Software Testing Basics

Instructor << >>

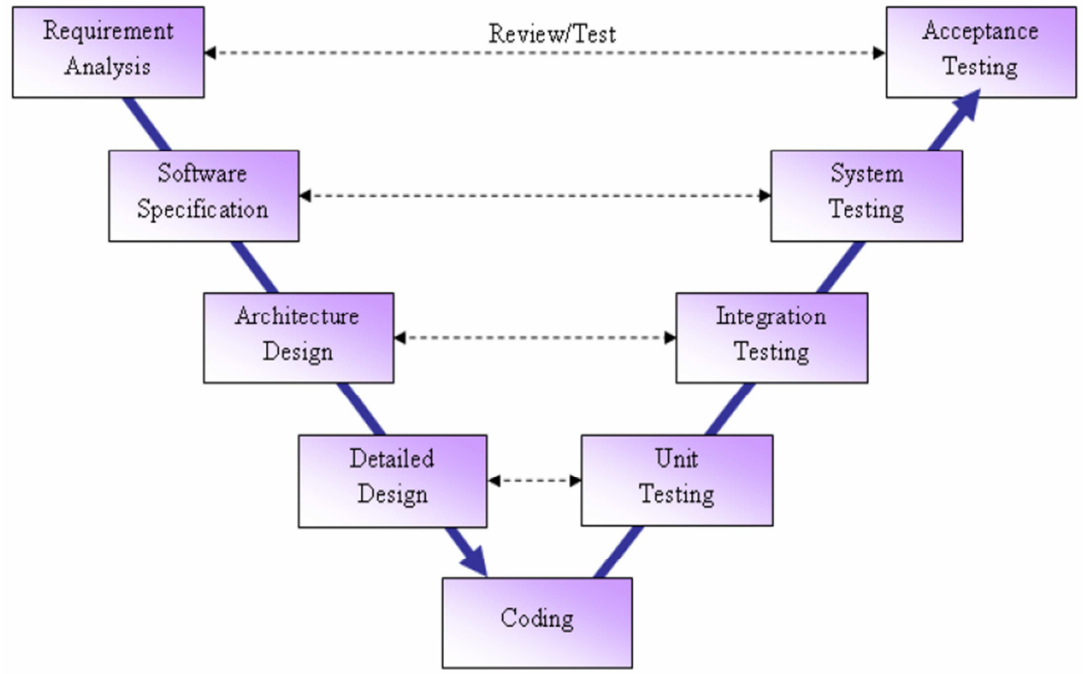
Agenda

- Testing Definition
- The Test Process
- Test Stages
- Testing types
- Test techniques /methods
- Test Documents
- Defect Concepts

