Back to

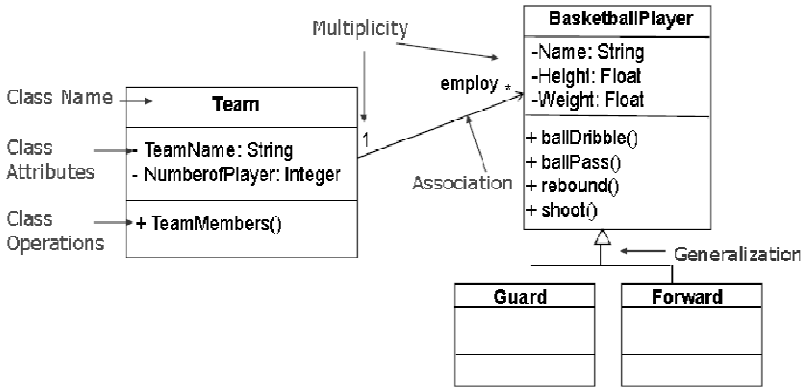
UML

Part II

Class Diagram

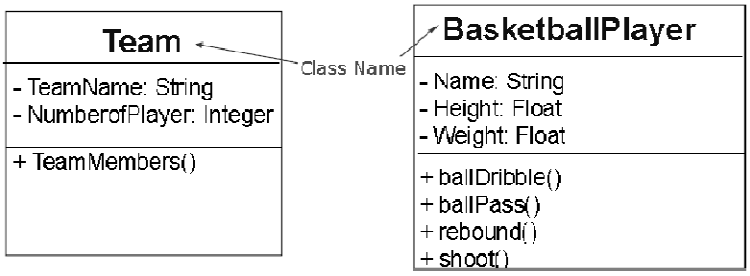
- Motivated by Object-Oriented design and programming (OOD, OOP).
- A class diagram partitions the system into areas of responsibility (classes), and shows "associations" (dependencies) between them.
- Attributes (data), operations (methods), constraints, part-of (navigability) and type-of (inheritance) relationships, access, and cardinality (1 to many) may all be noted.

Class Diagram



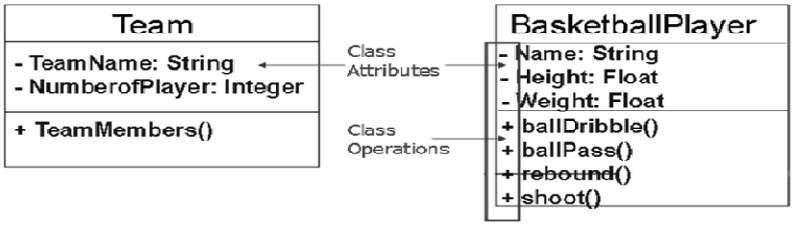
Class Name

- Most important building block of any object-oriented system
  - Description of a set of objects
  - Abstraction of the entities



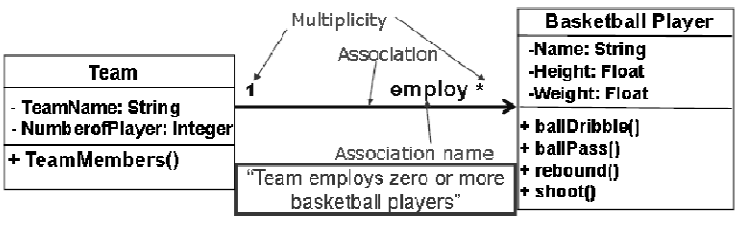
Class Attributes & Operations

- Attributes
  - Represent some property of the thing being modeled
  - Syntax: attributeName : Type
- Operations
  - Implement of a service requested from any object of the class
  - Syntax: operationName (param1:type, param2:type, ...) : Result



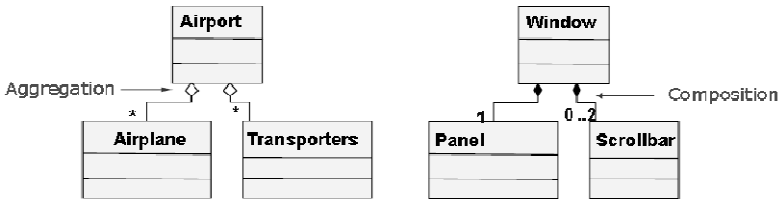
Association and Multiplicity

- Association
  - Relationship between classes that specifies connections among their instances
- Multiplicity
  - Number of instances of one class related to ONE instance of the other class



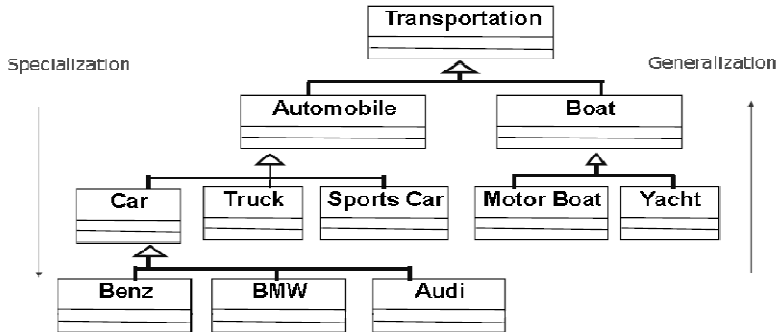
Aggregations and Compositions

- Aggregation
  - Weak "whole-part" relationship between elements
    - **An airport has many airplanes in it.**
- Composition
  - Strong "whole-part" relationship between elements
    - **Window 'contains a' scrollbar**



