Object Oriented System Testing

Dafydd Vaughan

December 03 2006

CS339 Advanced Topics in Computer Science: Testing
Department of Computer Science, Swansea University
Outline

• System Testing
  - Overview of System Testing
  - Object Oriented System Testing

• Unified Modelling Language (UML)

• Test Case Generation Example
  - High Level Use Cases (HLUC)
  - Essential Use Cases (EUC)
  - Expanded Essential Use Cases (EEUC)
  - Real Use Cases (RUC)
  - System Test Cases (SysTC)

• Summary
• Outline

**System Testing**

• UML
• Test Case Generation Example
• Summary
System Testing

- System Testing
  - Tests the system as a whole
  - Concerned with what happens
  - Not how it happens
  - Black box
Object Oriented System Testing

• Almost identical to normal System Testing

• Except for generation of test cases

• UML can be used to generate test cases
• Outline
• System Testing
• **UML**
  • Test Case Generation Example
• Summary
UML (Unified Modelling Language)

- Standard for designing and modelling systems

- Consists of many different diagrams
  - Structure diagrams
    (Class, Object, Component etc.)
  - Behaviour diagrams
    (Use Case, Activity, State Machine etc.)
  - Interaction diagrams
    (Sequence, Communication, Timing etc.)

- Diagrams explain the specification of the system
UML (Unified Modelling Language)

- UML diagrams can be used to develop
  - Program Prototypes
  - Use Cases (Functional requirements)
  - Test Cases

- Potentially used for automated testing
  - Research area
• Outline
• System Testing
• UML

Test Case Generation Example

• Summary
Generating Test Cases

• Use UML description
  - Generate list of System Functions

• Generate & Expand Use Cases
  - High Level Use Cases (HLUC)
  - Essential Use Cases (EUC)
  - Expanded Essential Use Cases (EEUC)
  - Real Use Cases (RUC)

• Generate Test Cases from Real Use Cases
Currency Converter

• Example from Ian’s Talk (Chapter 19)

• Converts US Dollars to:
  - Brazilian real (R$)
  - Canadian dollars (C$)
  - European Union euros (€)
  - Japanese yen (¥)

• User can revise inputs

• User can perform repeated conversions
System Functions

- The functions of the system as the user describes them

- Developed from the UML Specification

- Identifies 3 types of functions:
  - Evident - Obvious to the user
  - Hidden - Not immediately obvious
  - Frill - “Bells and Whistles”
## System Functions

<table>
<thead>
<tr>
<th>Ref. No.</th>
<th>Function</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>Start application</td>
<td>Evident</td>
</tr>
<tr>
<td>R2</td>
<td>End application</td>
<td>Evident</td>
</tr>
<tr>
<td>R3</td>
<td>Input US dollar amount</td>
<td>Evident</td>
</tr>
<tr>
<td>R4</td>
<td>Select country</td>
<td>Evident</td>
</tr>
<tr>
<td>R5</td>
<td>Perform conversion calculation</td>
<td>Evident</td>
</tr>
<tr>
<td>R6</td>
<td>Clear user inputs and program outputs</td>
<td>Evident</td>
</tr>
<tr>
<td>R7</td>
<td>Maintain exclusive-or relationship along countries</td>
<td>Hidden</td>
</tr>
<tr>
<td>R8</td>
<td>Display country flag images</td>
<td>Frill</td>
</tr>
</tbody>
</table>
Use Cases

• Describe the functional requirements of a system

• Each Use Case describes a scenario

• Shows how the system should interact with the user (actor)

• Several levels of use cases
  - High Level
  - Essential
  - Expanded Essential
  - Real
High Level Use Cases (HLUC)

- Brief description of the main functions of the system

- High level view of program

- Very few details shown
  - Name of function
  - Actors involved
  - Type of use case
  - Description of function

- 2 types of use cases
  - Primary (essential & required)
  - Secondary (rarely occur / not required)
High Level Use Cases (HLUC)

<table>
<thead>
<tr>
<th>HLUC1</th>
<th>Start application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user starts the currency conversion application in Windows</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HLUC2</th>
<th>End application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user ends the currency conversion application in Windows</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HLUC3</th>
<th>Convert dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user inputs a US dollar amount and selects a country; the application computes and displays the equivalent in the currency of the selected country</td>
</tr>
</tbody>
</table>
Essential Use Cases (EUC)

