



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



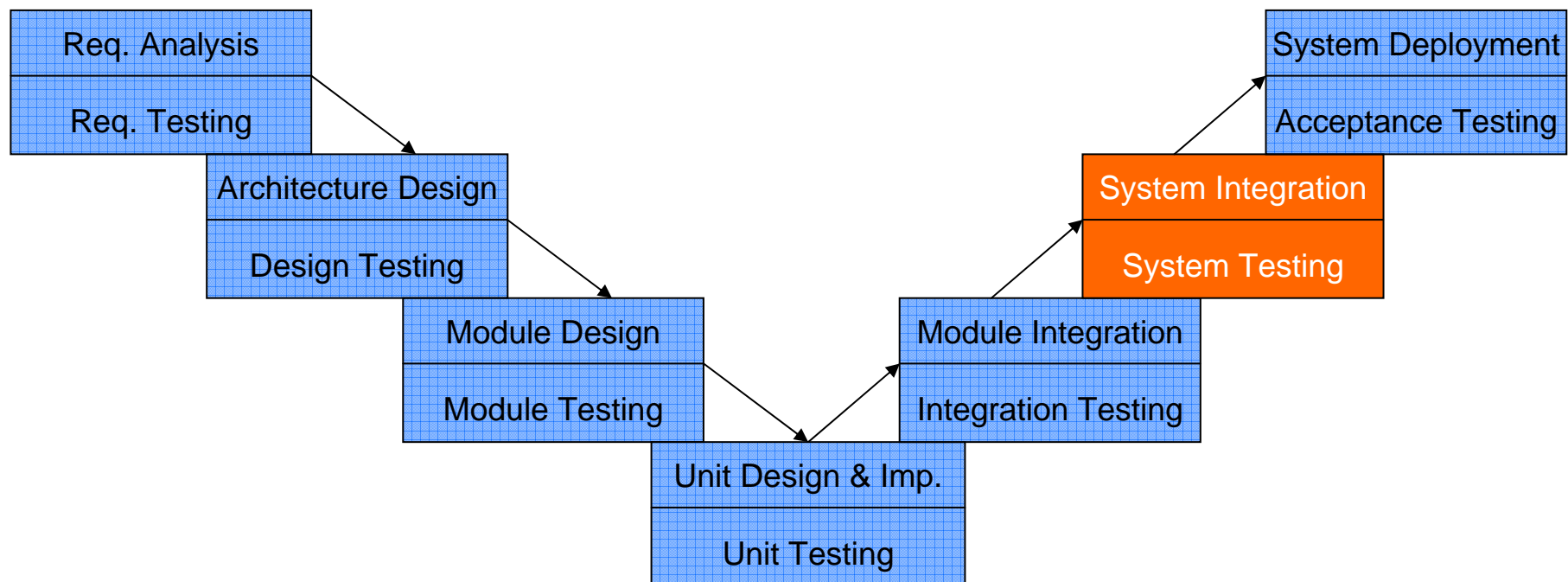
System Testing

- 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

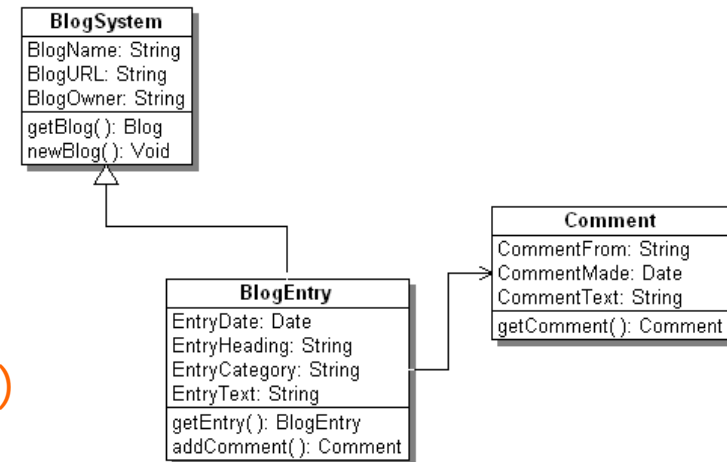


UML (Unified Modelling Language)

- 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

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



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)

<i>HLUC1</i>	Start application
<i>Actor(s)</i>	User
<i>Type</i>	Primary
<i>Description</i>	The user starts the currency conversion application in Windows

<i>HLUC2</i>	End application
<i>Actor(s)</i>	User
<i>Type</i>	Primary
<i>Description</i>	The user ends the currency conversion application in Windows

<i>HLUC3</i>	Convert dollars
<i>Actor(s)</i>	User
<i>Type</i>	Primary
<i>Description</i>	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



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

<i>HLUC1</i>	Start application
<i>Actor(s)</i>	User
<i>Type</i>	Primary
<i>Description</i>	The user starts the currency conversion application in Windows



