CMSC 128

Introduction to Software Engineering
Second Semester AY 2007-2008

jachermocilla@uplb.edu.ph

Design Patterns

- A design pattern is a widely accepted solution to a recurring design problem in OOP
- Describes how to structure classes to achieve a requirement
- Provides a general blueprint when implementing part of a program
- Popularized by the “Gang of Four”

Benefits

- Learn from community wisdom
- Determine implementation faster
- Make code more readable to other programmers

Categories

- Creational Design Patterns
  - For creating objects
- Structural Design Patterns
  - Organizing subsystems
- Behavioral Design Patterns
  - Object behaviors

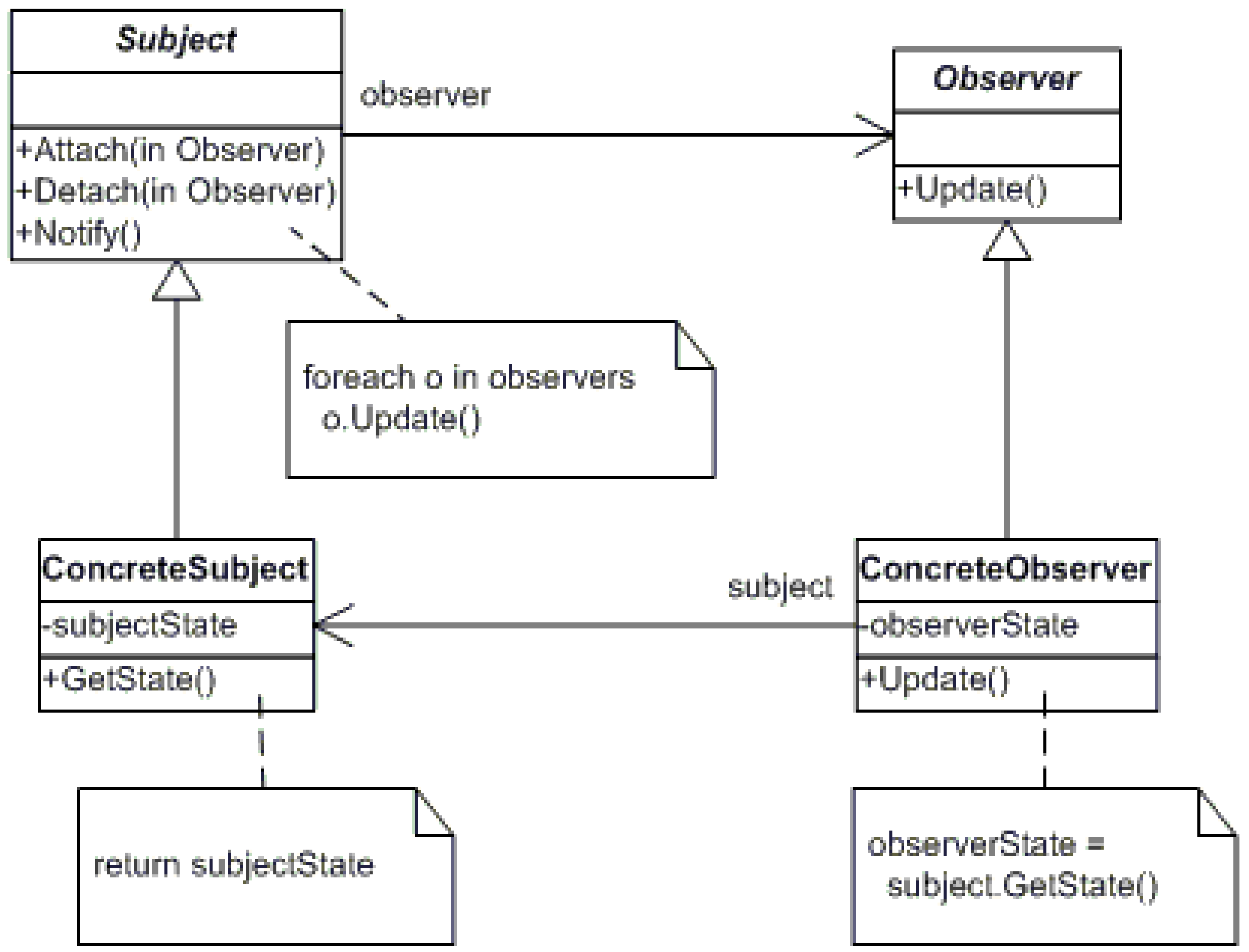
Observer Design Pattern

- Behavioral design pattern
- You want to notify objects that a particular event has happened
- Object that changes is the subject
- Objects that receive updates are observers
- Case study: Friendster Bulletin Board
  - Subject: You
  - Observers: Friends

Observer Design Pattern

- Responsibilities of Subject
  - Maintain a list of observer objects
  - Provide methods for adding/removing observers
  - Informing observer of events
- Responsibilities of Observer
  - Process message sent by subject

Observer Design Pattern



Observer Design Pattern

```
public interface Subject {
   public void addObserver(Observer o);
   public void removeObserver(Observer o);
   public void inform(Object o);
}
```

Observer Design Pattern

```
public interface Observer {
   public void update(Object o);
}
```

Observer Design Pattern

```
public class FriendsterAccount implements
  Observer, Subject{
   private String name;
   private List<Observer> friends = new
     ArrayList<Observer>();


   ...


}
```

Observer Design Pattern

```
public class FriendsterAccount implements
  Observer, Subject{

   ...

   public void inform(Object msg){

      Iterator<Observer> ite =
       friends.iterator();

      while (ite.hasNext()){

         Observer ob = ite.next();

         ob.update(msg);

      }

   }}
```

Observer Design Pattern

```
public class FriendsterAccount implements
  Observer, Subject{
   ...
   public void postBulletin(String msg){
      inform(name+": " +msg);
   }
}
```

Observer Design Pattern

```
public class FriendsterAccount implements
  Observer, Subject{
   ...
   public void update(Object msg){
      System.out.println(name+": New bulletin
        from "+msg);
   }
}
```

Reference

- http://www.moock.org/lectures/introToPatterns/