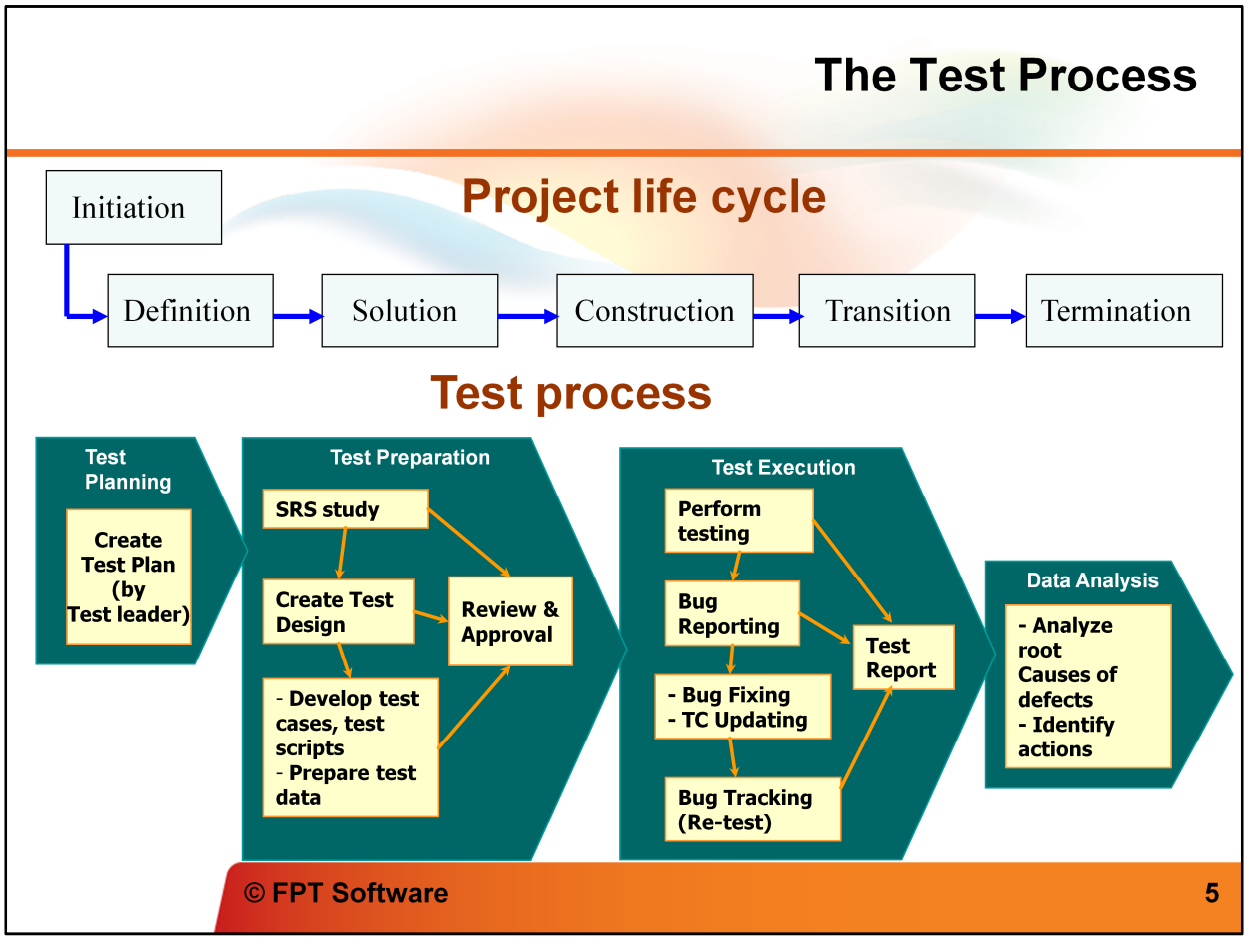
Testing Definition

- Testing is the process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements - (IEEE 83a)
- Testing Objectives
  - ***Primary:*** Execute a program with the intent of finding errors to
    - Determine whether system meets specifications
    - Determine whether system meets user's needs
  - ***Secondary****:* Continuously improve the testing process

Testing Definition
Where testing tasks are?