<i>EUC1</i>	Start application	
<i>Actor(s)</i>	User	
<i>Type</i>	Primary	
<i>Description</i>	The user starts the currency conversion application in Windows	
<i>Sequence</i>	<i>Actor action</i>	<i>System response</i>
	1. The user starts the application, either with a Run command or by double-clicking the application icon	2. The currency conversion application GUI appears on the monitor and is ready for user input



Essential Use Cases (EUC)

<i>EUC3</i>	Convert dollars	
<i>Actor(s)</i>	User	
<i>Type</i>	Primary	
<i>Description</i>	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	
<i>Sequence</i>	<i>Actor action</i>	<i>System response</i>
	1. The user enters a dollar amount on the keyboard 3. The user selects a country 6. The user requests a conversion calculation	2. The dollar amount is displayed on the GUI 4. The name of the country's currency is displayed 5. The flag of the country is displayed 7. The equivalent currency amount is displayed



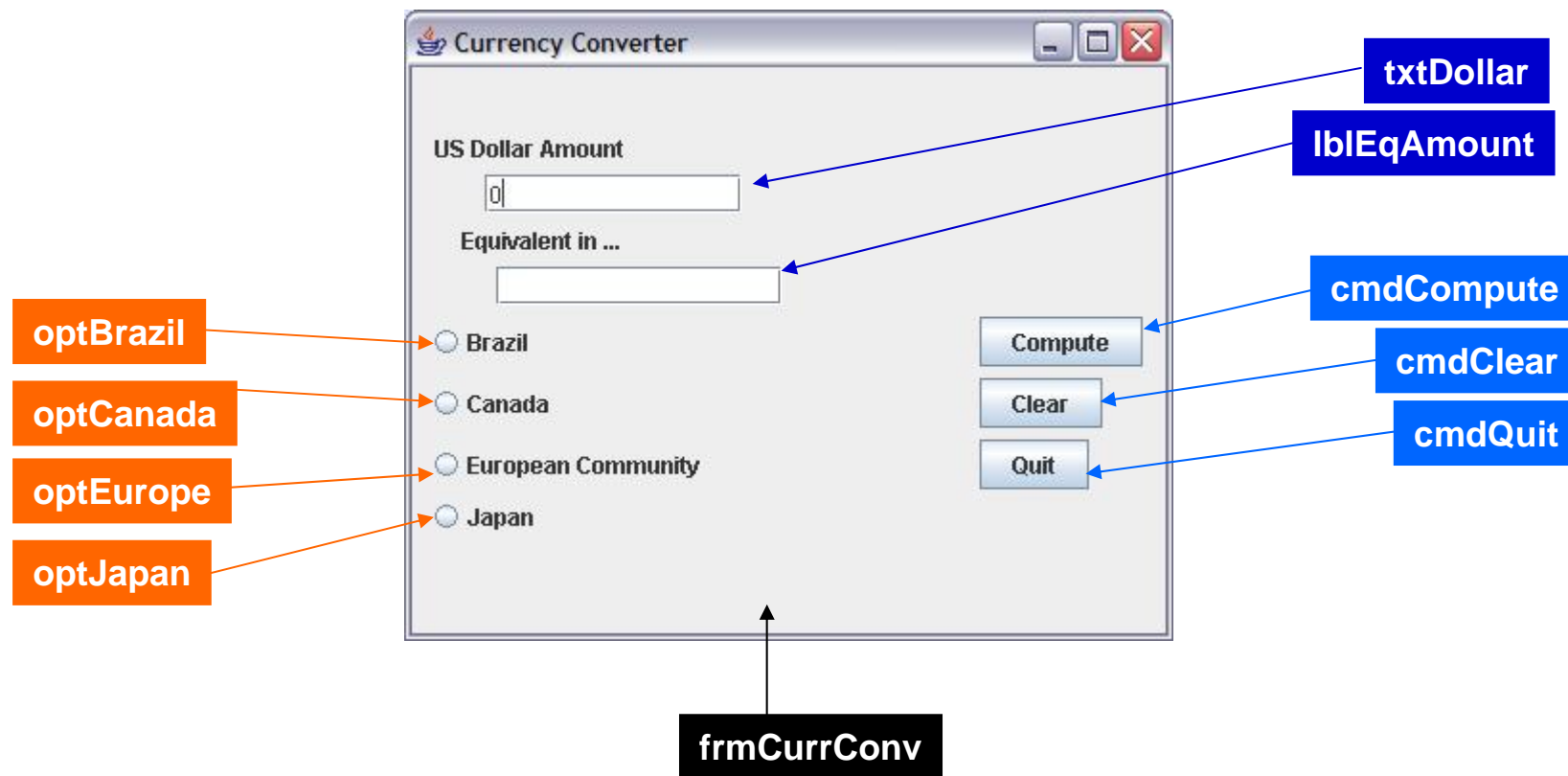
What Next?

