Coverage Criteria for GUI Testing

Atif M. Memon*, Mary Lou Soffa, Martha E. Pollack
atif@cs.umd.edu
Dept. of Computer Science
University of Pittsburgh

*now at the University of Maryland
Research focus

Interactions between the GUI and the underlying code

GUI

Underlying Code

50% of code
GUI Test Case

• **Sequence of Events**
  - [IEEE TSE Feb '01]

• **Not just individual events**
Coverage Criteria

• Two purposes
  - Test data selection criteria
    • Rules used to select test cases
  - Test data adequacy criteria
    • Rules used to determine how much testing has been done

• Common Examples for Conventional Software
  - Statement coverage
  - Branch coverage
  - Path coverage

  Structural Representation of the Code
Coverage Criteria for GUIs

• Cannot use code-based coverage
  - Source code not always available
  - Event-based input
    • Different level of abstraction

• Our Contribution
  - Hierarchical structure of the GUI in terms of events
  - Coverage criteria based on events
Role of the Coverage Criteria

GUI Specifications

GUI Implementation: Tools (Languages/Toolkits)

GUI Implementer

Executing GUI

Coverage Criteria

Regression Tester

Test Coverage Evaluator

Test Oracle

Test Executor

Test Case Generator

GUI Representation
Outline

- **GUI Definition**
- **Representation of GUIs**
- **Coverage Criteria**
- **Case Study**
- **Conclusions**
GUI Definition

- Hierarchical
- Graphical Front-end
- Accepts User-generated and System-generated events
- Fixed sets of events
- Deterministic Output
- State of the GUI is the set of Objects and their Properties
GUI Representation

- **Motivation**
  - GUI testing needs a “Unit of Testing”
    - Manageable
    - Test the unit comprehensively
    - Test interactions among units
  - **GUIs are created using library elements**
    - Need to test these elements before packaging them for reuse
      - Certain level of confidence that the element has been adequately tested
    - User of these elements should be able to test the element in its context of use
Model GUI Hierarchically

- **Hierarchy**
  - GUIs are decomposed into a hierarchy of components
  - Hierarchical decomposition makes testing intuitive and efficient
  - Several hierarchical views of GUIs
  - We examine Modal Dialogs to create the hierarchical model
Modal Windows in GUIs
Modal Windows in GUIs

- **Main** invokes **Print**
Modal Windows in GUIs
**Definition:** Integration tree is a triple \( \langle N, R, B \rangle \)
- \( N \) is the set of components in the GUI
- \( R \) ? \( N \) is a designated component called the **Main** component
- \( B \) is the set of directed edges showing the invokes relation between components, i.e., \( (C_x, C_y) \in B \) iff \( C_x \) invokes \( C_y \).
Definition: Event $e_x$ follows $e_y$ iff $e_x$ can be performed immediately after $e_y$. 
**Definition:** Event-flow graph is a 4-tuple $\langle V, E, B, I \rangle$

- $V$ is the set of vertices, representing events,
- $E$ is the set of directed edges, showing the follows relationship,
- $B$ is the set of events first available (shown in red),
- $I$ is the set of events that invoke other components (dotted lines).
Classifying Events

• Classification

- A new classification of events aids in creating the hierarchical model of the GUI
  - Opening modal windows
    - Restricted-focus events
  - Closing modal windows
    - Termination events
  - Opening modeless windows
    - Unrestricted-focus events
  - Opening menus
    - Menu-open events
  - Interacting with underlying software
    - System-interaction events
Coverage Criteria

• **Intuitively**
  
  - Each component is a unit of testing
  - Test events within each component
    - Intra-component coverage criteria
  - Test events across components
    - Inter-component coverage criteria
Coverage Criteria

• **Intra-component Coverage**
  - **Event coverage**
    • Individual events
    • Each node in the event-flow graph
  - **Event-interaction coverage**
    • Each pair of events
    • Each edge in the event-flow graph
  - **Length-n event sequence coverage**
    • Sequences of events
    • Bounded by length
      - Length-1 event sequences
      - Length-2, length-6 event sequences
    • Paths in the event-flow graph
Coverage Criteria

• **Inter-component Coverage**
  
  - **Invocation coverage**
    - Invoke each component
    - Each restricted-focus event
  
  - **Invocation-termination coverage**
    - Invoke each component and terminate it
    - Restricted-focus event followed by a termination event
  
  - **Inter-component length-n coverage**
    - Longer sequences from one component to another
    - Bounded by length
Case Study

- **Purpose**
  - To determine:
    - How many test cases do we need to test WordPad
    - Correlation between event and code-based coverage
    - How well did our planning-based approach [ICSE '99] do

- **Experimental design**
  - **GUI**: our version of MS WordPad (36 modal windows, 362 events)
  - **Hardware platform**: 350 MHz Pentium based machine, 256 MB RAM
## Test Cases for WordPad