The Test Process

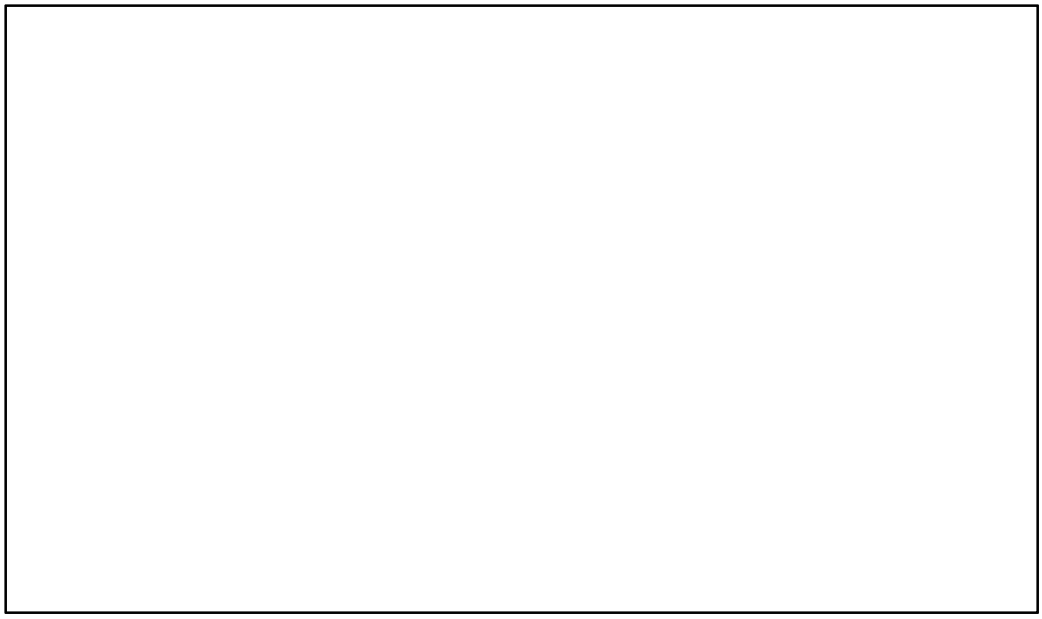


The Test Process
Inputs and Outputs

- Test Planning:
  - Input: Project plan, Customer Requirement & Acceptance criteria/SRS
  - Output: TP document
- Test Preperation:
  - Input: Test plan, SRS, detail design
  - Output: Test cases, test scripts, test data
- Test Execution:
  - Input: approved test documents: Test cases, test data, test scripts
  - Output: Test report, Defect list

The Test Process
Common Testing Resources

- **Guideline:** Test process & Test guideline
- **Templates** for test documents
  - Test Plan
  - Test case specification
  - Test report
  - Defect analysis report
- **Testing tools**
  - Defect tracking tool
  - Test Effort tracking tool
  - Test schedule
  - Test automation tools
    - *Rational Robot (Functional & Performance test)*
    - *OpenSTA (Open source)*
    - *Witir (Open source)*
- **Test Council**



Test Stages

- **There are 4 stages of Testing**
  - Unit Test
  - Integration Test
  - System Test
  - Acceptance Test

Testing Stages
Unit Test

- Purpose: to verify that the component/module functions properly
- Check:
  - internal data structures
  - Logic
  - boundary conditions for input/output data
- Normally: White box oriented
- Doer: Development team
- Metrics: Test coverage:
  - Statement
  - Branch
  - Path
  - Condition

Testing Stages
Integration Test

- Purpose:
  - To ensure that code is implemented and designed properly
  - to take unit tested modules and build a program structure that has been dictated by design

  → IT only after UT
- Combining the individual components to uncover errors associated with interfacing
- Normally: Black box oriented
- Doer: independent Test team

Testing Stages
System Test

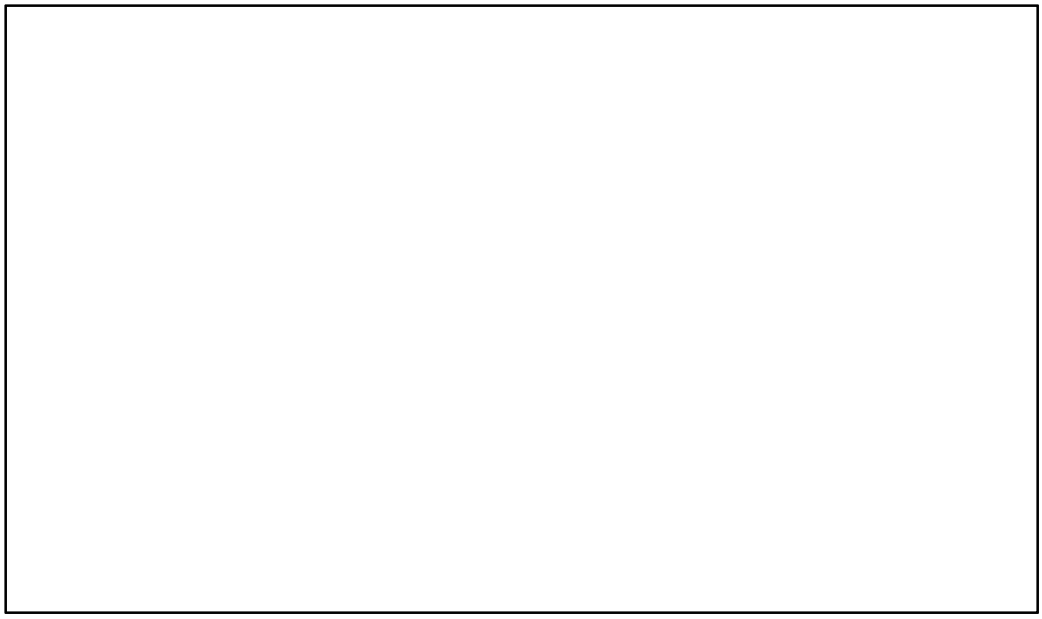
- Purpose: To ensure that the system does what the requirement specific
- Types
  - Function Testing
  - Performance Testing
  - Installation Testing
- Normally: Black box oriented
- Doer: independent Test team

