

Testing Graphical User Interfaces

Jeff Offutt

<http://www.cs.gmu.edu/~offutt/>

SWE 632

User Interface Design

Types of GUI Testing

- Two types :
 1. Usability Testing : Does it satisfy usability requirements?
 2. Functional Testing : Does it satisfy functional requirements?
- Most of this lecture is about functional testing

GUI Usability Testing

- Gathering information
 - Get real usage data
 - Use video cameras to watch users
- What to look for
 - Capture user actions
 - Capture common actions (to optimize)
 - Look for common errors (to add more avoidance)
 - Look for “menu browsing” (or “screen browsing”)
 - lots of exploration with no actions
 - this may indicate poor menu structure
- Use independent testers – especially for interfaces
 - “Familiarity breeds content”

22-Feb-11

© Jeff Offutt, 2004-2011

3

GUI Functional Testing

- Must have separate list of requirements for GUI
- Develop test cases :
 - Cover actions and events
 - Include expected results
- Compare expected results with actual results
- Aspects of functional testing
 - Automation
 - Test design

22-Feb-11

© Jeff Offutt, 2004-2011

4

GUI Capture / Replay Tools

- Capture/replay tools are widely used tools for regression testing
 - Regression testing is checking whether changes were made correctly
- Very useful automation
- But don't help with the hard intellectual part – designing test cases

22-Feb-11

© Jeff Offutt, 2004-2011

5

Capture / Replay Tools History

1. Hand testing
2. First generation tools
3. Second generation tools
4. Emerging technologies

22-Feb-11

© Jeff Offutt, 2004-2011

6

C/R – Hand Testing

- People sit at terminals
- Type and take notes
- Problems:
 - Hard to repeat failures
 - Testing was serialized – one component at a time
 - No notion of coverage and no rigor or well-defined process
 - Very labor expensive (inefficient)

22-Feb-11

© Jeff Offutt, 2004-2011

7

C/R – First Generation Tools

- Replicate the hand testing process
- Record activities and play back later to compare with new revisions
- Bitmap snapshots were taken at regular intervals
 - “bitmap-verified record & playback”
- Fewer testers and more repeatability

22-Feb-11

© Jeff Offutt, 2004-2011

8

Problems with First Generation Tools

- Automates the weak hand process
 - Application must be complete, testing must follow development
 - Some background components are hard to test
- Verifying screens match on a pixel level
 - Small changes to font, color, or location show up as errors
- Changes in application's appearance show up as errors
- Small changes make existing tests invalid
 - Updates tests or create a new test ?
- All the bitmaps take huge amounts of storage

22-Feb-11

© Jeff Offutt, 2004-2011

9

C/R – Second Tools

- Use position independence

movePtr (10,10)	}	pressButton ("filter")
click		pressButton ("ok")
movePtr (20,20)		
click		

- Application must be available
- Much of testing is independent of implementation
- Pixel matching: Test tools add optical character recognition and fuzzy bitmap comparisons
 - Slow and memory intensive

22-Feb-11

© Jeff Offutt, 2004-2011

10

Emerging Technology

- Capture / Replay tools reached a dead end
- New tools bringing in modern research techniques :
 - Test logical view of the software
 - Test scripting abilities – easier to make changes
 - Specify the entire sequence of inputs

Test tools should depend on the interface, not the underlying objects – if the implementation changes, the GUI tester should not notice the difference

C/R Application Hints

- Use very small test cases
 - Changes affect fewer tests
- Compress screen dumps
- Run comparisons separately from captures

GUI Test Design

The intellectually challenging part is deciding what to test

1. Every action and event
 - Buttons, menu items, etc
2. All sequences
 - This may be infinite ☹
3. Define typical scenarios – sequences of events
 - This requires careful judgment and understanding of the users
 - UML use cases can provide an excellent starting point
4. Base testing on a transition diagram ...

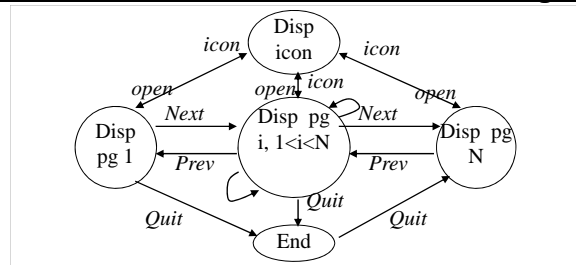
22-Feb-11

© Jeff Offutt, 2004-2011

13

GUI Transition Diagram Testing

Acrobat Reader State-Transition Diagram



- Cover every transition (arc)
- Cover “reasonable paths”
- Try invalid transitions
 - Prev in “Disp pg 1”
 - Next in “End”

22-Feb-11

© Jeff Offutt, 2004-2011

14

General GUI Testing Hints

1. Determine usability goals early
 - Must be specific
 - Must be testable
 - As part of requirements
2. Evaluate each *interface point* separately
 - Command, option, menu choice, button, ...
 - Compare with documentation
 - Measure against requirements & goals
 - Cannot be done by developer!

22-Feb-11

© Jeff Offutt, 2004-2011

15

General GUI Testing Hints

3. Define and carry out semantically meaningful, complete tasks
 - Depends on application
 - Separate this from step 2
 - Measure against requirements & goals
4. Perform extensive consistency checks
5. Acceptance testing is crucial
 - Get a “real” user to use the system
 - Carefully track each choice and especially mistakes
 - Track automatically if possible – less intrusive

22-Feb-11

© Jeff Offutt, 2004-2011

16

GUI Testing Summary

- Our knowledge is still fairly weak
- Tool support is primitive
- Research is in the early days
- Most companies do most of their testing manually
 - Inefficient
 - Ineffective