Use Cases and Interaction Diagrams (Sequence and Collaboration)

Hours 6, 7, 9, and 10

Scenarios
The starting point
- A scenario is a sequence of steps
- Scenarios are presented in text format
- No UML diagram contains the text
- Drives use cases, sequence, and collaboration diagrams
- Include in the documentation
- Models a single flow of control
- Doesn’t show possibilities (i.e., if... else) just one path
- Other possibilities are shown with other scenarios

Use Case Diagrams
Models system interactions
- Models a system at a very high level
- Starting point for
  - Designing class diagrams
  - Interaction diagrams
  - Statechart diagrams (where appropriate)
  - Activity diagrams (where appropriate)
  - Creating documentation
  - Building test cases
- Readable by a wide range of stakeholders
- No single use case needs to capture everything about a system

Two Interaction Diagrams
They are “isomorphic”
- Interaction diagrams model the messages exchanged between objects
- Sequence diagrams emphasize the time ordering of messages
  - “A single sequence diagram can only show one flow of control”
  - What Schmuller calls an instance sequence diagram
  - A generic sequence diagram considers all of the use case’s scenarios
- Collaboration diagrams emphasize the structural connections between objects

Sequence Diagram
Models a single scenario

Elements of A Sequence Diagram
Basic components
- Objects
  - Conveniently ordered (i.e., to simplify the diagram) at the top
  - Typically, the object that initiates the sequence is on the left
  - Increasingly subordinate objects are placed on the right
- Life lines
  - Vertical dashed line descending from an object
  - Denotes the existence of an object over a period of time
  - Most objects exist for the duration of the interaction
  - Can show the creation and destruction of objects
- Focus of control or an activation
  - Tall, thin rectangle
  - Shows the period when an object is executing an operation
  - Program control lies within an object
### Elements of A Sequence Diagram

**Messages**

<table>
<thead>
<tr>
<th>Schmuller</th>
<th>3 Amigos / Visio</th>
</tr>
</thead>
</table>
| • Simple
  • transfer of control to another object | • Simple
  • transfer of control to another object |
| • Synchronous
  • sender waits for receiver | • Flat
  • models interactions of actors with use cases or objects |
| • Asynchronous
  • sender does not wait | • Procedure call (synchronous)
  • one object invokes an operation on another |
|                        | • Asynchronous
  • "mail box" semantics |
|                        | • Return
  • returns a value to the caller |

### Creating and Destroying Objects

- Use stereotype to show create and destroy messages
- Move object symbol down to show relative creation time
- "X" to show object destruction

### Showing Recursion

**Sequence diagram**

- Simple recursion: a function calls itself
- Complex recursion: function A calls function B, function B calls function C, function C calls function A

### Collaboration Diagram

Same information as a sequence diagram

### Statechart Diagram

**Example 1**

- Example of a statechart diagram illustrating state transitions and events.

**Example 2**

- Another statechart diagram with different states and transitions.

### Statechart Diagram

- Diagram showing state transitions and events.