SOFTWARE TESTING

DR. VIVEK NALLUR

VIVEK.NALLUR@SCSS.TCD.IE

OUTLINE OF THIS TALK

- Software Testing

SOFTWARE TESTING

Testing is the process of exercising a program with the specific intent of finding errors prior to delivery to the end-user

WHAT CAN TESTING SHOW?

- Errors
- Conformance with requirements
- Performance
- An indication of quality

V & V

Verification refers to the set of tasks that ensure that a software correctly implements a specific function

Validation refers to a different set of tasks that ensure that a software that has been built is traceable to the customer's requirements

WHO TESTS SOFTWARE?

Developer	**Independent Tester**
Understands the system	Must learn about the system
Tests 'gently'	Tests 'forcefully'
Driven by delivery	Driven by quality

TESTING VOCABULARY

Term	Meaning
Mistake	A human action that produces an incorrect result
Fault (or defect)	An incorrect step, process or data in a program
Failure	Inability of a system/component to perform its required function
Error	Difference between a computed/observed/measured value or condition and true/specified/correct value or condition

OPACITY OF TESTING

Depending on how much visibility into the code is available to the tester:

- *Whitebox Testing*: The tester can see all the code, and tries to test all possible paths through program logic
- *Blackbox Testing*: The tester only knows what the expected outputs are, to a particular set of inputs

SCALE OF TESTING

We begin by *testing-in-the-small* and move towards *testing-in-the-large*

- Unit Test (Class)
  - Data structures
  - Boundary conditions
  - Independent paths
  - Error-handling paths

Opacity: Whitebox

HOW MANY TESTS PER CLASS?

Coverage Criteria: Measure what percentage of the code has been exercised by the test suite. Main types are:

- Function / Method coverage: Has every function / method in the class been called?
- Statement coverage: Has every statement been executed?
- Branch coverage: Has every branch of every control-construct (`if-else` or `while`) been executed?
- Condition coverage: Has each boolean expression been evaluated to both `true` and `false`?

SAMPLE CODE

```
int foo (int x, int y)
{
    int z = 0;
    if ((x>0) && (y>0))
    {
        z = x;
    }
    return z;
}
```

WHAT'S THE COVERAGE?

- If `foo` is called at least once, function-coverage is satisfied
- If a test calls `foo(1,1)`, then statement-coverage is satisfied
- If a test calls `foo(0,1) and foo(1,0)` then condition-coverage is satisfied
- Is branch-coverage satisfied with the previous test?

BOUNDARY-VALUE ANALYSIS

Testing at the boundaries between partitions of valid and invalid input

Consider a photocopier, which can take valid values from `1` - `99` for number of copies. Number of boundary value tests:

- At `1`
- At `99`
- Below `1`, i.e., `0`
- Above `99`, i.e., `100`

EQUIVALENCE PARTITIONING

All possible input is sometimes infinite. Hence, divide input-space into equivalent classes

Consider a decision-table that converts marks (out of 100) to grades

Marks	Grade
< 40	D
40 - 59	C
60 - 79	B
> 80	A

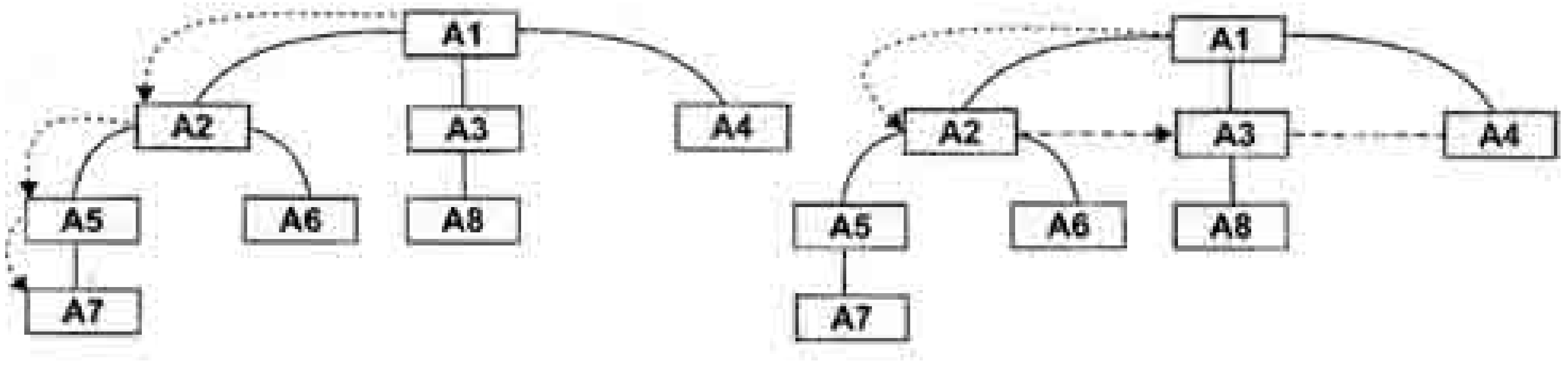
6 partitions for the above table!