- 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 Cases (EEUC)

<i>EUC1</i>	Start application	
<i>Actor(s)</i>	User	
<i>Type</i>	Primary	
<i>Description</i>	The user starts the currency conversion application in Windows	
<i>Sequence</i>	<i>Actor action</i>	<i>System response</i>
	1. The user starts the application, either with a Run command or by double-clicking the application icon	2. The currency conversion application GUI appears on the monitor and is ready for user input



Expanded Essential Use Case



Expanded Essential Use Cases (EEUC)

Essential Use Case



<i>EEUC1</i>	Start application				
<i>Actor(s)</i>	User				
<i>Preconditions</i>	Currency conversion application in storage				
<i>Type</i>	Primary				
<i>Description</i>	The user starts the currency conversion application in Windows				
<i>Sequence</i>	<table border="1"><thead><tr><th><i>Actor action</i></th><th><i>System response</i></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>	<i>Actor action</i>	<i>System response</i>	1. The user double-clicks currency conversion application icon	2. <code>frmCurrConv</code> appears on the screen
<i>Actor action</i>	<i>System response</i>				
1. The user double-clicks currency conversion application icon	2. <code>frmCurrConv</code> appears on the screen				
<i>Alternative sequence</i>	User opens currency conversion application within the Windows Run command				
<i>Cross-reference</i>	R1				
<i>Postconditions</i>	<code>txtDollar</code> has focus				



Expanded Essential Use Cases (EEUC)

<i>EEUC3</i>	Convert dollars								
<i>Actor(s)</i>	User								
<i>Preconditions</i>	<code>txtDollar</code> has focus								
<i>Type</i>	Primary								
<i>Description</i>	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								
<i>Sequence</i>	<table border="1"><thead><tr><th><i>Actor action</i></th><th><i>System response</i></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>	<i>Actor action</i>	<i>System response</i>	1. User enters a dollar amount on the keyboard	2. Dollar amount appears in <code>txtDollar</code>	3. User clicks on a country button	4. Country currency name appears in <code>lblEquiv</code>	5. User clicks <code>cmdCompute</code> button	6. Computed equivalent amount appears in <code>lblEqAmount</code>
<i>Actor action</i>	<i>System response</i>								
1. User enters a dollar amount on the keyboard	2. Dollar amount appears in <code>txtDollar</code>								
3. User clicks on a country button	4. Country currency name appears in <code>lblEquiv</code>								
5. User clicks <code>cmdCompute</code> button	6. Computed equivalent amount appears in <code>lblEqAmount</code>								
<i>Alternative Sequence</i>	Actions 1 and 3 can be reversed, and consequently responses 2 and 4 will be reversed								
<i>Cross-reference</i>	R3, R4, R5 and R8								
<i>Postconditions</i>	<code>cmdClear</code> has focus								



System Functions Recap

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



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



<i>RUC3</i>	Convert dollars								
<i>Actor(s)</i>	User								
<i>Preconditions</i>	<code>txtDollar</code> has focus								
<i>Type</i>	Primary								
<i>Description</i>	The user inputs a US \$10 and selects the European Community; the application computes and displays the equivalent: 7.50 euros								
<i>Sequence</i>	<table border="1"><thead><tr><th><i>Actor action</i></th><th><i>System response</i></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>	<i>Actor action</i>	<i>System response</i>	1. User enters 10 on the keyboard	2. 10 appears in <code>txtDollar</code>	3. User clicks on the European Community button	4. Euros appears in <code>lblEquiv</code>	5. User clicks <code>cmdCompute</code> button	6. 7.50 appears in <code>lblEqAmount</code>
<i>Actor action</i>	<i>System response</i>								
1. User enters 10 on the keyboard	2. 10 appears in <code>txtDollar</code>								
3. User clicks on the European Community button	4. Euros appears in <code>lblEquiv</code>								
5. User clicks <code>cmdCompute</code> button	6. 7.50 appears in <code>lblEqAmount</code>								
<i>Alternative Sequence</i>	Actions 1 and 3 can be reversed, and consequently responses 2 and 4 will be reversed								
<i>Cross-reference</i>	R3, R4, R5 and R8								
<i>Postconditions</i>	<code>cmdClear</code> 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



<i>RUC3</i>	Convert dollars	
<i>Test Operator</i>	Dafydd Vaughan	
<i>Preconditions</i>	<code>txtDollar</code> has focus	
<i>Type</i>	Primary	
<i>TO Sequence</i>	<i>Tester inputs:</i>	<i>Expected system response:</i>
	1. Enters 10 on the keyboard 3. Click on the European Community button 5. Click <code>cmdCompute</code> button	2. Observe 10 appears in <code>txtDollar</code> 4. Observe europs appears in <code>lblEquiv</code> 6. Observe 7.50 appears in <code>lblEqAmount</code>
<i>Postconditions</i>	<code>cmdClear</code> has focus	
<i>Test Result</i>	Pass/Fail	
<i>Date Run</i>	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