Testing Stages
Acceptance Test

- Customer's way to verify that what was wanted is what is built
- Uncovers more than requirements discrepancies
- Allows the customers to determine what they really want, whether specified in the document or not.
- New problems may arise
- Customers may decide that the problem as changed and a different solution is needed
- Normally: Black box oriented
- Doer: Customer/independent Test team

Testing types

- **Functional test**
  - Function testing
  - User Interface testing
  - Data and database integrity testing
  - Business cycle testing
- **Performance test**
  - Performance profiling
  - Load testing
  - Stress testing
  - Volume testing
- **Access control & Security test**
- **Regression test**



Test techniques /methods

- **White Box Testing**
  - It guarantees that all independent paths within a module have been exercised at least once
  - Exercise all logical decisions on their true an false sides
  - Execute all loops their boundaries and within their operational bounds
  - Exercise internal data structures to assure their validity
- **Black Box Testing**
  - Focuses on functional requirement of the software
  - Derive sets of input conditions that will fully exercise all functional requirements for a program
  - Not an alternative to white box

Test Documents
Test Planning 1/2

- Test planning is to:
  - Define Test strategy:
    - What to test: list of requirements to test
    - How to test (Test techniques)
    - Test types
    - Test tool, ….
  - Define the responsibilities
  - Decide test metrics & manage through metrics
  - “When to Stop Testing?”
    - Not good Unit Test
    - Low quality of code,…
    - Meet requested Test coverage
    - All Test cases are tested and passed

Test Documents
Test Plan 2/2

- Test Plan to define:
  - Scope of test: stages and types of test
  - Risks may affect testing: quantify the loss of the risk
  - Training needs for testers
  - Requirements to test: what will be tested
  - Testing strategy: how will the test be performed
  - Completion criteria (Pass/Fail)
  - Criteria to stop testing
  - Tools to be used for testing
  - Testing resource and environment (Responsibilities of testers)
  - Test Milestone: schedule of testing activities
  - Test deliverables: outputs of testing

Test Documents
Test design 1/2

- **Specifications**
  - Basic flow
  - Alternative flow
- **Test Design**
  - All test cases should be traceable to requirements
  - Design framework and rules for test cases (large/average items)
  - Plan number of test cases for normal, abnormal & boundary data
  - Assign testers for module/function/large items

Test Documents
Test Design 2/2

- **Structure/Framework for Test cases (TC):**
  - Modules or functions
  - Large items
  - GUI
  - Functions
  - Performance
- **Number of TC**:
  - Total
  - For each module
  - For each type: GUI/Function
  - For normal/abnormal/boundary cases

Test Documents
Test Case

- Test environment
- Test condition
- A test case includes:
  - Action (sample: click on Add button, on a link, input birth day…)
  - Input data (valid/invalid/boundary data)
  - Expected result (message, …)

❖ **Good test case:**

- Structure of TC is clear and reasonable.
- Follow requirement closely.
- Cover all of cases that can occur.
- Description and test conditions is brief and easy to understand
- Procedure and expected result is a step by step process

Test measurements

- Product: We've tested 80% of the lines of code
- Plan: We've run 80% of the test cases
- Results: We've discovered 593 defects
- Effort: We've worked 80 hours a week on this for 4 months. We've run 7,243 tests
- Quality of Testing: Beta testers have found 30 defects that we missed. Our regression tests seems ineffective
- History across projects: At this milestone on previous projects, we had fewer than 12% of the defects found still open. We should be at that percentage on this product too

Test Measurements
Common Testing Measurements

- **Measurements:**
  - Defects
  - Test effort
  - Defect rate: Weighted defects/ project size (in UCP)
  - Test coverage: number of executed test cases/ Total number of TCs
  - Test successful coverage: number of test cases executed successfully/total number of test cases
  - Defect removal efficiency
- **Metrics to assess tester performance:**
  - **Test effectiveness**
    - Weighted defects/ Test execution effort
  - **Leakage**
    - Weighted defects found after release/ project size

