First Experience Teaching Software Testing
Lessons Learned

Gary Pollice
Worcester Polytechnic Institute and Rational Software Corp.
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
CS525T: Fall 2002

- Graduate course: *Software Testing*
  - http://www.cs.wpi.edu/~gpollice/cs525t-f02/
- 14 weeks
- Text: *A Practical Guide to Testing Object-Oriented Software*, McGregor and Sykes

Goals
- General understanding of testing principles
- The role of testing in modern SDLCs
- Experience testing software and using tools
- Prepare students for further research
CS525T: Fall 2002

- **Students**
  - 28 graduate students
  - No previous testing courses

- **Student motivation**
  - Will help entry into industry
  - Needed a project course

- **Department motivation**
  - Add to software engineering offerings
CS525T: Fall 2002

- Two projects
  - Small teams, 2-5 students
  - Evaluate a testing tool
    - Commercial or non-commercial
    - Prepare an evaluation report (template provided)
    - No two teams could do the same tool
  - Test an open-source product
    - radical Java GUI builder from SourceForge
    - Several types of testing (left mostly up to the team)
    - Prepare a software readiness report
CS525T: Fall 2002

- Templates provided for several artifacts
  - Tool evaluation report
  - Test plan
  - Defect report
  - Test report
  - Simple test cases
  - JUnit TestCase and TestSuite files

- Several templates taken from, or adapted from the Rational Unified Process®
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
Black Box Testing

- Easiest for students to understand
- Simple examples
  - Triangle problem
  - Roman numeral converter
- Easy to grasp the concept of test cases
- It was fun for the students
  - Free pizza challenge
Project 1: Tool Evaluation

- Goals:
  - Understand something about a “real” testing tool
  - Learn how to evaluate a tool for its appropriateness to different situations
  - Learn how to use the tool

- Template provided
Project 1: Results

- Ten reports produced
  - One ended up on tool vendor’s Web pages
- What I didn’t do
  - Provide enough time for presentation and discussion
(Semi)Formal Methods

- Used OCL to specify software and develop constraint test cases
  - Focus on the technique rather than strict adherence to the formal method
- Homework assignment: Develop OCL specifications for Java TreeMap class and implement tests in JUnit
Formal Methods Results

- Coupled theory and practice
- Helped develop good techniques for test implementation
- Common errors
  - Using a method to test itself
  - Making tests too specific
- Grading OCL took a long time
- Grading the JUnit tests was easy
Examples
Examples
Examples

/**
 * Test the containsKey method.
 * 
 * TreeMap:::containsKey(obj : Object) : boolean
 * Pre: true
 * Post: result = self->keySet().contains(obj)
 * 
 * Test Cases:
 * (self->containsKey(obj), true)
 * (!self->containsKey(obj)), false)
 */

public void testContainsKey()
{
    System.out.print("\n\ntestContainsKey: ");
    
    TreeMap map = makeTreeMap();
    // Test Case 1
    Object obj = new Integer(1);
    boolean result = map.keySet().contains(obj);
    Assert.assertEquals(map.containsKey(obj), result);
    Assert.assertTrue(result);
    // Test Case 2
    obj = new Integer(99);
    result = map.keySet().contains(obj);
    Assert.assertEquals(map.containsKey(obj), result);
    Assert.assertFalse(result);
}
Project 2: Test Real Software

- Challenges
  - Where do you get the software
  - Like the 3 Bears’ porridge, not too easy, not too hard, but just right
  - Can you test multiple builds / iterations?

- Open source provides a lot of possibilities
  - Over 7000 projects on SourceForge in Java
  - We used the Radical Java GUI builder project
Project 2: Minimum Deliverables

- Test plan
- Coverage of at least 80% line coverage
- Use case test
  - Students had to write the use case
- UI testing
- Exploratory testing
- Defect Report
Project 2: Results

- All teams achieved coverage goals
- Automated GUI testing was not effective
  - Only a single snapshot so record/playback was not necessary
- Number of defects reported was low
  - Largest number was 16
  - Lowest number was 5
- Students learned coverage tool well
- Some students worried more about form than results
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
Testing in an Iterative Process

- You need iterations and the materials that go with them
- Learning about iterations and process is not the same as participating in an iterative project
Iterative Development Testing

Code

Tests
Building Test Tools

- Significant part of the software tester’s job
- Not enough time
  - Perhaps a separate course on toolsmithing, not just test tools
- Emphasizing the need for good development skills for testers
Miscellaneous Disasters

- OATS
  - Difficult to motivate
- Distribution
  - Lack of resources
- Product lines and frameworks
  - Time and resources
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
Teaching Resources

- Notes and slides
- Recommended readings
- Techniques
Tool Resources

- Commercial
- Free
- Where to get them
- Applicability
- Reviews
Software Resources

- The software testing tutorial resources!
  - Code
  - Models
  - Multiple technologies
- Allow instructors to mix-and-match
- Open source projects may be a place to mine
  - Current individual study at WPI looking into this
Agenda

- Background
- Things that worked
- Things that didn’t work
- Wish list
- Questions
Thank You!

gpollice@cs.wpi.edu

gpollice@rational.com