GUI Testing

Dr. Mark C. Paulk

SE 4367 – Software Testing, Verification, Validation, and Quality Assurance

The Need for GUI Testing

GUIs can constitute as much as 60 percent of an application's total code today.

- ***A.M. Memon, "GUI Testing: Pitfalls and Process," IEEE Computer, August 2002.***

Conventional testing techniques to not apply directly to GUIs.

- ***A.M. Memon, M.L. Soffa, and M.E. Pollack, "Coverage Criteria for GUI Testing," ESEC/FSE 2001, Vienna, Austria, 2001.***

GUI Test Tools

Current GUI testing techniques are incomplete, ad hoc, and largely manual.

Most common tools use record-playback techniques.

- **A test designer interacts with the GUI, generating mouse and keyboard events.**
- **The tool records the user events, captures the GUI session screens, and then stores the session — usually as a script.**
- **The tester later plays back the recorded sessions to re-create the events with different inputs.**

GUI Testing and FSMs

(Mathur, Chapter 5)

You could model the graphical user interface with a finite state machine.

- **use FSM test generation techniques**

Windows could be represented by states.

Menu options could be represented by transitions.

Depending on the GUI design, some parts of the input alphabet would not apply to some states.

GUI Design and Equivalence Classes
Mathur, Section 3.3.7

GUI may offer only correct choices via menu.

GUI may ask the user to fill in a box.
- **illegal values are possible**

Test design must take into account GUI design.

Makes the assumption the GUI has been correctly implemented.

GUI Oracles

Verifying whether the GUI executes correctly poses a problem.

The traditional verification tool is a test oracle.

A GUI test case requires interleaving the oracle invocation with the test case execution

- **an incorrect GUI state can lead to an unexpected screen, which in turn can make further test case execution useless**
- **a GUI test case should terminate as soon as the oracle detects an error**

GUI Coverage Criteria

Decompose a GUI into a set of GUI components, each of which is used as a basic unit of testing.

An event-flow graph represents the interaction of events within a component.

Intra-component criteria are used to evaluate the adequacy of tests on these events.

The hierarchy of relationships among components is represented by an integration tree.

- **inter-component coverage criteria are used to evaluate test adequacy for sequences that cross components**

Modal Windows

A modal window is a GUI window that once invoked, monopolizes the GUI interaction, restricting the focus of the user to a specific range of events within the window, until the window is explicitly terminated.

The events within the modal dialog form a GUI component.

- **A GUI component C is an ordered pair (RF, UF), where RF represents a modal window in terms of its events and UF is a set whose elements represent modeless windows also in terms of their events. Each element of UF is invoked either by an event in UF or RF.**

Event Flow Graphs

An event flow graph for a GUI component C is a 4-tuple <V, E, B, I> where:

- **V is a set of vertices representing all the events in the component. Each v in V represents an event in C.**
- **E, a subset of V x V, is a set of directed edges between vertices.**
  - **We say that event e_i follows e_j iff e_j may be performed immediately after e_i. An edge (v_x,v_y) in E iff the event represented by v_y follows the event represented by v_x.**
- **B, a subset of V, is a set of vertices representing those events of C that are available to the user when the component is first invoked.**
- **I, a subset of V, is the set of restricted-focus events of the component.**

Event Coverage

A set P of event-sequences satisfies the event coverage criterion if and only if for all events v in V, there is at least one event-sequence p in P such that event v is in p.

A set P of event-sequences satisfies the event-interaction coverage criterion if and only if for all elements (e_i, e_j) in E, there is at least one event-sequence p in P such that p contains (e_i, e_j).

A set P of event-sequences satisfies the length-n event-sequence coverage criterion if and only if P contains all event-sequences of length equal to n.

Invocation Coverage

A set P of event-sequences satisfies the invocation coverage criterion if and only if for all restricted-focus events i in I, where I is the set of all restricted-focus events in the GUI, there is at least one event sequence p in P such that event i is in p.

The invocation-termination set IT of a GUI consists of all possible length 2 event sequences $< e_i, e_j >$, where e_i invokes component C and e_j terminates component C, for all components C in N.

A set P of event sequences satisfies the invocation-termination coverage criterion if and only if for all i in IT, there is at least one event-sequence p in P such that i is in p.

Summary – Things to Remember

GUIs can constitute the majority of an application's code

Most GUI testing tools use record-playback

GUIs can be modeled using finite state machines

Questions and Answers