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Event-sequence Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1' 2' 1 2 3 4 5 6</td>
</tr>
<tr>
<td><strong>Main</strong></td>
<td>56 791 14354 255720</td>
</tr>
<tr>
<td></td>
<td>4490626 78385288</td>
</tr>
<tr>
<td><strong>FileOpen</strong></td>
<td>10 80 640 5120 40960</td>
</tr>
<tr>
<td></td>
<td>327680</td>
</tr>
<tr>
<td><strong>FileSave</strong></td>
<td>10 80 640 5120 40960</td>
</tr>
<tr>
<td></td>
<td>327680</td>
</tr>
<tr>
<td><strong>Print</strong></td>
<td>12 108 972 8748 78732</td>
</tr>
<tr>
<td></td>
<td>708588</td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td>13 143 1573 17303</td>
</tr>
<tr>
<td></td>
<td>190333 2093663</td>
</tr>
<tr>
<td><strong>PageSetup</strong></td>
<td>11 88 704 5632 45056</td>
</tr>
<tr>
<td></td>
<td>360448</td>
</tr>
<tr>
<td><strong>FormatFont</strong></td>
<td>9 63 441 3087 21609</td>
</tr>
<tr>
<td></td>
<td>151263</td>
</tr>
<tr>
<td><strong>Print+Properties</strong></td>
<td>1 2 13 260 3913</td>
</tr>
<tr>
<td></td>
<td>52520 663013</td>
</tr>
<tr>
<td><strong>Main+FileOpen</strong></td>
<td>1 2 10 100 1180</td>
</tr>
<tr>
<td></td>
<td>17160 278760</td>
</tr>
<tr>
<td><strong>Main+FileSave</strong></td>
<td>1 2 10 100 1180</td>
</tr>
<tr>
<td></td>
<td>17160 278760</td>
</tr>
<tr>
<td><strong>Main+PageSetup</strong></td>
<td>1 2 11 110 1298</td>
</tr>
<tr>
<td></td>
<td>18876 306636</td>
</tr>
<tr>
<td><strong>Main+FormatFont</strong></td>
<td>1 2 9 81 909</td>
</tr>
<tr>
<td></td>
<td>13311 220509</td>
</tr>
<tr>
<td><strong>Main+Print+Properties</strong></td>
<td>12 145 1930 28987</td>
</tr>
</tbody>
</table>
Correlation between Event-based & Code-based Coverage

- Code Instrumentation
- Generated all event sequences up to length 3. Total test cases: 21,659
- Executed all 21,659 cases and obtained execution traces
- Statement coverage
Correlation between Event-based & Code-based Coverage

Results
Evaluating the Planning Approach

- Used our earlier-developed planning-based approach
- 500 test cases of different lengths
## Evaluating the Planning Approach

### Component Name

<table>
<thead>
<tr>
<th>Component Name</th>
<th>1'</th>
<th>2'</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td>88</td>
<td>41</td>
<td>10.92</td>
<td>0.36</td>
<td>0.03</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FileOpen</td>
<td>90</td>
<td>56</td>
<td>17.50</td>
<td>0.72</td>
<td>0.06</td>
<td>0.05</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FileSave</td>
<td>90</td>
<td>41</td>
<td>20.63</td>
<td>1.27</td>
<td>0.47</td>
<td>0.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print</td>
<td>92</td>
<td>34</td>
<td>32.20</td>
<td>9.00</td>
<td>3.92</td>
<td>0.19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Properties</td>
<td>92</td>
<td>45</td>
<td>27.59</td>
<td>1.80</td>
<td>0.97</td>
<td>0.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PageSetup</td>
<td>91</td>
<td>49</td>
<td>25.43</td>
<td>2.56</td>
<td>0.66</td>
<td>0.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FormatFont</td>
<td>89</td>
<td>37</td>
<td>39.00</td>
<td>13.67</td>
<td>0.66</td>
<td>0.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print+Properties</td>
<td>100</td>
<td>0</td>
<td>46</td>
<td>51.15</td>
<td>8.18</td>
<td>3.87</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>Main+FileOpen</td>
<td>100</td>
<td>0</td>
<td>40</td>
<td>11.00</td>
<td>10.17</td>
<td>1.30</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>Main+FileSave</td>
<td>100</td>
<td>0</td>
<td>20</td>
<td>13.00</td>
<td>8.64</td>
<td>1.26</td>
<td>0.28</td>
<td></td>
</tr>
<tr>
<td>Main+PageSetup</td>
<td>100</td>
<td>0</td>
<td>45</td>
<td>60.91</td>
<td>4.31</td>
<td>1.94</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td>Main+FormatFont</td>
<td>100</td>
<td>0</td>
<td>33</td>
<td>28.40</td>
<td>5.17</td>
<td>0.97</td>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td>Main+Print+Properties</td>
<td>50</td>
<td>38.62</td>
<td>6.37</td>
<td>0.65</td>
<td>0.09</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Results
Future Work

• GUI’s Structure and its Testability
• Apply Criteria to
  - Object-oriented Software
  - Component-based Software
  - Reactive Software