Software Systems Verification and Validation

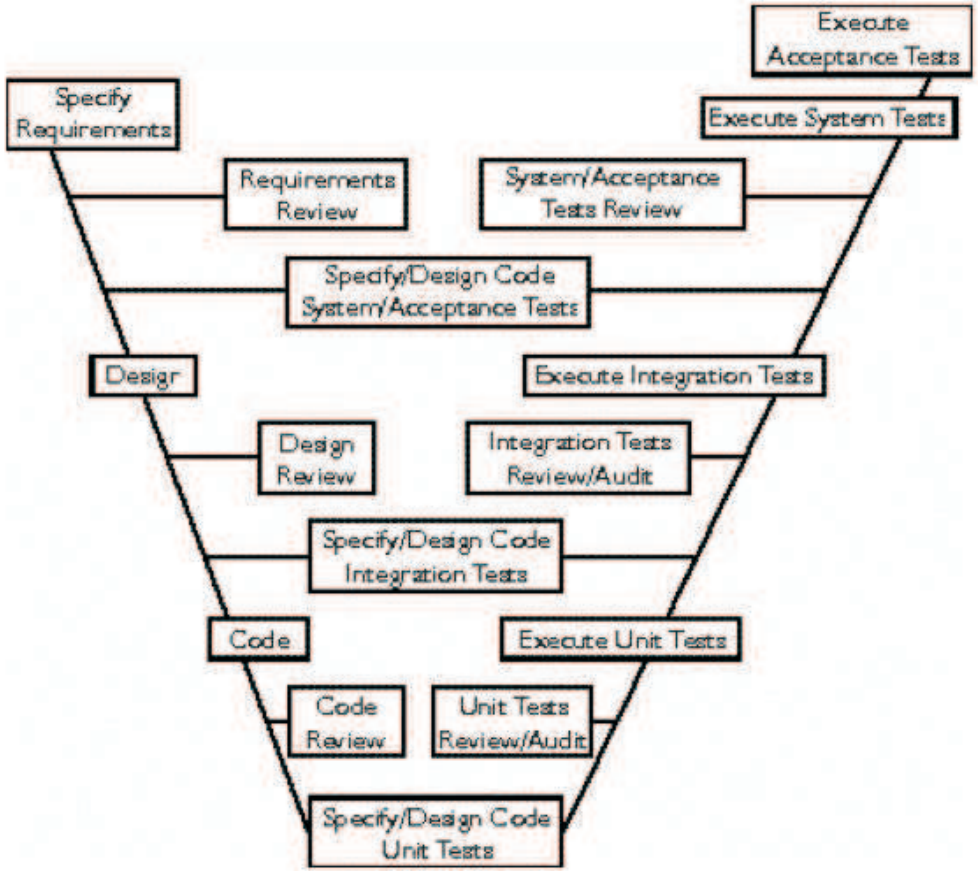
Lecture 02 - Testing

Lect. dr. Andreea Vescan

Babeș-Bolyai University
Cluj-Napoca

2015-2016

Extended V-Model



Definition of Testing

- False definitions
  - Testing is the process of demonstrating that errors are not present. "Testing can only reveal the presence of errors, never their absence." [Dij75]
  - The purpose of testing is to show that a program performs its intended functions correctly.
  - Testing is the process of establishing confidence that a program does what it is supposed to do.
- Definition

  Testing is the process of executing a program with the intent of finding errors. [Mye04]
  - Human beings tend to be highly goal-oriented.
  - Testing is a destructive process.
  - Successful and unsuccessful.
  - Poor performance with infeasible task.
  - Error: the program does not do what it is supposed to do, BUT also if the program does what it is not supposed to do.

Testing principles [Mye04]

1. Definition of the expected output or result;
2. Avoid testing your own program.
3. A programming organization should not test its own programs.
4. Thoroughly inspect the results of each test.
5. Test cases for valid/invalid input conditions.
6. Test if the program does not do what it is supposed to do, AND if the program does what it is not supposed to do.
7. Do not throwaway test cases.
8. Plan testing assuming errors will be found.
9. The probability of the existence of more errors in a section of a program is proportional to the number of errors already found in that section.
10. Testing is an extremely creative and intellectually challenging task.

Testing axioms [Pat05]

1. It is impossible to test a program completely.
2. Software testing is a risk-based exercise.
3. Testing can't show that bugs don't exist.
4. The more bugs you find, the more bugs there are.
5. The pesticide paradox.
6. Not all the bugs you find will be fixed.
7. When a bug's a bug it is difficult to say.
8. Product specification are never final.
9. Software testers aren't the most popular member of a project team.
10. Software testing is a disciplined technical profession.

