Unified Modeling Language 2

Nelson Padua-Perez  
Chau-Wen Tseng

Department of Computer Science  
University of Maryland, College Park

UML Class Diagrams

- Represent the (static) structure of the system
- General  
  - Name  
  - State  
  - Behavior  
- In Java  
  - Name  
  - Variables  
  - Methods

```
Clock
seconds:int
minutes:int
hours:int

start()
adjustTime()
reset()

Name

State

Behavior
```
Relationships Between Classes

- Association
  - Permanent, structural, “has a”
  - Solid line with arrowhead

- Dependency
  - Temporary, “uses a”
  - Dotted line with arrowhead

- Generalization
  - Inheritance, “is a”
  - Solid line with open triangular arrowhead

- Implementation
  - Dotted line with open triangular arrowhead

Association

- Denotes permanent, structural relationship
  - Occurs when state of class A contains class B
  - View as a “has a” relationship between classes
  - Represented by solid line (with arrowhead)

Example

```
Class Car {
    Engine myEngine;
}
```

```
Class Engine {
    ...
}
```

Car “has a” Engine
Association w/ Navigation

Navigation information
- Relationship between classes may be directional
- Arrowhead indicates direction of relationship

Car class knows about Engine class
Engine class doesn’t know about Car

Association w/ Navigation

One-way association
- Only one class contains other class
- Use arrowhead to indicate direction of relationship
  - Point to class contained in 2nd class

Example

```java
Class Car {
    Engine myEngine;
}
```

```java
Class Engine {
    ...
}
```
Association w/ Navigation

- Bi-directional association
  - Both classes contain object of other class
  - Use undirected solid edge (no arrowheads)

Example

```java
Class Car {
    Engine myEngine;
}
```

```java
Class Engine {
    Car myCar;
}
```

Multiplicity of Associations

- Some relationships may be quantified
- Multiplicity denotes how many objects the source object can legitimately reference

Notation

- `*` ⇒ 0, 1, or more
- `5` ⇒ 5 exactly
- `5..8` ⇒ between 5 and 8, inclusive
- `5..*` ⇒ 5 or more
### Multiplicity of Associations

**Many-to-one**
- Bank has many ATMs, ATM knows only 1 bank

**One-to-many**
- Inventory has many items, items know 1 inventory

### Dependency
- Denotes **dependence** between classes
- Always directed (Class A depends on B)
- Represented by dotted line with arrowhead
Dependency

- Caused by class methods
- Method in Class A temporarily "uses a" object of type Class B
- Change in Class B may affect class A

A uses object of class B

Dependency

- Dependence may be caused by
  - Local variable
  - Parameter
  - Return value

Example

```java
Class A {
    B Foo(B x) {
        B y = new();
        return y;
    }
}
```
```java
Class B {
    ...
    ...
    ...
}
```
Dependency Example

Class Driver depends on Class Car

Generalization

- Denotes inheritance between classes
- Can view as “is-a” relationship
- Represented by line ending in (open) triangle

Laptop, Desktop, PDA inherit state & behavior from Computers
Implementation

- Denotes class `implements` Java interface
- Represented by dotted line ending in (open) triangle

![Diagram: A implements interface B]

UML Examples

- Read UML class diagram
- Try to understand relationships
- Examples
  - Pets & owners
  - Computer disk organization
  - Library books
  - Banking system
  - Home heating system
  - Printing system
Try to read & understand UML diagram

- 1 or more Pets associated with 1 PetOwner
UML Example – Computer System

Try to read & understand UML diagram

- 1 CPU associated with 0 or more Controllers
- 1-4 DiskDrives associated with 1 SCSIController
- SCSIController is a (specialized) Controller
UML Example – Library System

Try to read & understand UML diagram

- 1 or more Book associated with 1 or more Pages
- Patron & Shelf temporarily use (depend on) Books
Try to read & understand UML diagram

1. 1 Bank associated with 0 or more Accounts
2. Checking, Savings, MoneyMarket are Accounts
Try to read & understand UML diagram

- Each Thermostat has 1 Room
- Each Thermostat associated with 0 or more Heaters
- ElectricHeater is a specialized Heater
- AubeTH101D is a specialized Thermostat
UML Class Diagrams ↔ Java

- Different representation of same information
  - Name, state, behavior of class
  - Relationship(s) between classes
- Practice deriving one from the other
  - Accurately depicting relationship between classes

UML → Java: Veterinary System
UML → Java : Veterinary System

**UML**

```
Pet 1..* 1 PetOwner
```

**Java**

class Pet {
    PetOwner myOwner; // 1 owner for each pet
}
class PetOwner {
    Pet[] myPets;     // multiple pets for each owner
}

Java → UML : Veterinary System

**Java**

class Pet {
    PetOwner myOwner; // 1 owner for each pet
}
class PetOwner {
    Pet[] myPets;     // multiple pets for each owner
}

**UML**
Java → UML : Veterinary System

Java

class Pet {
    PetOwner myOwner;       // 1 owner for each pet
}
class PetOwner {
    Pet[ ] myPets;          // multiple pets for each owner
}

UML

UML Class Diagrams ↔ Java

UML

Java

class Pet {
    PetOwner myOwner;       // 1 owner for each pet
}
class PetOwner {
    Pet[ ] myPets;          // multiple pets for each owner
}
class Controller {
}

class SCSIController extends Controller {
}
Design code using all available information in UML...

class CPU {
    Controller [ ] myCtirs;
}
class Controller {
    CPU myCPU;
}
class SCSIController extends Controller {
    DiskDrive [ ] myDrives = new DiskDrive[4];
}
Class DiskDrive {
    SCSIController mySCSI;
}
Java  →  UML: Printing System

Java

class Registry {
    PrintQueue findQueue();
}
class PrintQueue {
    List printJobs;
    Printer myPrinter;
    Registry myRegistry;
    void newJob();
    int length();
    Resources getResource();
}

Java  →  UML: Printing System

Java

Class Printer {
    Resources myResources;
    Job curJob;
    void print();
    boolean busy();
    boolean on();
}
class Job {
    Job(Registry r) {
        ...
    }
}
Java → UML: Printing System

UML Summary

- Graphics modeling language
- Visually represents design of software system
- We focused on class diagrams
  - Contents of a class
  - Relationship between classes
- You should be able to
  - Draw UML class diagram given Java code
  - Write Java code given UML class diagram