* INFO2413 System Development Project

Dr. Jendy Wu
Spring 2018

Computer Science and Information Technology
Kwantlen Polytechnic University

Today's Outlines

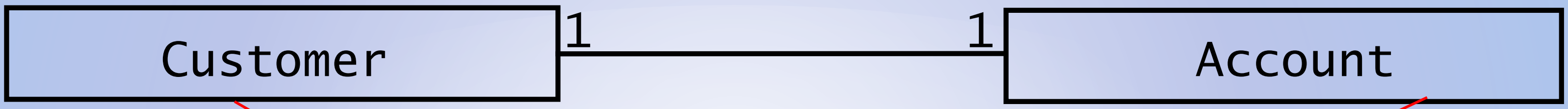
* Implementation
  * From model space to source code space
  * Implementation document
* Introduction of Testing
  * Testing strategy
  * Test report document

* Mapping Associations

1. Unidirectional one-to-one association
2. Bidirectional one-to-one association
3. Bidirectional one-to-many association
4. Bidirectional many-to-many association

* Unidirectional one-to-one association

Object design model before transformation:

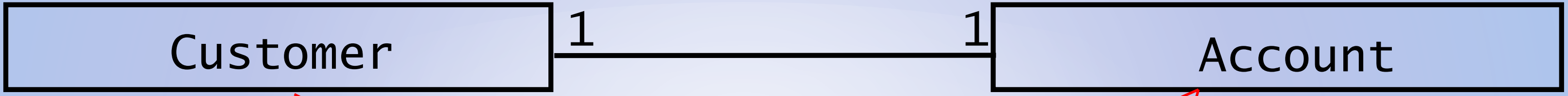


Source code after transformation:

```
public class Customer {
      private Account account;
      public Customer() {
            account = new Account();
      }
      public Account getAccount() {
            return account;
      }
}
```

* Bidirectional one-to-one association

Object design model before transformation:



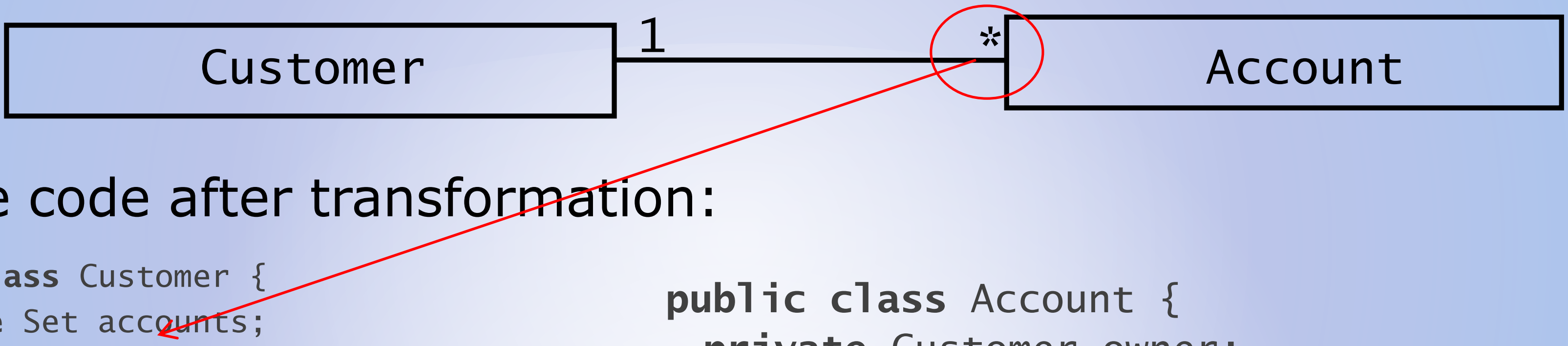
Source code after transformation:

```
public class Customer {
/* account is initialized
 * in the constructor and never
 * modified. */
  private Account account;
  public Customer() {
          account = new Account(this);
  }
  public Account getAccount() {
          return account;
  }
}
```

```
public class Account {
  /* owner is initialized
   * in the constructor and
   * never modified. */
  private Customer owner;
  public Account(Customer owner) {
          this.owner = owner;
  }
  public Customer getOwner() {
          return owner;
  }
}
```