Testing -fundamental questions

- What do we test? What is our goal?
  $\Rightarrow$ Find bugs!
- How do we organize the process of testing?
  $\Rightarrow$ Testing strategy problem!
- When we have tested enough?
  $\Rightarrow$ Testing measurement problem!
- Exhaustive testing?
  $\Rightarrow$ Input domain subset selection problem!
- Testing strategies?
  $\Rightarrow$ Different testing techniques are appropriate at different points in time!

Program under test

- Program P [Fre10]
- $P : D \rightarrow R$, where:
  - D - set of input data;
  - R - set of output data.

Test case

- Test case $< i, r >$
  - $i \in D,\ r \in R$.
  - for input i the expected result is r.
- Test case: " A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement." [IEE90]
- "Good" test case attributes:
  - High probability of finding an error.
  - Is not redundant.
  - "Best of breed".
  - Neither too simple nor too complex.

Test

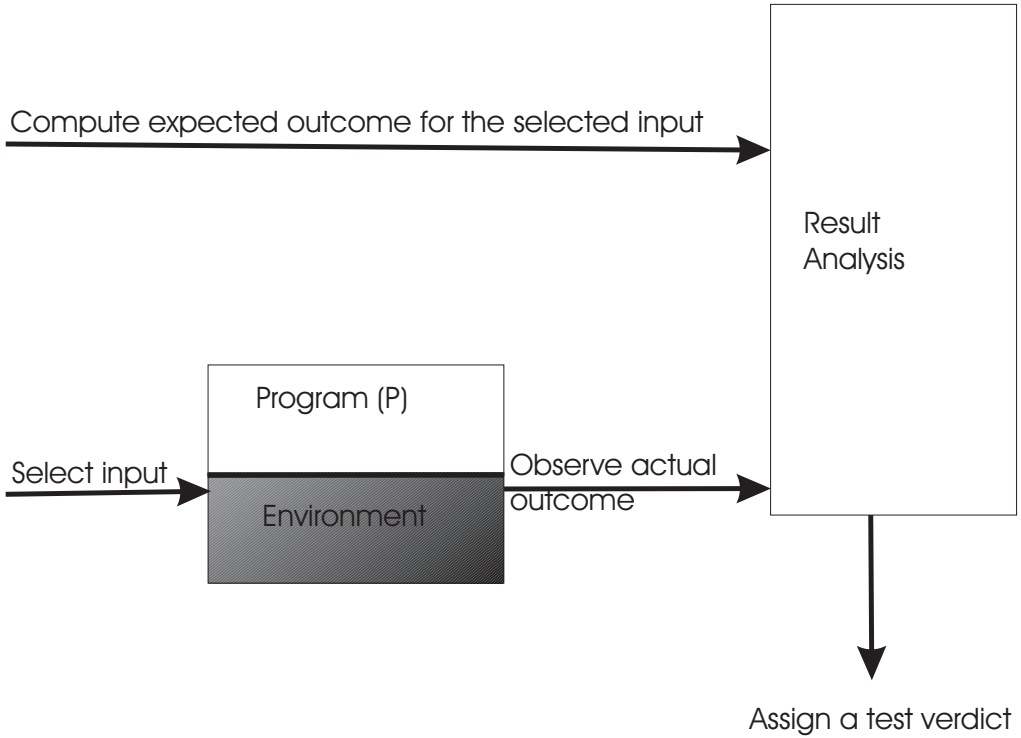
- Test T
  - finite set of test cases $< i, r >$
- Ideal (Successful) test case [Fre10]:
  - $\exists < i, r > \in T$ that highlights defects in the program P.

Types of testing [Fre10]

- Exhaustive testing - all the possible inputs.
  - if D is finite, then P is executed for all possible inputs.
- Selective testing
  - if D is not finite, then we choose inputs $i \in S \subset D$.

