CSC 405, Week 3

Requirements and Acceptance Test Discussions,

(Postponed from Week 2)

Overview of Testing Concepts

I. Weekly Lecture/Lab Overview

A. *Monday*:

1. Individual requirements reports
2. Individual acceptance testing reports
3. Details of phase 1 work assignment

B. ***Wednesday:***

1. Further review of requirements and acceptance test reports.

2. Lab time for development and testing

C. ***Friday:***

1. Overview of Salome’s automated testing
2. Overview of testing process overall
3. Lab time for development and testing

II. **Discussion of Requirements Updates**

A. Postponed from week 2

B. Everyone summarizes their rqmts updates.

C. Note any issues requiring group discussion.

D. Finalize phase 1 features list.

III. **Discussion of Initial Acceptance Testing**

A. Postponed from week 2.

B. Everyone summarizes their testing.

C. Note any issues requiring group discussion.

D. Finalize phase 1 must-fix list.

IV. **Phase 1 work assignments**

A. See 40266 wiki table for summary.

B. See Jira postings from E & K for details.

C. Discuss specific work details during lab.

V. Current Status of Phase 1 Feature List

A. Brief overview from K & E.

B. Discussion of any serious existing roadblocks at the moment.

VI. **Current Automated Testing**

A. Oveview from Salome.

B. Brief discussion.

VII. Overview of general testing concepts

A. In lecture, cover conceptual and theoretical details of software testing.

B. In lab, come up with testing framework(s) and tools to use for testing the project.

VIII. A bit of testing motivation.

A. Last decade has seen highly significant shift in the industry mindset on testing.

B. A number of good studies provide evidence that testing can be very cost effective.

Motivation, cont'd

C. Emphasis placed on testing in industrial settings is likely to increase in coming years.

D. Increasingly, software test engineers

- ***get paid well***
- ***boss the developers around***

Motivation, cont'd

E. My all-time favorite testing-related failures:

   1. *NASA deep space network 2-day crash*

   2. *massive northeast power blackout*

F. The same cause for both -- ***what was it?***

IX. Review of testing terminology.

-- sitck "test" or "testing" in front of or after each:

1. unit
2. module
3. integration
4. system
5. acceptance
6. black box
7. white box
8. design
9. plan
10. top-down
11. bottom-up
12. case
13. oracle
14. stub
15. driver
16. regression
17. coverage
18. subsumption
19. automation
20. mutation
21. harness
22. framework
23. suite

X. **Unit Testing**

A. Done at the level of function, aka method.

B. Provide inputs, expected outputs.

C. Check the actual outputs meet expected.

XI. **Module Testing**

A. Done at level of class, aka, module.

B. Define test fixtures.

C. Define unit-by-unit test execution.

D. Consider inter-function communication.

XII. **Integration Testing**

A. Done at level of package, aka, namespace.

B. Integrate multiple module tests.

C. Defined external data source test fixtures.

XIII. System Testing

A. Done at level of sub-systems, aka separate launch points

B. Super-integrate previously tested packages

XIV. **Acceptance**

A. Done at level of HCI/API.

B. Provide inputs at external interface, not at code level.

XV. **Black Box Testing**

A. Tests based on external specification.

B. Code is not used to generate tests.

XVI. **White box**

A. Tests based on internal implementation.

B. Code paths used to generate tests.

XVII. **Testing Design**

A. Organize all of the different levels of testing.

B. Define critical paths.

XVIII. **Test Plan**

A. The framework-independent documentation of a testing level.

B. Function comment for unit test plan.

C. Class comment for module test plan.

D. Package comment for system test plan.

XIX. **Top-down Testing**

A. Top-level components tested first.

B. "Stubs" written for lower-level methods.

XX. **Bottom-up Testing**

A. Lower-level components tested first.

B. Function "drivers" written for upper-level methods.

XXI. **Test Case**

A. One input/output pair in a test plan.

XXII. **Testing Oracle**

A. The entity that determines the expected out-put.

B. The entity that validates the actual and expected output are equal.

XXIII. **Testing Stub**

A. A place holder for an unimplemented soft-ware component.

B. Provides "canned" data for other components being tested

XXIV. **Test Driver**

A. Executes components being tested with upper-level components are not yet implemented.

XXV. **Regression Testing**

A. Record results of step phase *n*.

B. Compare same-unit results with test phase *n-1*, expecting no differences.

XXVI. **Test Coverage**

A. Ensure that test cover all white box execution paths.

XXVII. **Test Subsumption**

A. When the results of one test of test case fully cover another case or test.

B. Allows redundant tests to be removed from a suite.

XXVIII. **Test Automation**

A. Computational support for any and all aspects of testing.

B. Most typically automated are results recording, regression differencing, and coverage

XXIX. **Mutation Testing**

A. Systematic changes to code being tested and re-execution of tests.

B. Goal is to uncover test weaknesses.

XXX. Testing Harness

A. System-level test driver

XXXI. **Testing Framework**

A. Organizational structure of the tests and there execution.

B. Different frameworks support different testing styles.