Testing Software Quality Characteristics – Part 2

Error Detection, Recovery and Serviceability Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Develop error detection, recovery and serviceability tests

Error Detection and Recovery Testing

- Overall system reliability and availability is dependent upon the system's ability to detect and recover from a variety of failures
  - User
  - Hardware
  - Software
  - Other systems

Error Detection and Recovery Testing (cont'd)

- It is essential to have a list of the errors to recover from specified in the requirements

- Usual testing approach consists of error injection

Serviceability Testing

- Important for system availability
- Objective is to verify serviceability requirements are being met
  - e.g. "critical problems will receive fixes or workarounds within 4 hours"
- Serviceability includes all aspects of problem reporting, isolation, correction, verification and fix release
- Usual testing approach is to inject a failure and assess response

Summary

Testing Software Quality Characteristics – Part 2

Reliability Testing

Objective

Objective

Apply operational profile testing to assess software reliability

Reliability Definition (from John Musa)

- “The probability that a system or a capability functions without failure for a specified time or number of natural units in a specified environment"

- Natural units correspond to the processing performed such as the number of calls or transactions completed (e.g. one transaction lost per 50,000)

- Probabilities have a range of 0 to 1

Availability Definition from John Musa

“The probability at any given time that a system or capability of a system functions satisfactorily in a specified environment"

Software Availability Calculation

- Availability can also be performed as the percent of time the system performs satisfactorily

$$\text{availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}} \times 100\%$$

5NINES Availability Requirement

- The system shall be available 99.999% of the time, i.e. the probability the system functions satisfactorily at any point in time is 99.999

- In terms of service to a customer over a year, this translates into approximately 5 minutes of allowable down time per year
  - There are 525,600 minutes per year
  - 5NINES implies one minute of down time per 100,000 minutes

Achieving High Reliability and Availability

- Achieving high reliability and availability in a cost effective manner requires the prudent application of Software Reliability Engineering techniques

- Appropriate development techniques must be applied for:
  - Fault prevention
  - Fault tolerance

Achieving High Reliability and Availability (cont'd)

- **Appropriate testing techniques must be applied along with models for assessing whether reliability and availability objectives are being met.**
  - Operational profile testing
  - Error detection and recovery testing
  - Serviceability testing

Introduction to Operational Profiles

- An operational profile describes how users utilize a product
- An operational profile consists of a set of major functions performed by the system and their occurrence probabilities
- An operational profile is essential for reliability prediction

Basic Operational Profile Construction Steps

1. **Identify the major functions performed by the system**
   - Identify different types of users / external entities
   - Use-cases are good candidates for basing the operational profile
2. **Identify the occurrence rates**
   - Historical data
   - Marketing
3. **Calculate the occurrence probability**

ATM Example

Use-Cases	Occurrence Rate (xact / hr)	Occurrence Probability
Deposit	95	0.095
Withdraw	900	0.9
Transfer	5	0.005

Development of Tests

- Tests are developed based on the operational profile
- Test generation is modified to incorporate critical functions with low occurrence probabilities
- Number of tests to execute is based on the reliability objectives

Interpreting Failure Data

Development Testing

- Goal: To remove faults that have causes failures

Certification Testing

- Goal: To determine whether a software component or system should be accepted or rejected

Other Uses of Operational Profiles

- **Operational profiles may also be used to:**
  - Guide development priorities
  - Assist in performance analysis

Summary

Testing Software Quality Characteristics – Part 2

Reliability Models

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Identify how software reliability models work

Modeling Software Reliability Growth

- Reliability growth model shows how reliability changes over time
- Models support answering "when to stop testing?" question
- Numerous models exist

- Effectiveness of any reliability measurement is directly related to the effectiveness of collecting the right data during testing such as:
  - Failure intensity: the number of failures per natural or time unit

Time to Failure

Cumulative Number of Failures

Number of Failures Per Unit of Time

Software Reliability Models

- All models possess assumptions such as:
  - No new errors are introduced by fixes
- Models generally utilize a mathematical distribution to represent reliability growth
  - Poisson
  - Exponential
- Effective model predictions require testing with an operational profile

Statistical Testing

- Testing software for reliability rather than fault detection

- An acceptable level of reliability should be specified and the software tested and amended
until that level of reliability is reached

Reliability Testing Problems

- **Operational profile uncertainty**
  - Is the operational profile an accurate reflection of the real use of the system
- **High costs of test data generation**
  - Very expensive to generate and check the large number of test cases that are required
- **Statistical uncertainty for high-reliability systems**
  - It may be impossible to generate enough failures to draw statistically valid conclusions

Growth Model Selection

- Many different reliability growth models have been proposed
- No universally applicable growth model
- Reliability should be measured and observed data should be fitted to several models
- Best-fit model should be used for reliability prediction

