Testing Software Quality Characteristics – Part 1

Configuration Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Utilize strategies for configuration testing

Multiple Configurations

- Should performance tests be repeated for each possible configuration?

Configuration Testing

Verify that the functional and performance requirements of the system are met for the different configurations that the system must run on.

Configuration Testing Steps

1. **Identify the parameters that define each configuration that could have an impact on the system's ability to meet its functional and performance requirements**
   - CPU
   - Operating system
   - Memory
   - Data Base
2. **Partition (group similar parameters to reduce possible number of configurations)**
3. **Identify configuration combination to test**
   - Boundaries (maximum and minimum
   - Risk based
   - Design of experiments pairwise combinations (used to select combinations of configuration parameters when testing all combinations is impractical or not needed)

Example: Performance Test of Car

Engine	Transmission	2D/4D	Tires
3.0	auto (a)	2D	15 inch normal (15 n)
3.8	manual (m)	4D	15 inch high performance (15 hp)
5.0			

There are 24 configuration combinations.

Performance Test of Car (continued)

Slowest Configuration

3.0 / auto / 4D / 15 n

Fastest Configuration

5.0 / manual / 2D / 15 hp

Risk-based Selection Based on Projected Sales

3.8 / auto / 4D / 15 n

Pairwise Combinations

Engine	Transmission	2D/4D	Tires
3.0	a	2D	15 n
3.0	m	4D	15 hp
3.8	a	2D	15 hp
3.8	m	4D	15 n
5.0	a	4D	15 hp
5.0	m	2D	15 n

Summary

Testing Software Quality Characteristics – Part 1

Performance Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Utilize strategies for performance testing

Performance Testing

Objective

Verify that the system meets its performance requirements for specified load conditions including stress and volume scenarios.

Entry Criteria for Performance Testing

- **Quantitative and measurable performance requirements**
- **Reasonably stable system**
- **Test environment representative of customer site**
- **Tools**
  - Load generator
  - Resource monitor

Test Data Management

- Growing complexity and volume of data in today's applications requires emphasis on test data management
- Full data sets replicating real data from operational system is ideal

Exercise

- **Assume you are a system tester for a new airline reservation system.**
- **Discuss how to improve the following performance requirement:**
  - "The airline reservation system shall have excellent response time."

Load Specification

For relevant use-cases it is important to specify performance requirements in terms of load

Load may reflect:

- Different volumes of activity
  - Busy hour in a cell phone system
  - Same call profile used
- Different mixes of activity
  - Earthquake or tornado in a cell phone area
  - Different call profile

Varying Load During Performance Testing

- **For relevant use-cases the load should be varied and response time tracked**
  - Verify performance requirments
- **Resource usage can also be tracked as the load is varied to identify potential bottlenecks or sources of performance problems**

Summary

Testing Software Quality Characteristics – Part 1

Regression Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Utilize various strategies for regression testing

Regression Testing

- **Modifying existing software is a high risk-activity**
- **Modifications occur due to:**
  - Error fixes
  - Incorporation of new functions
- **Modifications may introduce errors due to:**
  - Code ripple effects
  - Unintended feature interactions
  - Changes in performance, synchronization, resource sharing, etc.

Examples

GTE / Arizona State University study found that almost half of problems detected during system test of a large switching system were in features that worked fine in the previous release

Caper Jones study found bad fix injection rates vary from less than 2% to more than 20%

Ariane 5 rocket controller

- $350 million dollar loss
- Didn't regression test code
- Assumed dynamics of rocket were the same as Ariane 4