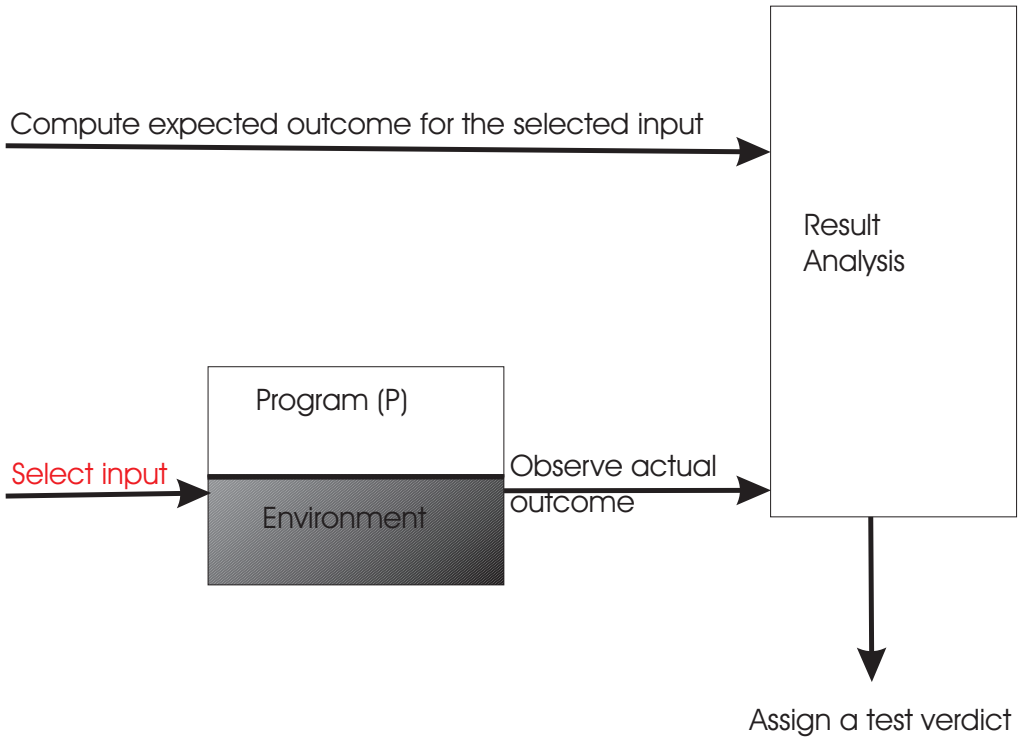
Testing activities

- Identify an objective to be tested.
- Select inputs.



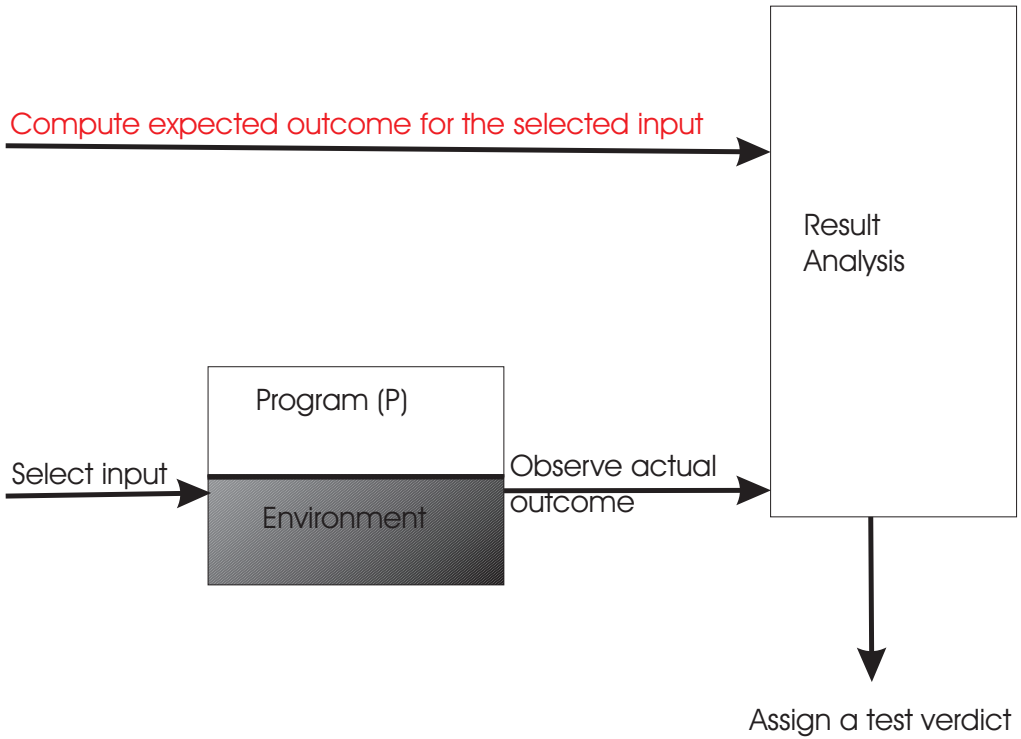
Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.