• Identifies what user expects to happen

• Adds “actor” and “system” events to the HLUC

• Actions / Responses are numbered

• Numbers show approximate sequence in time
### Essential Use Cases (EUC)

- **HLUC1 → EUC1**

<table>
<thead>
<tr>
<th>HLUC1</th>
<th>Actor(s)</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User</td>
<td>Primary</td>
<td>The user starts the currency conversion application in Windows</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EUC1</th>
<th>Actor(s)</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User</td>
<td>Primary</td>
<td>The user starts the currency conversion application in Windows</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Actor action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1. The user starts the application, either with a Run command or by double-clicking the application icon</td>
<td>2. The currency conversion application GUI appears on the monitor and is ready for user input</td>
</tr>
</tbody>
</table>
### Essential Use Cases (EUC)

<table>
<thead>
<tr>
<th>EUC3</th>
<th>Convert dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user inputs a US dollar amount and selects a country; the application computes and displays the equivalent in the currency of the selected country</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Actor action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The user enters a dollar amount on the keyboard</td>
<td>2. The dollar amount is displayed on the GUI</td>
</tr>
<tr>
<td>3.</td>
<td>The user selects a country</td>
<td>4. The name of the country’s currency is displayed</td>
</tr>
<tr>
<td>6.</td>
<td>The user requests a conversion calculation</td>
<td>5. The flag of the country is displayed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7. The equivalent currency amount is displayed</td>
</tr>
</tbody>
</table>
What Next?

• So far:
  - Generated a list of system functions
  - Developed a set of HLUCs
  - Extended these to create a set of EUCs

• Now:
  - Create a detailed GUI definition

• Next:
  - Use this to generate the Expanded Essential Use Cases (EEUC)
  - Generate Real Use Cases (RUC)
  - Generate System Test Cases (SysTC)
Detailed GUI Definition

- Graphical User Interface designed
- Controls in design to be used in EEUCs
Expanded Essential Use Cases (EEUC)

- Next level of Use Case refinement
- Detailed description of processes involved
- Adds:
  - Pre / Post conditions
  - Alternative sequences of events
  - References system functions found earlier

- Also:
  - New use cases are identified and added at this point
  - More detail provides greater insight
### Expanded Essential Use Case

<table>
<thead>
<tr>
<th>EUC1</th>
<th>Start application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user starts the currency conversion application in Windows</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Actor action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The user starts the application, either with a Run command or by double-clicking the application icon</td>
<td>2. The currency conversion application GUI appears on the monitor and is ready for user input</td>
</tr>
</tbody>
</table>
### EEUC1

#### Actor(s)
User

#### Preconditions
Currency conversion application in storage

#### Type
Primary

#### Description
The user starts the currency conversion application in Windows

#### Sequence

<table>
<thead>
<tr>
<th>Actor action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user double-clicks currency conversion application icon</td>
<td>2. <code>frmCurrConv</code> appears on the screen</td>
</tr>
</tbody>
</table>

#### Alternative sequence
User opens currency conversion application within the Windows Run command

#### Cross-reference
R1

#### Postconditions
`ttxDollar` has focus
### Expanded Essential Use Cases (EEUC)

<table>
<thead>
<tr>
<th>EEUC3</th>
<th>Convert dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actor(s)</strong></td>
<td>User</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td><code>txtDollar</code> has focus</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Primary</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The user inputs a US dollar amount and selects a country; the application computes and displays the equivalent in the currency of the selected country</td>
</tr>
</tbody>
</table>

#### Sequence

<table>
<thead>
<tr>
<th><strong>Actor action</strong></th>
<th><strong>System response</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User enters a dollar amount on the keyboard</td>
<td>2. Dollar amount appears in <code>txtDollar</code></td>
</tr>
<tr>
<td>3. User clicks on a country button</td>
<td>4. Country currency name appears in <code>lblEquiv</code></td>
</tr>
<tr>
<td>5. User clicks <code>cmdCompute</code> button</td>
<td>6. Computed equivalent amount appears in <code>lblEqAmount</code></td>
</tr>
</tbody>
</table>

#### Alternative Sequence

Actions 1 and 3 can be reversed, and consequently responses 2 and 4 will be reversed.

