System Sequence Diagrams and Contracts (Larman)

Part of the analysis process

• System Sequence Diagrams:
  – Identify system events and system operations
  – Document sequence of interactions

• Contracts:
  – Define the pre- and post-conditions for system operations
System Sequence Diagram

• Sequence Diagram
  – General UML graphical notation
  – Models dynamic behaviour
  – Can model any dynamic phenomenon as sequence of interactions

• System Sequence Diagrams
  – A particular kind of sequence diagram
  – Deals only with the “black-box” view of the system and the interactions between the external actors and the system
System behaviour

• Defining the black-box system behaviour
• The system is the software (plus hardware), i.e. ignoring manual systems considered part of the overall business
• (Normally identify just one single system, although large systems may have loosely coupled parts that might be identified as separate parts even in a black-box model, e.g. the ticketing system and the train control system for an automated train)
System behaviour

• System sequence diagram
  – for a particular use case
  – events generated by external actors
  – the sequence of events

• events that cross the boundary between actors and system(s)
Buy Items-version 1

- Actor
- Cashier

system as black box

Repeat until no more items

Text which clarifies control, logic, iteration, etc.
May be taken from the use case.

enterItem(UPC, quantity)

endSale()

makePayment(amount)

system event
it triggers a system operation

:System
System Operations

- System event initiates system operation
Record system operations

- Required system operations determined by the system events:
  - `enterItem(UPC,quantity)`
  - `endSale()`
  - `makePayment()`

- Record then operations associated with these events as operations of a type
Record system operations

• UML notation for operations of a type:

```
<table>
<thead>
<tr>
<th>TypeX</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation1()</td>
</tr>
<tr>
<td>operation2()</td>
</tr>
</tbody>
</table>

operations of the type
```

• Example:

```
<table>
<thead>
<tr>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>endSale()</td>
</tr>
<tr>
<td>enterItem()</td>
</tr>
<tr>
<td>makePayment()</td>
</tr>
</tbody>
</table>
```
Make a system sequence diagram

• For a use case scenario
  – identify each actor operating on the system
  – draw a vertical line for the system, lines for each actor
  – identify the external events generated by actors
  – illustrate the sequence of events on the diagram
USE CASE: BUY ITEMS

Typical Course Of Events

1. This use case begins when a Customer arrives at the POST checkout with items to purchase.

2. The Cashier records the universal product code (UPC) from each item. If there is more than one of the same item, the Cashier can enter the quantity as well.

3. System determines the item price and adds the item information to the running sales transaction. The description and price of the current item are displayed.

4. and so on.
Buy Items-version 1

Cashier

enterItem(UPC, quantity)

endSale()

makePayment(amount)

System

system boundary
Some guidelines

• Consider only actors directly interacting with the system
• describe system events at an abstract level reflecting the ultimate goal of the operation
• begin name of system event with a verb indicating the invocation of an operation
For all items, the Cashier records the UPC and quantity.

On completion of item entry, the Cashier indicates to the POST that the sale is complete.

The Cashier tells the Customer the total, and the Customer gives a payment to the Cashier.

The Cashier records the cash received amount.

System sequence diagram with use case text
Contracts

- Describe the effect of operations on the system: pre-conditions; post-conditions
- *Design by Contract*, Bertrand Meyer
  software model: a set of communicating components whose interaction is based on precise specification of the mutual obligations, i.e. contracts.
System operation contracts

• The pre- and post-conditions describing the changes in the state of the overall system when a system operation is invoked

• c.f Hoare triple: \{P\} C \{Q\}
Making system operation contract

- Identify and name system operations from system sequence diagrams
- Determine the contract for each operation
- State the purpose or responsibilities of the operation
- State the pre-conditions; these describe the state of the system prior to the operation
- State the post-conditions; these describe the state changes as they affect objects in the conceptual model
USE CASE: BUYING ITEMS

Typical Course Of Events
1. This use case begins ...

Operation: enterItem
Postconditions: 1. If a new sale, a new Sale has been created...

Operation: endSale

Operation: makePayment

System Operations

Contracts
Contract guidelines

• Pre- and post-conditions are really predicates, i.e. statements that can be true or false;
• We require that the pre-conditions be true before the operation, and assert that the post-conditions will be true afterwards
• Categories for describing post-conditions
  – instance creation and deletion
  – attribute modification
  – associations formed and broken
Contract

- **Name**: name of the operation and parameters
- **Responsibilities**: an informal description of the responsibilities this operation must fulfill
- **Notes**: cross reference use case; exceptions etc
- **Output**: stating what information is sent out of the system Non UI outputs – keep the UI out of the description. For example state what data or messages are passed back
- **Pre-conditions**: assumptions about the state of the system before execution of the operation
- **Post-conditions**: the state of the system after completion of the operation
Guidelines continued

• Pre-conditions
  – what is needed for success of the operation
  *(what later on in the implementation we might think of testing in the method)*

• Post-conditions are expressed in terms of the conceptual model; the instances, attributes and associations are of this model

• This is analysis; pre and post-conditions will not be complete
Guidelines continued

So, the contract emphasises how the operation changes the internal state of the system … with outputs such as printing etc as secondary

The state is the attributes of the objects from the concept (domain) class diagram
UML notation

• Textual descriptions of contracts are preferred
• pre, post-conditions and output may be written in text/natural language and added as comments to UML

(constraints are text in braces {}; OCL (Object Constraint Language) is a UML extension for expressing constraints such as pre, post-conditions)
Basic contract info

• **Name:** name of the operation and parameters
• **Output** stating what information is sent out of the system
• **Pre-conditions:**
  – assumptions about the *state of the system* before execution of the operation
• **Post-conditions:**
  – the *state of the system* after completion of the operation