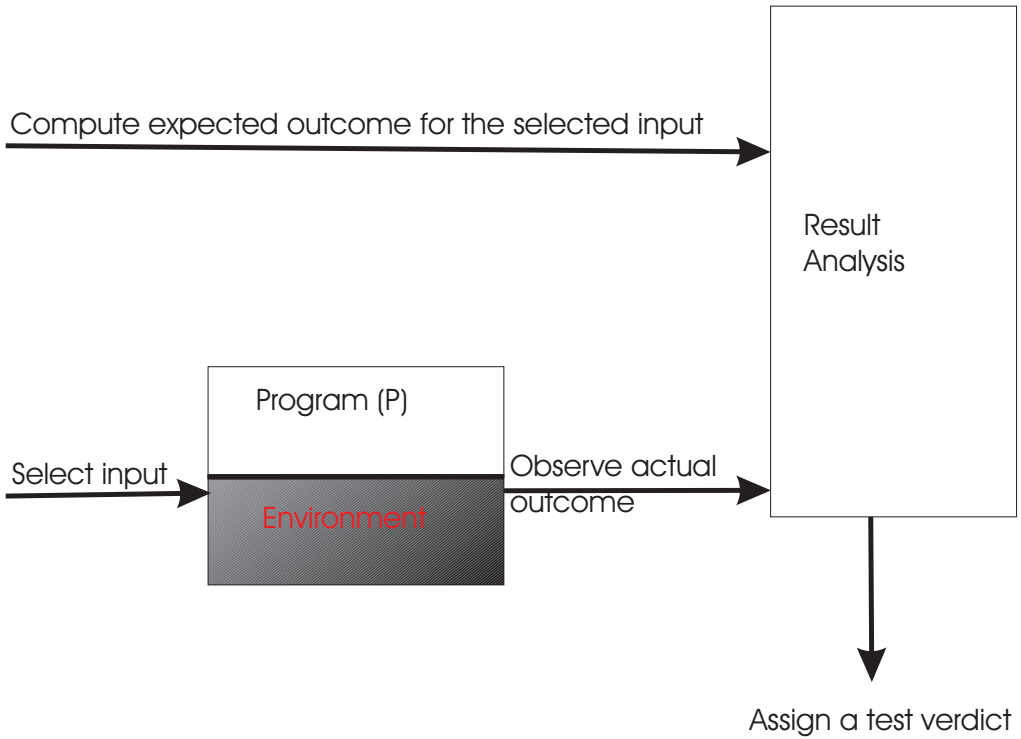
Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.
- Set up the execution environment of the program.



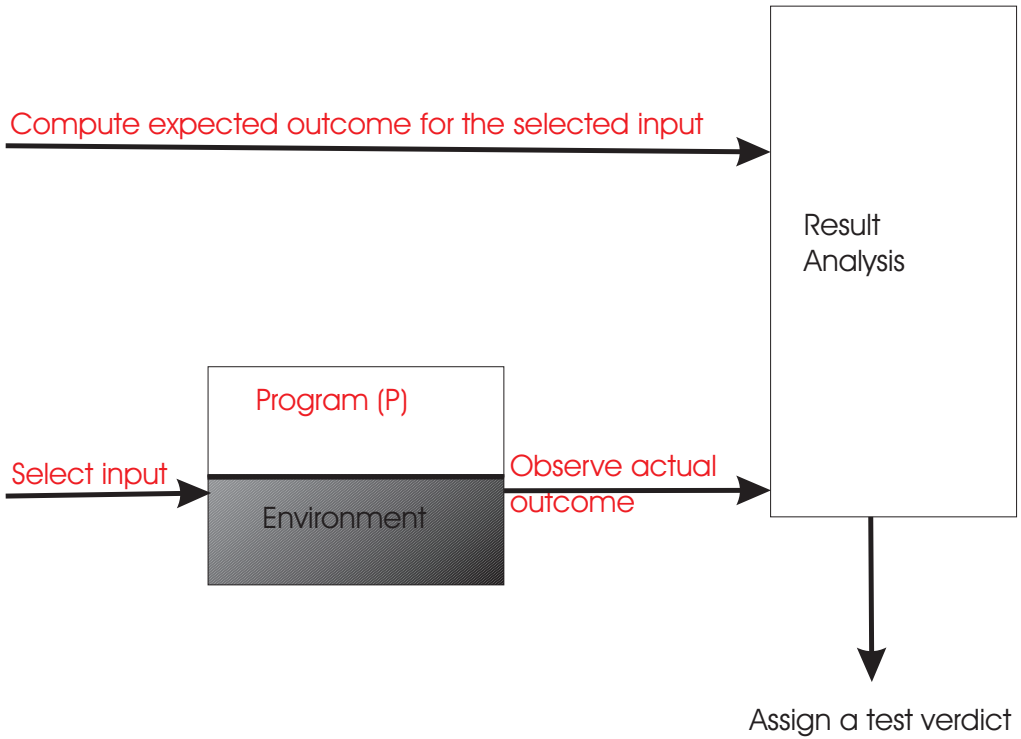
Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.
- Set up the execution environment of the program.
- Execute the program.