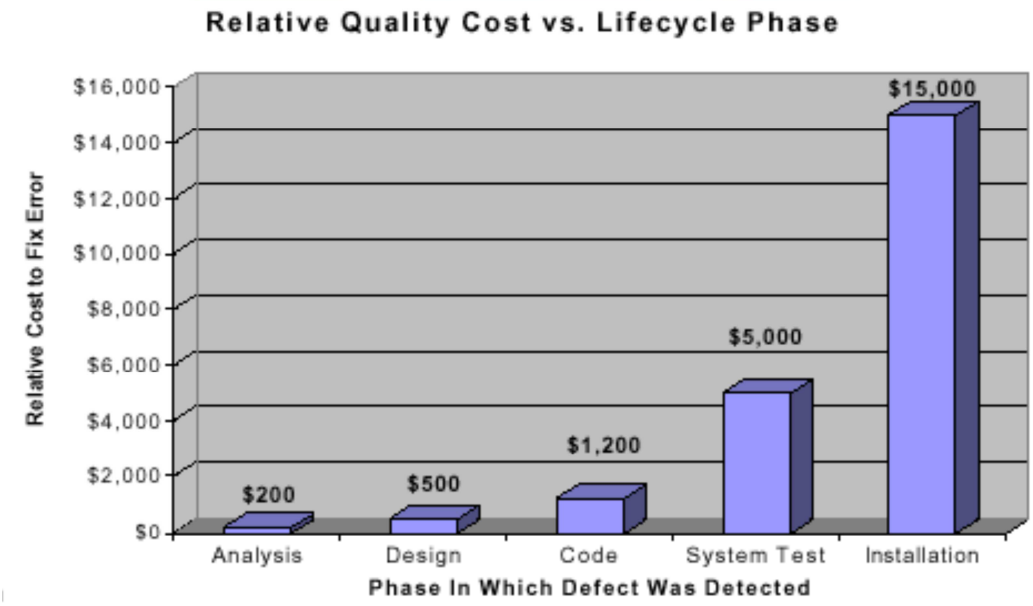
Defect Concepts

- What is defect?

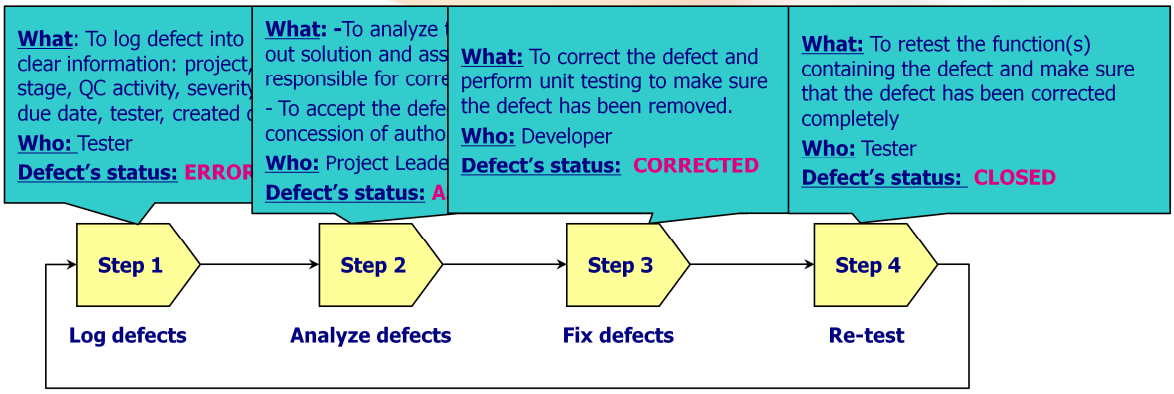
  A defect is any error found by testing and reviewing activities (All errors found by internal reviewer, external reviewer and customer).

- Where defects come from?
  - Products
    - SRS, DD, Source code, Test cases, etc.
  - Quality Control
    - Review, Test, Audit, Inspection
  - Processes
    - Requirements, Design, Coding, Test, etc.
  - Other sources:
    - Change requests, misunderstanding, carelessness, planning, etc.

