Design and Development Steps

- Requirements gathering
- Specification
- Architecture (for bigger systems)
- Design
- Implementation
- Test
Object-Oriented (OO) Design

• What’s an object?
  – State (instance variables) and
  – operations (methods that act on the state and other inputs).
  – What’s good about object orientation?
    • Abstraction
    • Encapsulation
    • Reusability
Key Steps In OO Design

• (Given a specification and an architecture.)
• Within a module identify real or abstract entities
• Map entities to objects (i.e., classes).
• Then determine
  – Object state and operations
    • i.e., instance variables and methods
  – Object relationships
    • i.e., inheritance and composition
  – Tests
• Implement
Approaches for Identifying Classes

- Noun phrase
- Use-case driven
- Classes, Responsibilities and Collaborators
- Common Class Patterns
Noun Phrase Technique

1. Enumerate Noun phrases
2. Eliminate irrelevant
3. Eliminate redundant
4. Identify attributes
5. Describe each class’s purpose

*Try this with the ViaNet Bank ATM*
Use-case Driven Approach

• Identify classes through sequence and collaboration modeling
• Develop sequence diagrams

Try this with the ViaNet Bank ATM
Classes, Responsibilities and Collaborators (CRC)

An object’s collaborators are objects that cooperate with the object

1. Describe each class on an index card:

<table>
<thead>
<tr>
<th>Class name</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• First</td>
</tr>
<tr>
<td></td>
<td>• Second</td>
</tr>
<tr>
<td></td>
<td>• Etc.</td>
</tr>
<tr>
<td></td>
<td>Collaborators</td>
</tr>
<tr>
<td></td>
<td>• Using classes</td>
</tr>
<tr>
<td></td>
<td>• Superclasses</td>
</tr>
<tr>
<td></td>
<td>• Subclasses</td>
</tr>
</tbody>
</table>

2. Iterate!
   • Keep collaborators physically nearby.
   • Combine similar classes in hierarchies

   Try this with the ViaNet Bank ATM
Common Class Patterns

• Concept class; e.g., [performance]
• Events class;
• Organization class
• People class
• Places class
• Things and devices class

Try this with the ViaNet Bank ATM
OO Design Methodologies

• *Unified* (Booch, Rumbaugh and Jacobson) approach
  – *Use-case driven development*
  – *OO analysis = specification*
  – **OO design**
  – *Incremental development and prototyping*
  – *Continuous testing*
**OO Design Axioms**

1. *The independence axiom.* Maintain the independence of components.

2. *The information axiom.* Minimize the complexity (information content) of the design.

- **Design rules (antecedent axioms):**
  - Design highly cohesive objects that require low coupling (1, 2).
  - Each class should have a single purpose (1, 2).
  - A large number of simpler classes enhances reusability (1, 2).
  - Map strongly from objects in the analysis to objects in the design (1).
  - Promote standardization by reusing classes and building to standard interfaces (2).
  - Design for inheritance—move common behavior to superclasses (2).
References

• Coad and Yourdon. *Object-Oriented Analysis*, 1991.
• Beck, Cunningham, *A Laboratory For Teaching OO Thinking*, OOPSLA, 1989 (CRC technique)