#### Cross-reference

R3, R4, R5 and R8

#### Postconditions

`cmdClear` has focus
## System Functions Recap

<table>
<thead>
<tr>
<th>Ref. No.</th>
<th>Function</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>Start application</td>
<td>Evident</td>
</tr>
<tr>
<td>R2</td>
<td>End application</td>
<td>Evident</td>
</tr>
<tr>
<td>R3</td>
<td>Input US dollar amount</td>
<td>Evident</td>
</tr>
<tr>
<td>R4</td>
<td>Select country</td>
<td>Evident</td>
</tr>
<tr>
<td>R5</td>
<td>Perform conversion calculation</td>
<td>Evident</td>
</tr>
<tr>
<td>R6</td>
<td>Clear user inputs and program outputs</td>
<td>Evident</td>
</tr>
<tr>
<td>R7</td>
<td>Maintain exclusive-or relationship along countries</td>
<td>Hidden</td>
</tr>
<tr>
<td>R8</td>
<td>Display country flag images</td>
<td>Frill</td>
</tr>
</tbody>
</table>
Real Use Cases (RUC)

- RUC only slightly different from EEUC
  - Instead of “Enter dollar amount”
  - “Enter 10 in txtDollar” used
  - Etc.

- System Test Cases can be derived from RUC
### Real Use Cases (RUC) - EEUC3

<table>
<thead>
<tr>
<th>RUC3</th>
<th>Convert dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor(s)</td>
<td>User</td>
</tr>
<tr>
<td>Preconditions</td>
<td><code>txtDollar</code> has focus</td>
</tr>
<tr>
<td>Type</td>
<td>Primary</td>
</tr>
<tr>
<td>Description</td>
<td>The user inputs a US $10 and selects the European Community; the application computes and displays the equivalent: 7.50 euros</td>
</tr>
</tbody>
</table>

#### Sequence

<table>
<thead>
<tr>
<th>Actor action</th>
<th>System response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User enters 10 on the keyboard</td>
<td>2. 10 appears in <code>txtDollar</code></td>
</tr>
<tr>
<td>3. User clicks on the European Community button</td>
<td>4. Euros appears in <code>lblEquiv</code></td>
</tr>
<tr>
<td>5. User clicks <code>cmdCompute</code> button</td>
<td>6. 7.50 appears in <code>lblEqAmount</code></td>
</tr>
</tbody>
</table>

#### Alternative Sequence

Actions 1 and 3 can be reversed, and consequently responses 2 and 4 will be reversed.

#### Cross-reference

R3, R4, R5 and R8

#### Postconditions

`cmdClear` has focus
Finally

• So far:
  - Generated a list of system functions
  - Developed HLUC, EUC, Graphical Interface Design and EEUC
  - Created RUC

• Finally:
  - Convert RUC to System Test Cases (SysTC)
## System Test Cases (SysTC)

- **RUC3**
- **Test Operator**: Dafydd Vaughan
- **Preconditions**: `txtDollar` has focus
- **Type**: Primary
- **TO Sequence**

<table>
<thead>
<tr>
<th>Tester inputs</th>
<th>Expected system response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Enters 10 on the keyboard</td>
<td>2. Observe 10 appears in <code>txtDollar</code></td>
</tr>
<tr>
<td>3. Click on the European Community button</td>
<td>4. Observe europs appears in <code>lblEquiv</code></td>
</tr>
<tr>
<td>5. Click <code>cmdCompute</code> button</td>
<td>6. Observe 7.50 appears in <code>lblEqAmount</code></td>
</tr>
</tbody>
</table>

- **Postconditions**: `cmdClear` has focus
- **Test Result**: Pass/Fail
- **Date Run**: December 03, 2006
Further Info

- Possible automation of Use Cases
- Automation of System Testing is being researched
- UML assists in development of test cases
• Outline
• System Testing
• UML
• Test Case Generation Example

Summary
Summary

- **System Testing**
  - Tests the system as a whole
  - OO System Testing identical to normal System Testing

- **Unified Modelling Language (UML)**
  - Used to describe OO applications
  - Used to create test cases

- **Use Cases**
  - High Level Use Cases (HLUC)
  - Essential Use Cases (EUC)
  - Expanded Essential Use Cases (EEUC)
  - Real Use Cases (RUC)
  - System Test Cases (SysTC)
Object Oriented System Testing

Dafydd Vaughan

December 03 2006

CS339 Advanced Topics in Computer Science: Testing
Department of Computer Science, Swansea University