system.
- The solid lines represent communication links or dependencies.

Example



Reference: UML Distilled: A Brief Guide to the Standard Object Modeling Language, Third Edition, by Martin Fowler (2003)

Interaction Diagrams

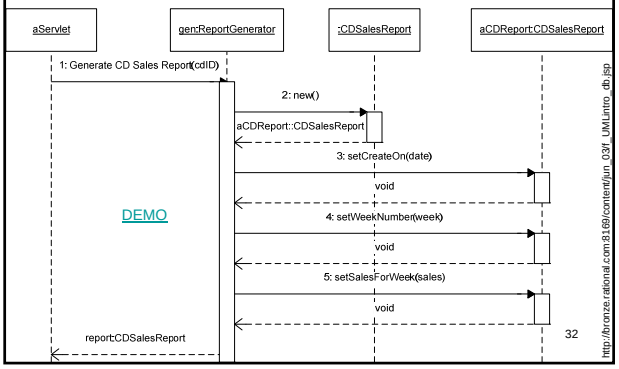
Sequence Diagram

Sequence Diagram

- Shows a detailed or partial flow for a specific use case
- Shows the calls between different objects in their sequence
- 2-D:
  - Vertical: time/order
  - Horizontal: object instances

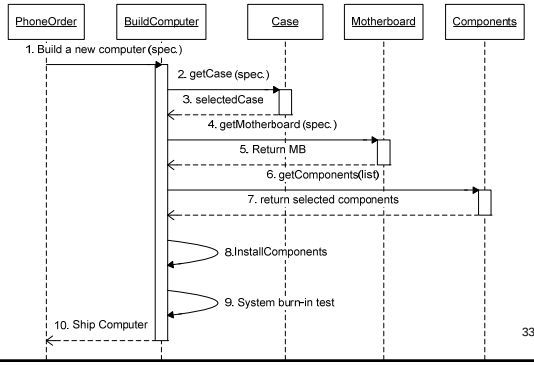
31

Example



32

Example



33

Summary

- Brief Introduction to UML Diagrams
  - Use case diagram
  - Class diagram
  - Deployment diagram
  - Sequence diagram

34

References for UML

http://www-306.ibm.com/software/rational/uml/

http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/index.htm

http://www.sparxsystems.com.au/uml-tutorial.html

UML Distilled: A Brief Guide to the Standard Object Modeling Language, Third Edition, by Martin Fowler (2003), ISBN 0-321-19368-7

35

Online UML Examples

- Google: uml examples
- http://www.geocities.com/SiliconValley/Network/1582/uml-example.htm
- http://www.smartdraw.com/examples/software-uml/
- http://portal.etsi.org/mbs/Languages/UML/uml_example.asp

36