Defect Concepts
Cost of defects



Defect Concepts
Fsoft Defect Lifecycle



Defect still exists in the product, change status to **ERROR**

- Defect classification:
  - Open defects
  - Closed defects
  - Leakage

Defect Concepts
Defect Classification 1/3

Defect status:

- Error
- Assigned
- Corrected
- Cancelled
- Accepted: by PM/PTL/Customer
- Closed: by tester/PM/PTL

Leakage Defects:

- Acceptance test
- After Release test
- After Release review

Defect Concepts
Defect Classification 2/3

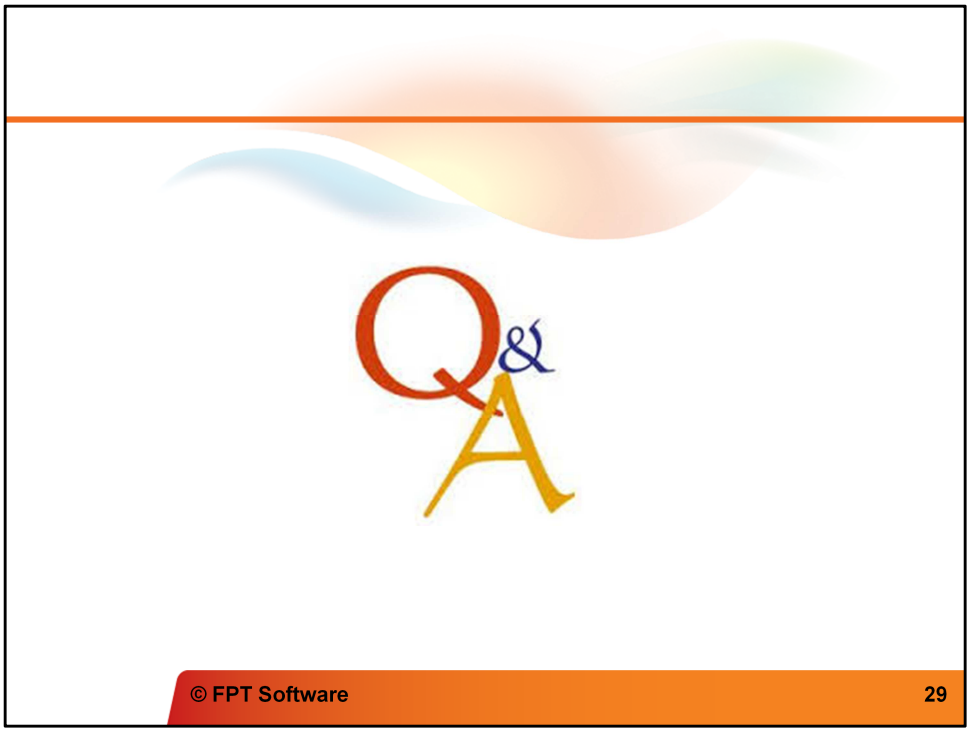
- **Defect type**
  - Functionality
    - Requirement misunderstanding
    - Feature missing
    - Coding logic
    - Business logic
  - User Interface
  - Performance
  - Design issue
  - Coding standard
- **Severity**: How bad the problem is
  (some time disagreement btw. Dev & Tester)
  - Fatal: the system is broken
  - Serious: can not work around
  - Medium: impact the function
  - Cosmetic: not impact the function

Defect Concepts
Defect Classification 3/3

- **Work product**
  - Software module
  - Software Package
  - ADD
  - DDD
  - Test Cases (UT/IT/ST)
- **Quality control activity**
  - Code review
  - Unit test
  - Integration test
  - System test
  - Document review
  - Inspection
  - Audit
- **Defect origin**
  - Requirement
  - Design
  - Coding
  - Test
  - CM
  - Document control
- **Priority**
  - Immediately
  - High priority
  - Normal priority
  - Low priority

Defect Concepts
Identify Defects

- Defect can be caused by a flaw in the application software or by a flaw in the application specification.
- Two approaches are commonly used for identifying defects: reviews and testing.
  - Testing can be used for identifying defects only in executable systems.
  - Reviews are more general and can be applied even to documents or work products that can not be executed.


Q&A
© FPT Software
29