Summary

Testing Software Quality Characteristics – Part 2

Security Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Identify basic security testing approaches

Security Testing

- Software correctness and security are not the same
- Most applications contain private data

- Goal of security testing is to ensure private data is protected from unauthorized users

Security Fundamentals

Confidentiality

- Application
- Data

Integrity

- Data modification
- Functions performed

Availability

- Denial of service

Security Testing Context

- Software may have unintended or unknown functionality that may produce side-effects contributing to security problems
- Security flaws require testing software interactions with its environment

Components that Might Exploit Software

- OS
- File System
- GUI
- Other systems (databases, libraries, etc.)

GUI Security Risks

- **Verify access control**
  - Entry to system
  - Access to functions and data
- **Look for all possible access methods to data**
  - Cut and paste
  - Screen capture
- **Evaluate malicious input**
  - Denial of service

File System Security Risks

- Evaluate how data is stored and retrieved
- Focus on encryption and data protection

OS Security Risks

- Evaluate decrypted data storage in memory

- Stress test with low memory
  - System under memory stress may leave data unprotected

Other Component Security Risks

- Consider results of component failure
- Components may consist of libraries, databases, etc.

Security Testing Strategies

- Deny application access to libraries it needs
  - Ensure crashes do not impact security
- Try to overflow input buffers by inputting long strings
- Try special characters as inputs
- Try default or common user names and passwords

Security Testing Strategies (cont'd)

- **Attempt to fake the source of data**
  - Consider a system with packets sent over the network which contain source identifier
  - Fake source in packet

- **Force system to use default values**
  - Do not enter data when prompted
  - Exploit time outs

Security Testing Strategies (cont'd)

- Test all routes to perform a task
  - Consider opening a file
  - Ensure all scenarios go through security validation
- Produce each error message and ensure that it does not compromise security

Approaches for Improving Security Testing

- Consult public security databases
  - CERT (www.cert.org)
  - Contain information about published software bugs

- Reason about errors in databases and possible vulnerabilities in your product
  - What caused the failure
  - How might it have been detected during test
  - Is system vulnerable to attack

Summary

Testing Software Quality Characteristics – Part 1

Usability Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Generate usability tests

Usability 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

Usability Testing

- Close to one-half of code in many applications is in the user interface

- Usability is the degree to which intended users are:
  - Able to perform tasks the product is intended to support in intended environment
  - Satisfied by the procedures they must follow and the resultant output
  - Protected from consequences of their actions

Usability Requirements

- **Usability requirements are typically stated in terms of:**
  - **Learnability**: the type and amount of training required to bring users to a desired level of performance
  - **Memorability**: addresses the ability to retain skills in using a product once it is learned
  - **Errors**: measures the number of incorrect actions a user makes in trying to accomplish a task
  - **Efficiency**: measures the speed with which tasks can be performed
  - **Subjective satisfaction**: the user's overall feeling about the product

Usability Testing Reliability and Validity Concerns

- **Reliability: would you get the same results if test were repeated**
  - Best user is 10X faster than slower
  - Best 25% are 2X faster than

- **Validity: does usability test measure something of relevance**
  - Wrong users
  - Wrong task

Test Goals

Formative Evaluation

- Learn which aspects of interface are good and bad
- How can design be improved

Summative evaluation

- Assess the overall quality of the interface
- Measurement test

Test Plan Concerns

- Who are the users?
- What task will they perform?
- What user aids will be available?
- What data is to be collected?
- What criteria will be used to determine success?

Pilot Tests

- Test procedures must be tried out in a pilot study

- Evaluate
  - Instructions
  - Success criteria
  - Time to perform tasks
  - Evaluation criteria

Identifying Test Users

- Users must be representative
- Evaluate with both novice and expert users
- Be prepared to train users to achieve expert level

Usability Comparison

- When evaluating usability choices care must be taken when using the following testing strategies:
  - Between subject testing
  - Within subject testing

- Within subject testing is preferable

Ethical Aspects with Human Subjects

- Subjects may have concerns about performing inadequately
- Need to make subjects feel comfortable
- Emphasize system is being tested and not the user
- Maintain privacy issues

Test Tasks

- Must be representative
- Begin with easy tasks to boost confidence
- Give tasks one at a time

Stages of Test

- Preparation (ensure environment is set-up)
- Introduction (welcome, purpose, overview)
- Running the test
- Debriefing

Thinking Aloud

- Test subject uses system which continuously thinking out loud
- Testers may need to periodically prompt test subject

Usability Lab

- Two-way mirror

- Video cameras
  - User faces
  - How user is interacting with system, doc, etc.

Summary