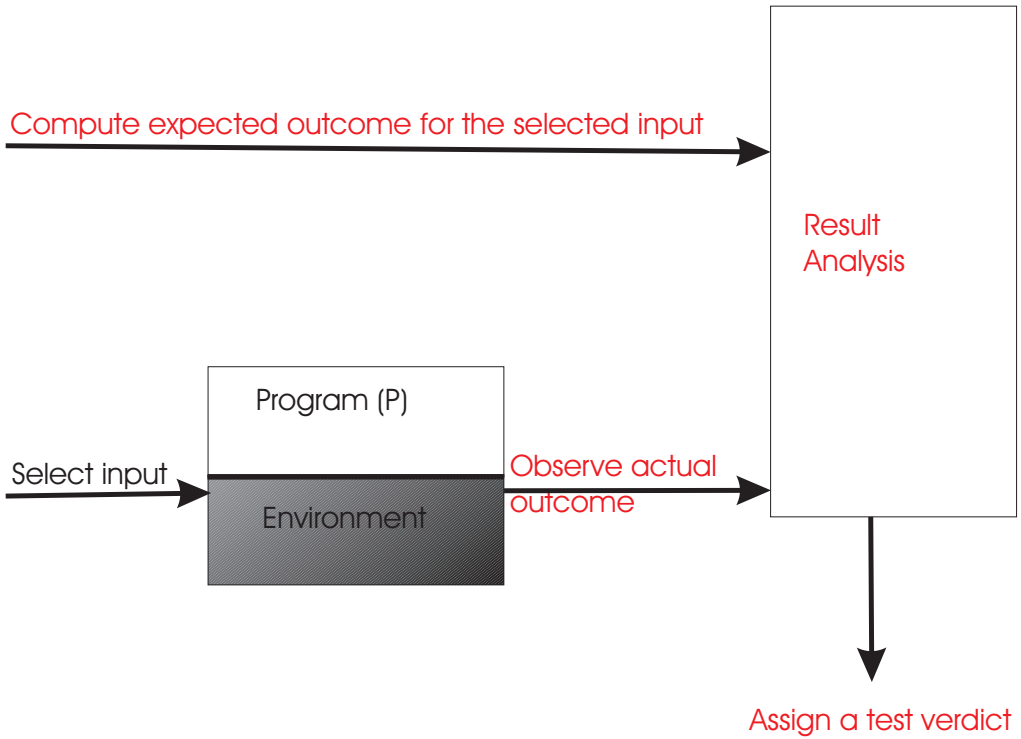
Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.
- Set up the execution environment of the program.
- Execute the program.
- Analyze the test result.



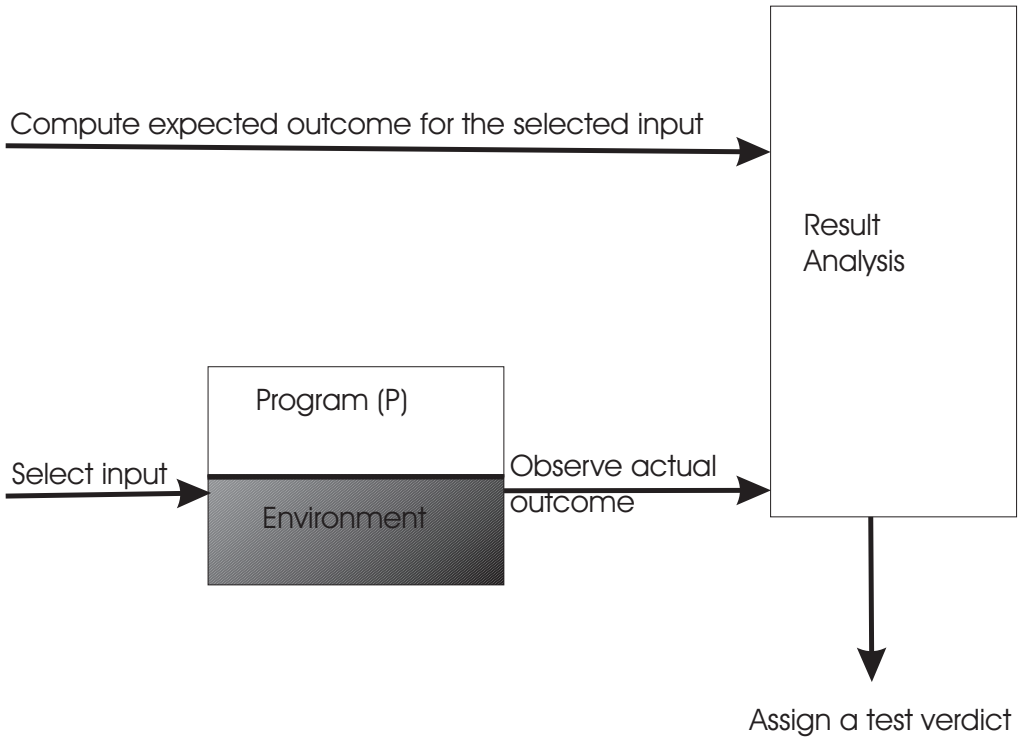
Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.
- Set up the execution environment of the program.
- Execute the program.
- Analyze the test result.