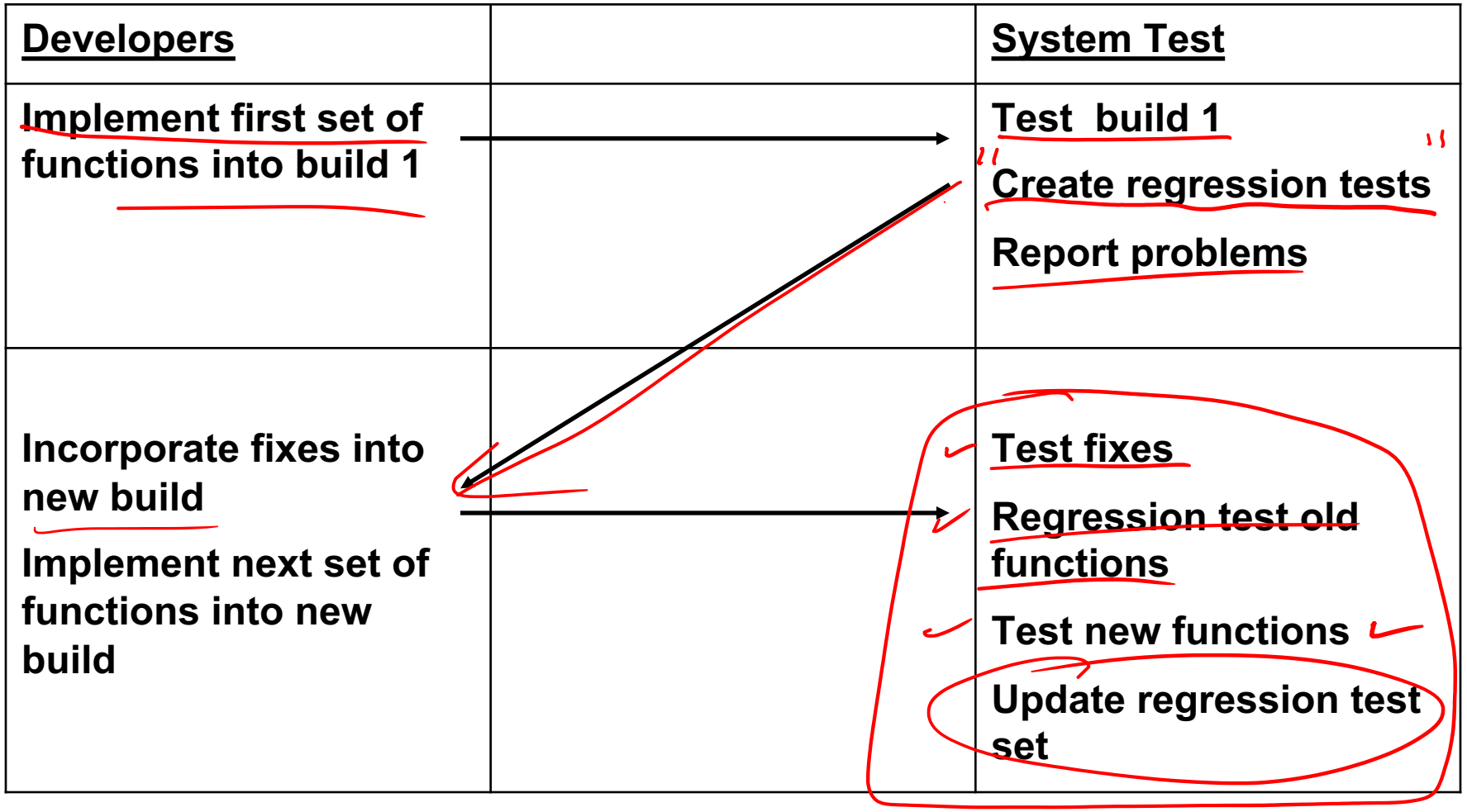
Objective

Objective of regression testing is to ensure that previously developed and tested functions continue to work as specified after software modifications have been made

Regression testing must be approached from a multi-level point of view

- Unit level regression tests
- Integration level regression tests
- System level regression tests

Incremental Development and Testing Process Overview



Detailed Defect Repair Process

System Test

Execute test and detect error

Write problem report.

Problem Analyst

Analyze problem and recommend corrective action

Development

repair code.

Inspect change.

Unit test change.

Change Control Board

Decide to repair, cancel or defer

Regression test components modified.

Add changes to next build.

Test fix.

Regression test.

Regression Test Strategies

Full

- Rerun all existing tests in response to a code modification
- Normally impractical

Selective

- Rerun a selected subset of tests based on the modification
- Execute a standard confidence test irregardless of the modification

Selective Regression Testing Based on Modification

- **Typically requires tools and close communication with developer**
- **Strategies include:**
  - Testing of code deltas
  - Ripple effect analysis

Testing of Code Deltas

- **Requires coverage tool for mapping test cases to code at desired granularity level:**
  - Code block
  - Component
- **Requires configuration management tool to identify code change deltas**
- **Strategy suggests re-running tests that traversed changed or deleted code**

Example

Tests

Components

	T1	T2	T3	T4	T5
A	X	X	X	X	X
B	X		X		
C		X			X
D	X			X	
E			X		
F		X			X

If components B and D are changed, tests T1, T3 and T4 are re-executed.

Ripple Effect Analysis

- Requires developers to identify the impact of changes on other requirements or features
- Best addressed via checklist items in a modification inspection
- Potentially impacted requirements and features are communicated to system test

Example

Tests

Components

	T1	T2	T3	T4	T5
F1	X				
F2		X			X
F3			X		X
F4				X	
F5	X				X
F6		X			
F7				X	

If a modification to F1 is determined to impact F3, then tests T1, T3 and T5 are re-executed.

Selective Regression Testing Using a Confidence Test Suite

- Select a subset of tests to execute to verify previous functionality
- Include tests addressing:
  - High frequency use-cases
  - Critical functionality
  - Functional breadth
- Inspection of test cases should include a checklist item addressing the suitability of particular tests for the confidence test suite

Revalidation Issue

- Regression tests must be revalidated to ensure they are consistent with the software modification
- Test inputs and expected outputs must be re-examined for correctness

Summary

Testing Software Quality Characteristics – Part 1

Stress Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Utilize strategies
for stress testing

Stress Testing

- Verify the behavior of the system meets its requirements when its resources are saturated and pushed beyond their limits.
- Attempt to find the stress points and ensure the system performs as specified

Analogies

- Human stress test
- Automobile stress test
- Bridge stress test

Stress Testing Steps

1. **Identify stress points**
   - Work with developers and architects
   - Identify potential bottlenecks
2. **Develop a strategy to stress the system at points identified in Step 1**
   - Often requires load generation tools
3. **Verify that intended stress is actually generated**
   - Stress testing strategy may be ineffective
   - System performance may be better than expected
4. **Observe behavior**
   - Verify stress related requirements are met
   - Ensure functional correctness

Summary

Testing Software Quality Characteristics – Part 1

Volume Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Utilize strategies
for volume testing

Volume Testing

Verify the behavior of the system meets its requirements when the system is subjected to a large volume of activity over an extended period of time.

Errors Targeted by Volume Testing

- Memory leaks
- Counter overflow
- Resource depletion

Summary