* Bidirectional one-to-many association

Object design model before transformation:



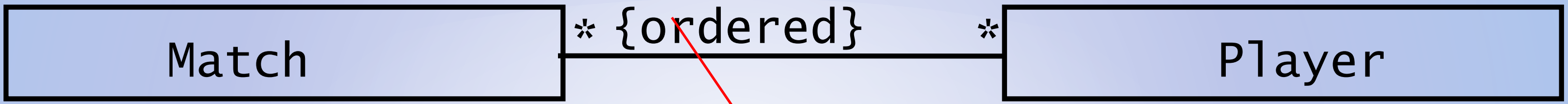
Source code after transformation:

```
public class Customer {
  private Set accounts;
  public Customer() {
        accounts = new HashSet();
  }
  public void addAccount(Account a) {
        accounts.add(a);
        a.setOwner(this);
  }
  public void removeAccount(Account a) {
        accounts.remove(a);
        a.setOwner(null);
  }
}
```

```
public class Account {
  private Customer owner;
  public void setOwner(Customer newOwner)
  {
    if (owner != newOwner) {
        Customer old = owner;
        owner = newOwner;
        if (newOwner != null)
            newOwner.addAccount(this);
        if (oldOwner != null)
            old.removeAccount(this);
    }
  }
}
```

* Bidirectional many-to-many association

Object design model before transformation



Source code after transformation

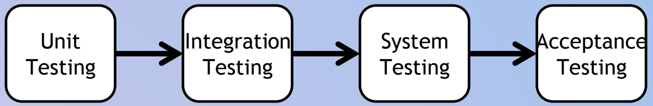
```
public class Match {
  private List players;
  public Match() {
      players = new ArrayList();
  }
  public void addPlayer(Player p) {
      if (!players.contains(p)) {
          players.add(p);
          p.addMatch(this);
      }
  }
}
```

```
public class Player {
  private List matches;
  public Player() {
      matches = new ArrayList();
  }
  public void addMatch(Match t) {
      if (!matches.contains(t)) {
       matches.add(t);
       t.addPlayer(this);
      }
  }
}
```

Marking Criteria of the Implementation Stage Submission

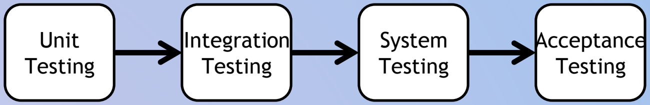
- Document format and general information: 4 marks
  - General information on software implementation, for example hardware/software requirements…
  - Detailed implementation explanation on key tasks, for example, key method on how to finish certain key tasks
  - Acceptance criteria
- Completed system implementation: 26 marks
  - Make sure the quality of implementation, for example, if it is a web application, then you can’t make it as a desktop application.
  - Completed implementation on all functionalities that are stated in the SRS.

*Types of Testing



- *Unit Testing
  - *Individual component (class or subsystem)
  - *Carried out by developers
  - *Goal: Confirm that the component or subsystem is correctly coded and carries out the intended functionality
- *Integration Testing
  - *Groups of subsystems (collection of subsystems) and eventually the entire system
  - *Carried out by developers
  - *Goal: Test the interfaces among the subsystems.

Types of Testing continued…



- System Testing
  - The entire system
  - Carried out by developers
  - Goal: Determine if the system meets the requirements (functional and nonfunctional)
- Acceptance Testing
  - Evaluates the system delivered by developers
  - Carried out by the client. May involve executing typical transactions on site on a trial basis
  - Goal: Demonstrate that the system meets the requirements and is ready to use.

Marking Criteria of the Testing Stage Submission

- Document format and general information: 2 marks
  - Testing general introduction
  - Testing quality
- Unit testing details: 6 marks
  - Unit testing for major methods
  - Unit testing should include test cases which have specific value as input
- Integration testing details: 2 marks
  - Clearly explain how you carry out integration testing and testing results
- System testing details: 10 marks
  - All functional requirements testing details and results
  - All non-functional requirements testing details and results
  - Please make sure that the functional and non-functional requirements match to your SRS.

Summary

- This topic
  - Implementation and Validation