Testing activities

- Identify an objective to be tested.
- Select inputs.
- Compute the expected outcome.
- Set up the execution environment of the program.
- Execute the program.
- Analyze the test result.



Test case design

- Black-box testing $\Rightarrow$ software requirements

Test case design

- Black-box testing $\Rightarrow$ software requirements
- White-box testing $\Rightarrow$ internal program logic

Testing

- Testing is the process of executing a program with the intent of finding errors. [Mye04] [chapter 2]
- Goal of a software tester
  - To find bugs,
  - find them as early as possible,
  - and make sure they get fixed!
- Achieve this goal:
  - properly communicating and documenting the test effort
  - with:
    test plans,
    test cases,
    test reports.

Test planning

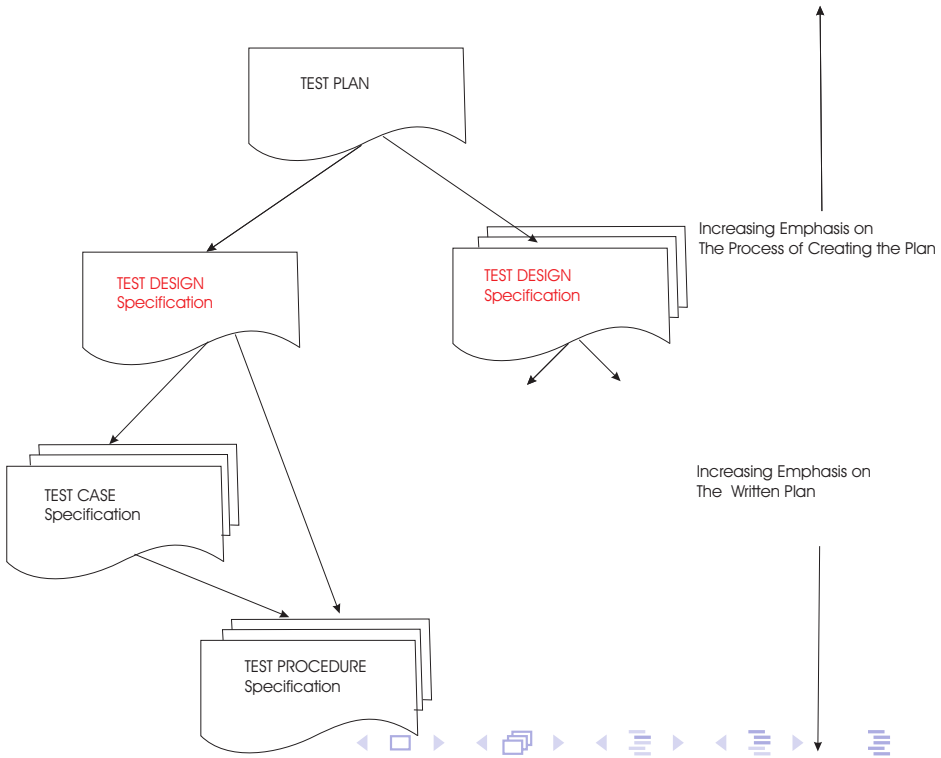
- Test plan?
  - testers communicate what they intend to do.
  - takes the form of a written document, not only the creation of the document, but also planning the testing tasks.
  - The ultimate goal of test planning process is *communicating* the software test team's *intend*, *expectations*, *understanding*.
- Test planning topics
  - A test plan template? Important topics?
    - The test team's high-level expectations
    - People, places and things, Definitions
    - Inter-group responsibilities
    - What will and won't be tested
    - Test phases,test strategy,Resource requirements $\Rightarrow$
    - Tester assignments
    - Test schedule, Test cases
    - Bug reporting

The goal of test planning

- Software creation - process
- product specification $\overset{a\ programmer}{\rightarrow}$ coding ???
- test plan $\overset{a\ tester}{\rightarrow}$ test cases ???
- Reasons for planning test cases
  - Organization
  - Repeatability
  - Tracking
  - Proof of testing

