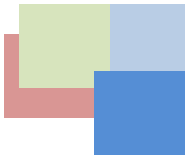
A Brief Essay on Software Testing

Antonia Bertolino and Eda Marchetti

200310404
유 광 현

Abstract & Introduction

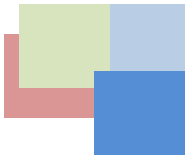
- Testing is not limited to the detection of “bugs” in software
- Recent trends in S/E evidence the importance of …
- Have to start at the requirements specification stage
- Testing is a challenging activity that involves several highly demanding tasks



On the Nature of the Testing Discipline

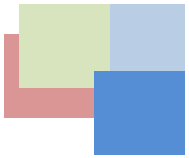
- Static analysis techniques
  - Do not involve the execution of the tested system
  - Check the adherence of the implementation to the to the specifications
  - to Detect flaws in the code

- Dynamic analysis technique
  - Exercise the software



A General Definition

Software testing consists of the dynamic verification of the behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, againts the specified expected



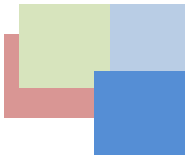
Fault Versus Failure

- Failure
  - the manifested inability of the program to perform the function required

- Fault
  - A missing or incorrect piece of code
    → cause of a failure!!

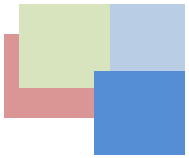
- Error

Fault → Error → Failure



The Notion of Software Reliability

- Some faults will inevitably escape testing and debugging
  → It will eventually show up to the final user
- Important in deciding whether a software product is ready for release
- Software Reliability is a probabilistic estimate
- Measures the probability that the software will execute without failure



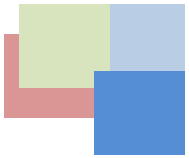
Types of Tests

- Static Techniques

  Based solely on the examination
  - of project documentation
  - of software models and code
  - other related information about requirements and design

☞ Heavily manual, error-prone, and time-consuming

- Software inspection
- Software reviews
- Code reading
- Algorithm analysis and tracing

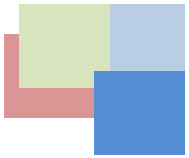


Types of Tests (cont'd)

- Dynamic Techniques
  - Obtain information of interest about a program by observing some executions
  - Based on the execution of the code on input value

☞ Must be adopted to find a trade-off

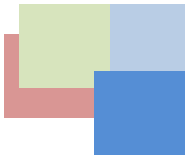
→ between the number of chosen inputs and the overall time and effort dedicated to testing purposes



Test Levels

- Unit test
  - The smallest testable piece of software
  - To ensure that the unit satisfies its functional specification

- Integration test
  - The Process by which software pieces or components are aggregated to create a larger component
  - Aimed at verifying that each component interacts



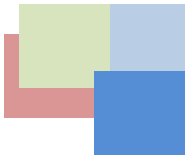
Test Levels (cont'd)

☐ System test

- Embedded in its actual hardware environment
- Verifying that the system behaves according to the user requirements
- Includes testing for performance, security, reliability etc.

The primary goals

- ○ Discovering the failure
- ○ Increasing the confidence
- ○ Collecting information useful for deciding the release of the product



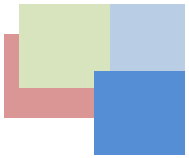
Strategies for test case selection

- Effective testing requires strategies to trade off between
  - amplifying testing thoroughness
  - reducing time and cost

- Provided by a *test criterion*
  - Guiding in a proactive way the selection of test cases

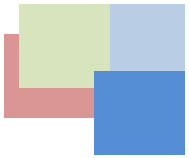
when C(P, RM, T) holds

→ T satisfied criterion C for p and RM



Selection Criteria Based on Code

- Also called “structural test” or “white-box testing”
- Potential failures can only be detected if the parts of code related to the causative faults are executed
- Tries to exercise thoroughly all “program elements”



Selection Criteria Based on Spec.

- Black-box testing
- RM is derived in general from the documentation relative to program specifications

  Equivalence classes

  Boundary conditions

  Cause–effect graphs

Test Design

☐ Must be organized into a coherent framework

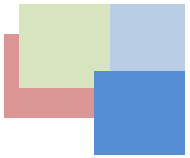
Test planning

- Outline the scope of testing activies

(rather than details)

Test design

- Which the objectives and the feature to be tested
- associated to each of them are defined
- the levels of test are planned



Test Execution

☐ Launching the Tests

- Forcing the execution of the test cases derived according to one of the criteria

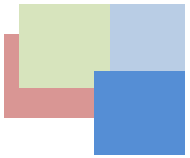
☐ Test Oracles

- A test is meaningful on If it is possible to decide about its outcome

○ Limited number of test cases is executed
→ the oracle can be the tester

But!

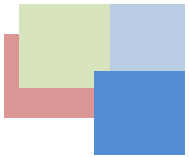
○ When the tests cases are automatically derive, or when their number is quite high
→ Automated oracles must then be implemented



Test Execution(cont'd)

- Test Oracle(cont'd)
  - Output
    - reject
    - approve
    - inconclusive

It should be evident that the oracle might not always judge correctly!!



Test Execution

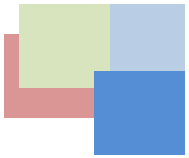
- Test Tools
  - Testing requires fulfilling many labor-intensive task, running numerous executions, and handling a great amount of information

  - *Test harness(drivers, stubs)*
  - *Test generators*
  - *Capture/Replay*
  - *Oracle/file comparators/assertion checking*
  - *Coverage analyzer/Instrumenter*
  - *Tracers*
  - *Reliability*

Test Documentation

- Documentation is an integral part of the formalization of the test process

  - *Test Plan*
  - *Test Design Specification*
  - *Test Procedure Specification*
  - *Test Log*
  - *Test Incident or problem Report*

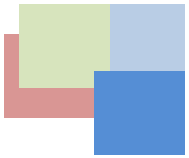


Test Management

☐ Concern different activities

- initiation, scope definition, planning, execution etc.

- Scheduling the timely completion of tasks
- *Estimate of the effort and the resources need to execute the tasks*
- *Quantification of the risk associated with the tasks*
- *Effort/cost estimation*
- *Quality control measures to be employed*

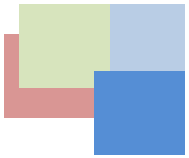


Test Measurements

- Nowadays applied in every scientific field for quantitatively evaluating parameters

- Allows managers and developments to monitor the effects of activities and changes on all aspects of development

  - Evaluation of the Program under Test
  - *Evaluation of the Test Performed*
  - *Measures for Monitoring the Testing Process*



Conclusions

- Presented a comprehensive overview of software testing
- In the past few years, software testing has evolved
  - from an “art” to and engineering discipline
- What we can and must pursue is
  - to transform testing from “trial-and-error” to a systematic, cost-effective, and predictable engineering discipline