TESTING-IN-THE-LARGE

- Integration Testing
- System Testing
- Acceptance Testing

INTEGRATION TESTING

- Opacity: Blackbox and whitebox
- Top-down Integration



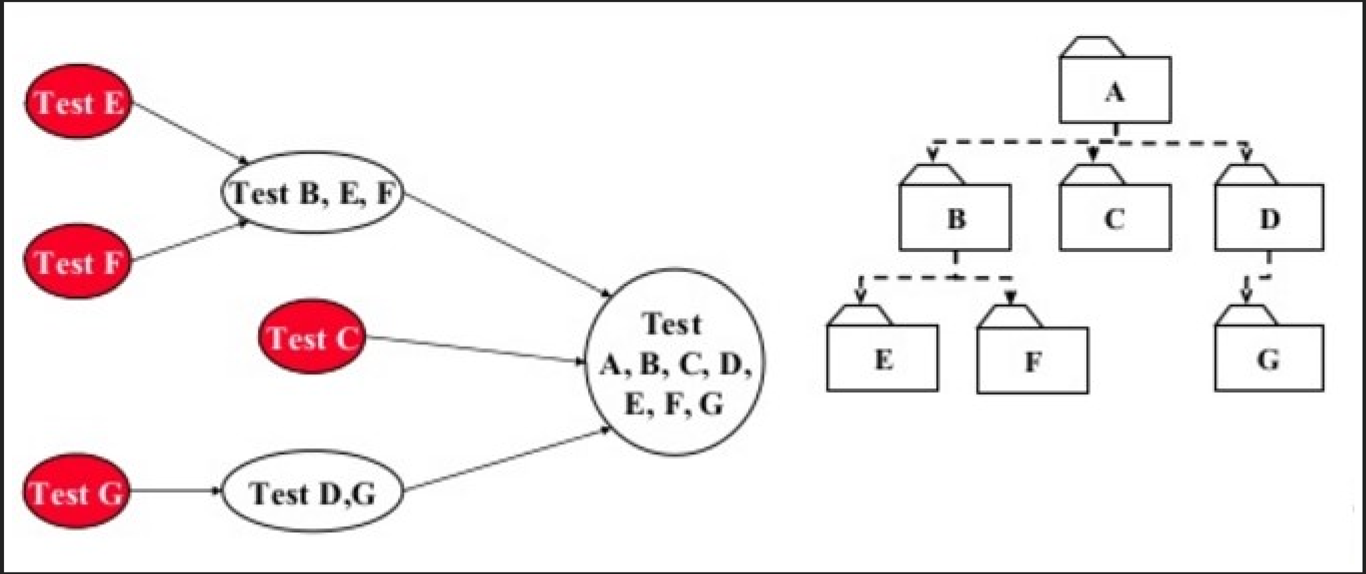
(a) Depth-first Integration

(b) Breadth-first Integration

Top-down Integration

INTEGRATION TESTING - II

- Bottom-up Integration



SYSTEM TESTING

Testing the behaviour of a system/software as per the user requirements.

Test the system along with other *external* peripherals/ software/systems to check if user requirements are met

Opacity: Blackbox

SYSTEM TESTING - II

Kinds of tests to be run include:

- Functional Testing
- Installation Testing
- Usability Testing
- Security Testing
- Scalability Testing
- Performance Testing
- Inter-operability Testing

ACCEPTANCE TEST

Run in the customer's environment, these are carried out by the customer

All aspects of the software can be tested, and evalauted against the software requirements specification

Right to refuse delivery if tests fail!

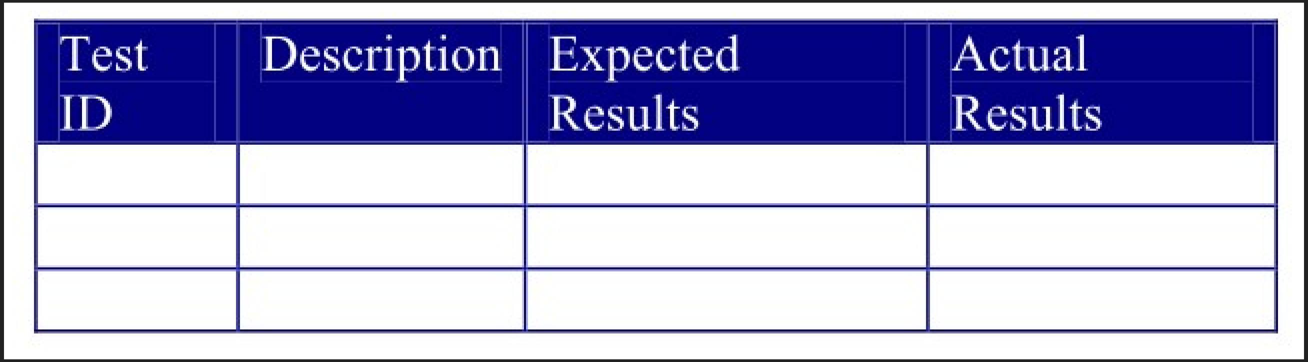
Opacity: Blackbox

OTHER TESTING

- *Regression Testing*: Selective re-testing of a system/component to verify modifications have not caused unintended effects
- *Beta Testing*: Ad-hoc testing by potential users in a production environment.
  Not very systematic, but can reveal unexpected errors

SYSTEMATIC TEST PLANNING

- Test early in the process
- Automate test execution as much as possible (e.g., jUnit, Jenkins, go, etc.)
- Create a list of tests and expected output *before* executing tests



Test ID	Description	Expected Results	Actual Results

LECTURE ANNOUNCEMENT

Lecture on 7th December cancelled! (SFI review)

THAT'S ALL, FOLKS!

Questions? Comments?