Software Testing

Part 2 of 4
Program Testing

- Can reveal the presence of errors NOT their absence
- A successful test is a test which discovers one or more errors
- The only validation technique for non-functional requirements
- Should be used in conjunction with static verification to provide full V&V coverage
Execution Based Testing

“Program testing can be a very effective way to show the presents of bugs but is hopelessly inadequate for showing their absence”

[Dijkstra]
Behavioral Properties

- **Correctness** - does it satisfy its output specification?
- **Utility** - are the user’s needs met
- **Reliability** - frequency of the product failure.
  - How long to repair it?
  - How long to repair results of failure?
- **Robustness** - How crash proof in an alien environment?
  - Does it inform the user what is wrong?
- **Performance** - response time, memory usage, run time, etc.
Testing and Debugging

- Defect testing and debugging are distinct processes
- Verification and validation is concerned with establishing the existence of defects in a program
- Debugging is concerned with locating and repairing these errors
- Debugging involves formulating a hypothesis about program behavior then testing these hypotheses to find the system error
The Debugging Process

1. Test results
2. Specification
3. Locate error
4. Design error repair
5. Repair error
6. Re-test program
7. Test cases
Testing Phases

Component testing

Integration testing

Software developer

Independent testing team
Testing Phases

• Component testing
  – Testing of individual program components
  – Usually the responsibility of the component developer (except sometimes for critical systems)
  – Tests are derived from the developer’s experience

• Integration testing
  – Testing of groups of components integrated to create a system or sub-system
  – The responsibility of an independent testing team
  – Tests are based on a system specification
Testing Priorities

• Only exhaustive testing can show a program is free from defects. However, exhaustive testing is impossible
• Tests should exercise a system's capabilities rather than its components
• Testing old capabilities is more important than testing new capabilities
• Testing typical situations is more important than boundary value cases
Test Data and Test Cases

- *Test data* Inputs which have been devised to test the system

- *Test cases* Inputs to test the system and the predicted outputs from these inputs if the system operates according to its specification
Development of test cases

• Test cases and test scenarios comprise much of a software systems *testware*.

• Black box test cases are developed by domain analysis and examination of the system requirements and specification.

• Glass box test cases are developed by examining the behavior of the source code.
The Defect Testing Process
Methods of Testing

• Test to specification:
  – Black box,
  – Data driven
  – Functional testing
  – Code is ignored: only use specification document to develop test cases

• Test to code:
  – Glass box/White box
  – Logic driven testing
  – Ignore specification and only examine the code.
Guaranteeing a Program Correct?

- This is called the Halting Problem (in general)

- Write a program to test if any given program is correct. The output is correct or incorrect.
- Test this program on itself.
- If output is incorrect, then how do you know the output is correct?