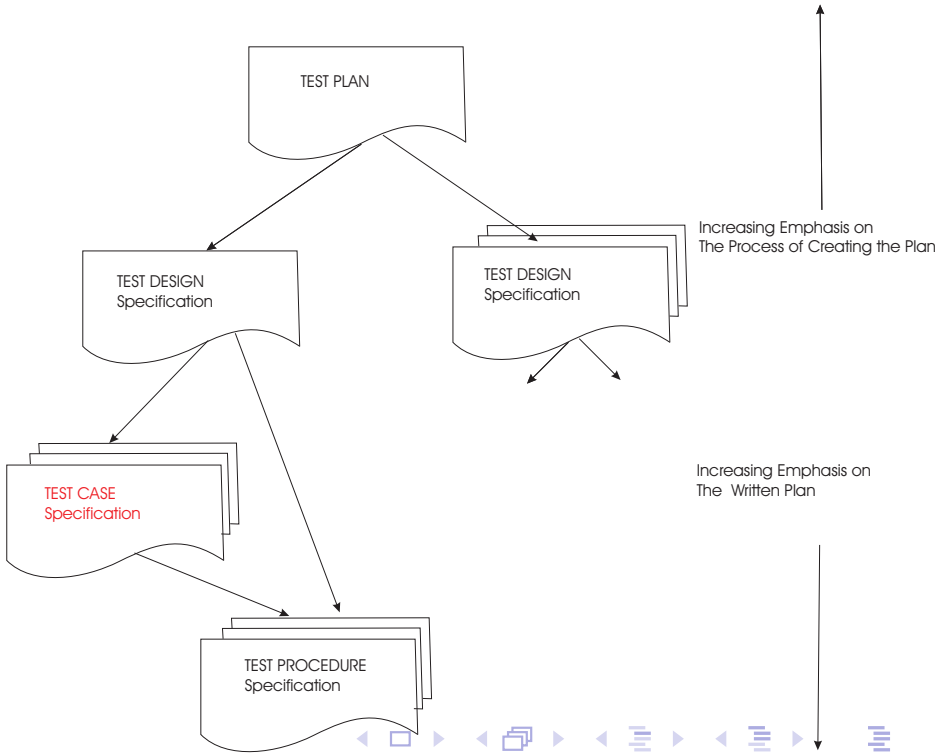
Planning test cases [Pat05]

- Test Design - topics
  - Identifiers, Features to be tested, Approach, Test case identification, Pass/fail criteria.



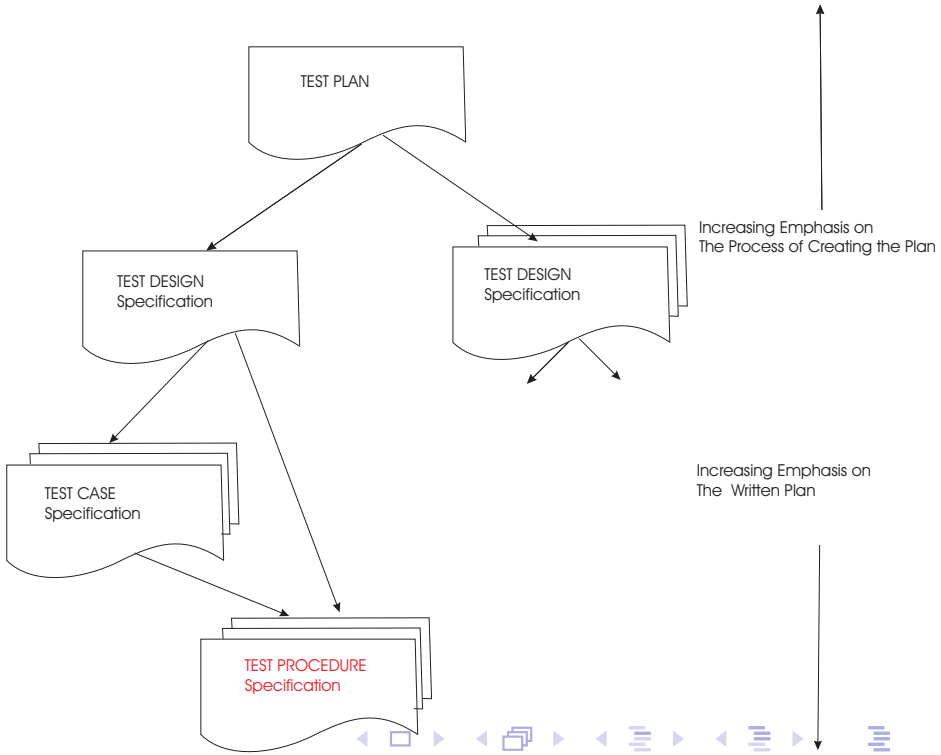
Planning test cases [Pat05]

- Test Design - topics
  - Identifiers, Features to be tested, Approach, Test case identification, Pass/fail criteria.
- Test Cases
  - Identifiers, Test item, Input and Output specification, Environmental needs, Special procedural requirements, Intercase dependencies.