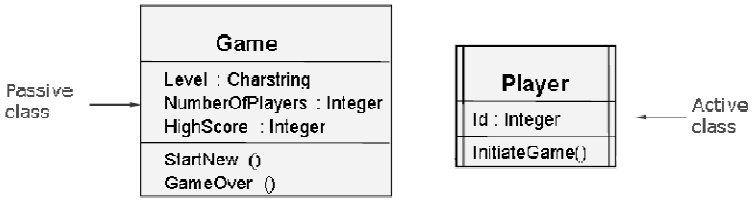
Inheritance

- Relationship between superclass and subclasses
  - All attributes and operations of the superclass are part of the subclasses



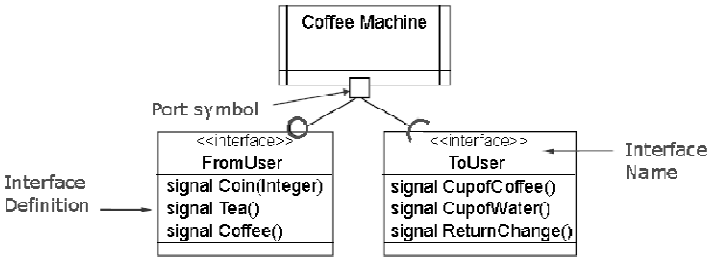
Active vs. Passive Class

- Active class
  - Own a thread control and can initiate control activity
    - **Used when asynchronous communication is necessary**
    - **Typically modeled with a state machine of its behavior**
    - **Encapsulated with ports and interfaces**
- Passive class
  - Own address space, but not thread of control
    - **Executed under a control thread anchored in an active object**



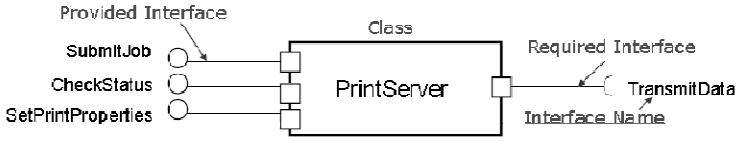
Ports and Interfaces

- Ports
  - Define an interaction point on a classifier with external environment
- Interfaces
  - Describe behavior of objects without giving their implementations
    - **Each class implements the operations found in the interface**

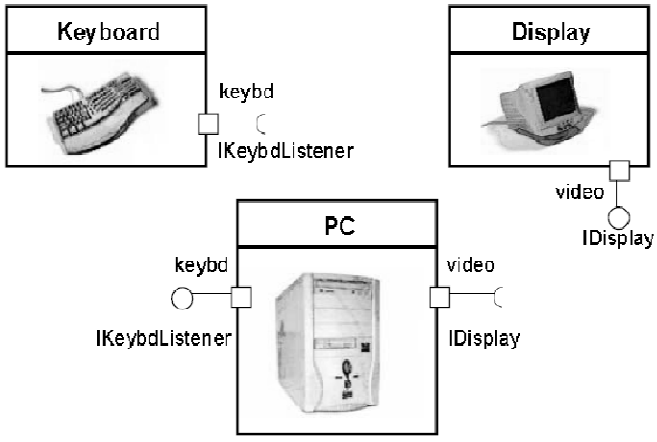


Provided/ Required Interface

- Provided interface
  - Class provides the services of the interface to outside callers
  - What the object can do
  - Services that a message to the port may request (incoming)
- Required interface
  - Class uses to implement its internal behavior
  - What the object needs to do
  - Services that a message from the port may require from external environment (outgoing)

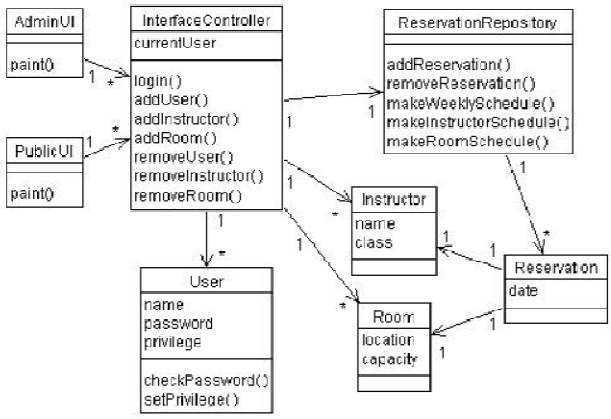


Computer Device Example



Class Diagram Example

Classroom Scheduling System



About the last example…

- Each box is a class, with necessary attributes and operations specified.
- Navigability arrows show which classes can reference which others.
- Cardinality marked in bi-directional manner on arrows.
- The classes together represent the complete system; thus the classes are a *partitioning of the system.*