

Object-Oriented Design (OOD)

• Design

- Given system requirements
- Prepare a plan for implementation

• Process (simplified)

- Classification
- Use cases

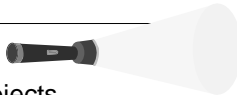


Brief introduction; more later!

Classification

• Identifying

- Classes - types of objects
- Objects - specific objects



• Describing

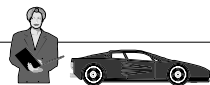
- Attributes - "knows"
- Behaviors - "can do"



Types of Classes

• Domain classes

- Represent "things" in problem domain



• Implementation classes

- Artifacts of software system itself
 - E.g., containers, user interface



Identifying Domain Classes

- **Read problem statement**
 - Identify relevant portions
- **Nouns**
 - Candidate objects
- **Adjectives or qualifiers**
 - Candidate attributes
- **Verbs**
 - Candidate behaviors

Ivar Jacobson



A Tool: Use Cases

- **Scenarios**
 - Typical uses of desired system
- **Detailed description**
 - Objects and interactions
- **Aids classification**
 - Validates class and object choices

Design Example



- **Registrar scheduling problem**
 - See problem statement
- **Software system needed**
- **Object-oriented design**
 - Manual methods for now
 - CASE tools later?

Problem Statement: Nouns

- Registrar
- Class
- Classrooms
- Labs
- Quarter
- Course
- Section
- Professor
- Campus
- Session
 - Lecture
 - Lab
- Student

Problem Statement: Qualifiers

- Classroom
 - Number
 - Capacity
- Section
 - Number
 - Enrollment
- Course
 - Number
- Session
 - Time period(s)
- Student
 - Name
 - Number

Problem Statement: Verbs

- Assign
 - Session to period, room
- Track
 - Classrooms
 - Laboratories
- Register
 - Students/sections
- Modify
 - Assignments
- Report
 - Session assignments

Selecting Relevant Details

• Can we ignore?

- Professors
- Students
- Registration



• Objects in problem domain

- But outside scope of our design



Sample Use Cases

- Add room to room list
- Assign section session(s) to period(s)/room(s)
- Delete section assignments
- Report assignments by room



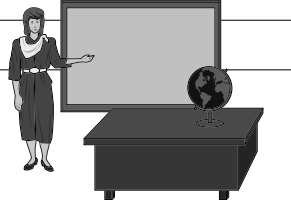
Use Case: Add Room to List

- Create a room object
- See if room is already in list
- If not, add room to list

Classes needed: room, room list



Room Class



•Attributes

- Room number
- Capacity

•Behaviors

- Constructor
- Accessors (inspectors)
- Mutators

Do we need anything else?

Room List Class

•Attributes

- ??

•Behaviors

- Constructor
- Add/delete room
- Find room by room number

•Related objects

- Rooms

S-359
S-360
S-362
L-307
S-210
CC-09

Document the Design (So Far)

•Classes

- UML diagrams
- Code (header files)

•Use cases

- Word processor?
- Code comments?

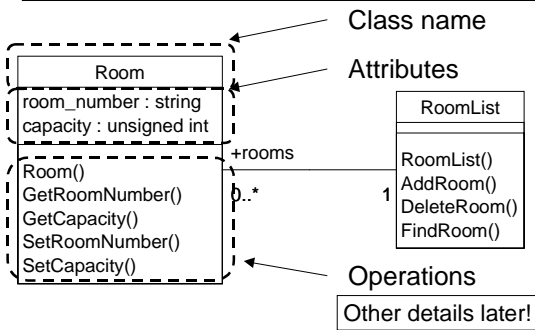


Unified Modeling Language (UML)

- **Standard notation**
 - Grady Booch, Jim Rumbaugh, others
- **Document OO designs**
 - Classes, relationships
 - Objects, interaction
 - Use cases
 - Physical partitioning (files)



UML Notation Example



Iterative Design Process

- **Model part of design**
 - Document classes
- **Complete partial design**
 - Validate with use cases, etc.
- **Expand and modify design**
 - Maintain current documentation