Planning test cases [Pat05]

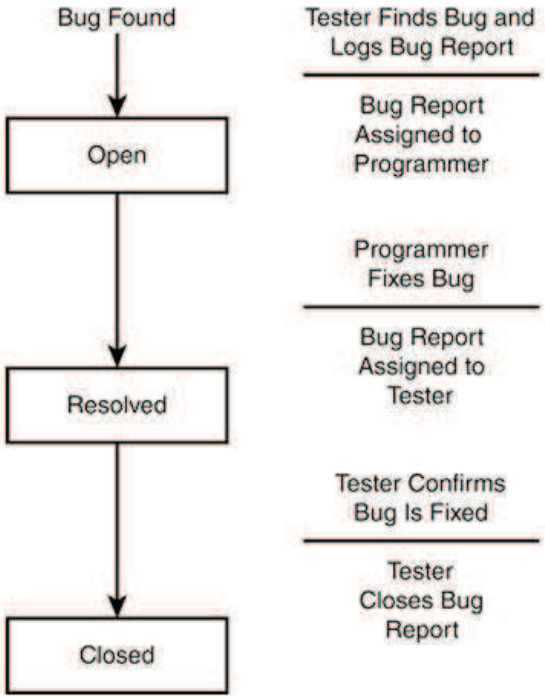
- Test Design - topics
  - Identifiers, Features to be tested, Approach, Test case identification, Pass/fail criteria.
- Test Cases
  - Identifiers, Test item, Input and Output specification, Environmental needs, Special procedural requirements, Intercase dependencies.
- Test procedures
  - Identifier, Purpose, Special req., Procedure steps.

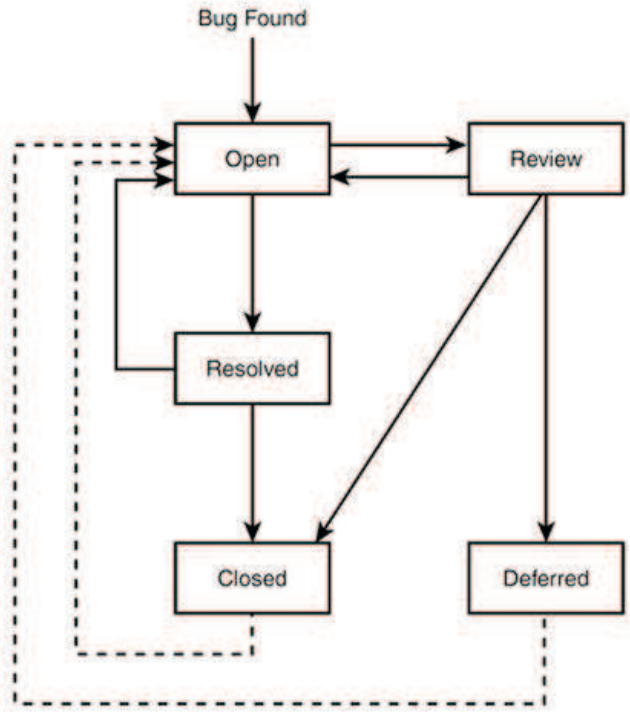


Reporting a bug

- Principles for reporting a bug [Pat05]
  - Report bugs as soon as possible
  - Effectively describe the bugs
  - Be nonjudgmental in reporting bugs
  - Follow up on your bug reports
- Isolating and reproducing bugs [Pat05] - suggestions in isolating a bug $\Rightarrow$
  - Don't take anything for granted
  - Look for time-dependent and race condition problems
  - White-box issues of boundary condition bugs, memory leaks, data overflows.
  - State bug
  - Resource dependencies and interactions with memory, network, hardware sharing.
  - Don't ignore the hardware

A bug's Life Cycle [Pat05]





Next Lecture (Still today!)

- Black-box testing

Bibliografie I

[Dij75] E. Dijkstra.
Guarded commands, nondeterminacy and formal derivation of programs.
CACM, 8(18):453–457, 1975.

[Fre10] M. Frentiu.
Verificarea și validarea sistemelor soft.
Presa Universitară Clujeană, 2010.

[IEE90] Ieee standard glossary of software engineering terminology.
https://standards.ieee.org/findstds/standard/610-1990.html, 1990.
Accessed: 2016-02-26.

[Mye04] G. Myers.
The Art of Software Testing, 2nd Edition.
John Wiley, 2004.

[Pat05] R. Patton.
Software Testing.
Sams Publishing, 2005.