System Behavior Analysis

OO Analysis Model

Modeling Objectives

• System Sequence Diagrams
  – Illustrate interaction between user and system
  – Identify system events and operations
  – Create system sequence diagrams for use cases

• Contracts
  – Describe the effect of system operations on objects in the conceptual model.
  – Define pre- and post-conditions of operations
Sequence Diagrams

• Show objects by their “lifelines”, and the sequence of stimuli that they exchange.
  – The vertical dimension represents time. Usually only time sequences are important, but in real-time applications the time axis could be an actual metric.
  – The horizontal dimension represents different objects. There is no significance to the horizontal ordering of the objects.
  – Stimuli are shown as arrows between lifelines.

System Sequence Diagrams

• During analysis, system behavior is defined as a “black box” (*what not how*).
• System sequence diagrams focus on events (stimuli) that cross the system boundary between actors and systems.
• System sequence diagrams illustrate specific *instances* of use cases.

System Sequence Diagrams (2)

<table>
<thead>
<tr>
<th>Use Case: Buy Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Course</td>
</tr>
<tr>
<td>1. The system begins when a Customer arrives at the POST to place an order</td>
</tr>
<tr>
<td>2. The Cashier records the UPC code from each item. If there is more than one of the same item, the cashier may also enter the quantity</td>
</tr>
<tr>
<td>3. System determines the price of the item and adds the item information to the running sales transaction. The description and price of the current item are displayed</td>
</tr>
<tr>
<td>4. And so on...</td>
</tr>
</tbody>
</table>

```plaintext
System

Cashier

User Interface

System

User Interface

Cashier

System
```

Customer

System

endSale()

System

makePayment(amount)

System

endSale()

System

makePayment(amount)

System

endSale()

System

makePayment(amount)

System

endSale()

System

makePayment(amount)

System

endSale()

System

makePayment(amount)

System

endSale()

System

makePayment(amount)
System Events and Operations

- System events are external input events generated by an actor to the system.
- System operations are operations that the system executes in response to a system event.
- For each event, there is a corresponding operation with the same name.

Sequence Diagram Process

- During analysis phase of each development cycle.
- Dependent on prior development of use cases.

Sequence Diagram Process (2)

1. Draw the system lifeline.
2. Identify each actor that directly interacts with the system. Draw a lifeline for each.
3. From the use case text, identify the system (external) events that each actor generates. Illustrate them on the diagram.
4. Optionally, include the use case text, or fragments of text to the left of the diagram.
Naming System Events

• Start names with verbs to emphasize the command orientation of these events.
• Express at the highest level of logical intent, rather than in terms of physical medium or GUI widget.

enterButtonPressed(amount)  worse
enterAmountTendered(amount) better
enterPayment(amount)
makePayment(amount)

System Operation Contracts

• Contracts describe what an operation does, but usually not how.
• Contracts are often specified in terms of pre- and post-conditions that define state changes.
• System operation contracts describe how high-level system operations effect the overall state of the system.

Example: enterItem

<table>
<thead>
<tr>
<th>Name</th>
<th>enterItem (upc: number, quantity: number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsibilities</td>
<td>Record sale of an item and add it to the sale. Display item description and price.</td>
</tr>
<tr>
<td>Type</td>
<td>System</td>
</tr>
<tr>
<td>Cross References</td>
<td>Use Cases: Buy Items, System functions: R1.1, R1.3, R1.9</td>
</tr>
<tr>
<td>Notes</td>
<td>UPC not valid</td>
</tr>
<tr>
<td>Output</td>
<td>Item description and price</td>
</tr>
<tr>
<td>Pre-conditions</td>
<td>UPC code is known to the system</td>
</tr>
<tr>
<td>Post-conditions</td>
<td>If a new sale, a Sale was created and associated with the POST. A SalesLineItem was created and associated with the Sale. The quantity attribute of the SalesLineItem was set to quantity. The SalesLineItem was associated with a ProductSpecification based on upc.</td>
</tr>
</tbody>
</table>
Contract Process
• During analysis phase within each development cycle.
• Dependent on prior development of conceptual model, system sequence diagrams, and identification of system operations.
• For each system operation, construct a contract.

Contract Dependencies

Contract Process (2)
1. Write the Responsibilities section, informally describing the purpose of the contract.
2. Complete the Post-Condition section, describing state changes that occur in the conceptual model:
   - Objects created and deleted
   - Attributes modified
   - Associations formed and broken
3. Complete the Pre-Condition section.
Post-conditions

- Declarations about changes to the problem domain state as a result of the operation. Not actions to be performed.
- The how is deferred until the design phase of the development cycle.
- Expressed in terms of conceptual model.
- May need to add new concepts, attributes, and associations to the conceptual model.

Post-conditions (2)

- There is a diminishing return on effort expended on defining post-conditions.
- Consider contracts written during analysis as drafts to be refined later.
- Contracts are valuable even if they are incomplete.
- Some of the details will be discovered during the design phase. The feedback is an advantage of the iterative approach.

Key Points

- System sequence diagrams are illustrations of use case instances, interactions between user and system.
- A system sequence diagram should be done for each basic use case, and possibly the more interesting alternative courses and extensions.
- System sequence diagrams are useful for discovering system operations.
<table>
<thead>
<tr>
<th>Key Points (2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Contracts describe operations in terms of pre- and post-conditions.</td>
</tr>
<tr>
<td>• Pre-conditions are declarations concerning what conditions must hold before the operation is invoked.</td>
</tr>
<tr>
<td>• Post-conditions define the state changes that will occur as a result of completing the operation.</td>
</tr>
<tr>
<td>• Conditions are defined in conceptual terms.</td>
</tr>
</tbody>
</table>