Software Engineering Principle

(SEP521)

Software Testing - I

Jongmoon Baik

KOREA ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY
KAIST
한국과학기술원
SINCE 1971

Objectives

- To define and understand what software testing is
- To understand software testing strategies
- To describe software testing processes
- To understand different software testing approach
  - Different testing levels and approaches
  - Whitebox vs. Blackbox tests

What is Software Testing?

"[**Software testing**] is the design and implementation of a special kind of software system: one that exercises another software system with the intent of **finding bugs**."

Robert V. Binder, *Testing Object-Oriented Systems: Models, Patterns, and Tools* (1999)

"[Software testing] is the technical operation that consists of the determination of one or more characteristics of a given product, process, or service according to a specified procedure."

ISO/IEC 1991

What is Software Testing?

- Software Testing typically involves:
  - Execution of the software with representative inputs under an actual operational conditions
  - Comparison of predicted outputs with actual outputs
  - Comparison of expected states with resulting states
  - Measurement of the execution characteristics (execution time, memory usage, etc)

Goals of Software Testing

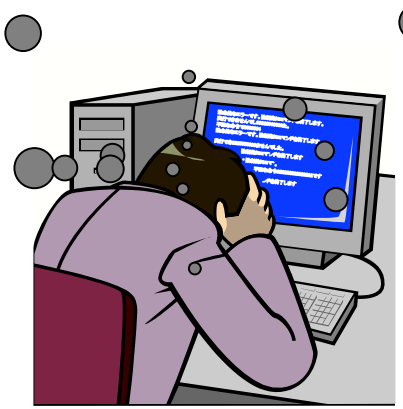
- Immediate goal
  - Identify discrepancies b/w actual results and expected behavior
- Ultimate goal
  - Demonstrate conformance to specification
    - Guaranteeing correctness is not realistic in general.
  - Provide an indication of correctness, reliability, safety, security, performance, usability, etc.
- Practical goal
  - Improve software reliability at an acceptable cost
    - Increase its usefulness

Software Testing Issues

Is an exhaustive test possible?

How much testing is required?

How to find a few input that represent the entire input domain?



When we can stop testing?

How to generate test data?

Which procedure is proper?

Test Case, Test Oracle, & Test Bed

- Test Case:
  - A test related items which contains the following information
    - A set of test inputs
    - Execution conditions
    - Expected outputs
- Test Oracle:
  - A document, or piece of software that allows tester to determine whether a test has been passed or failed
    - e.g.: a specification (pre- and post conditions), a design document, and a set of requirements
- Test Bed
  - An environment that contains all the hardware and software needed to test a software component or a software system

Software Testing Principles

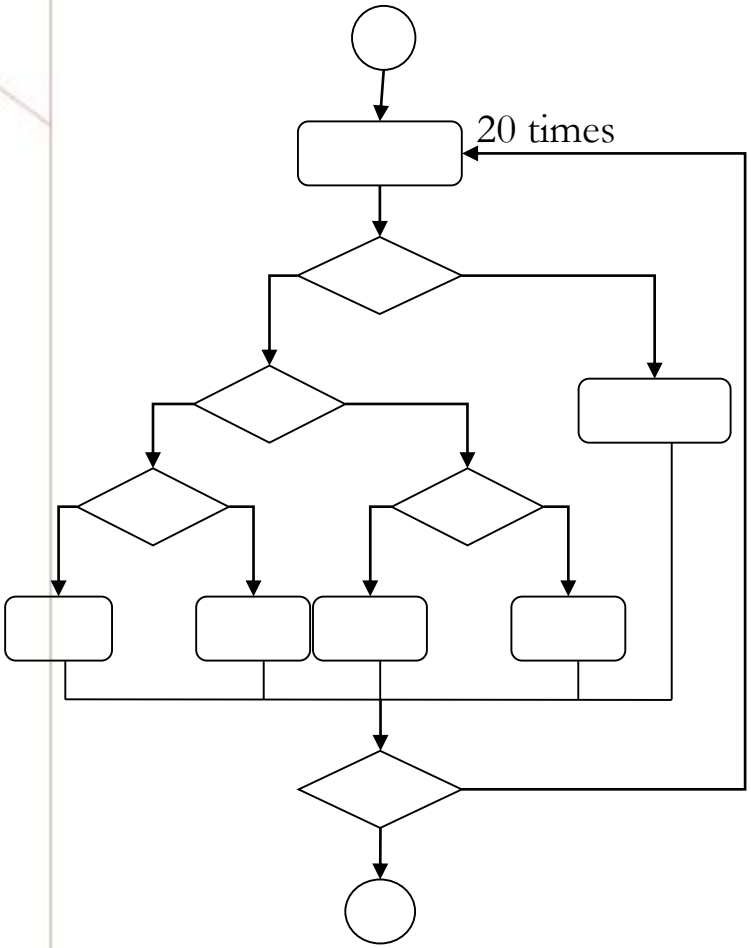
1. Testing is the process of exercising a software component using a selected set of test cases, with the intent of
   - Revealing defects
   - Evaluating quality
2. When the test objective is to detect defects, then a good test case is one that has a high probability of revealing a yet undetected defect (s)
3. Test results should be inspected meticulously
4. A test case must contain the expected output or result
5. Test cases should be developed for both valid and invalid input conditions

Software Testing Principles

6. The probability of the existence of additional defects in a software component is proportional to the number of defects already detected in that component.
7. Testing should be carried out by a group that is independent of the development group
8. Tests must be repeatable and reusable
9. Testing should be planned
10. Testing activities should be integrated into the software life cycle
11. Testing is a creative and challenging task

Ilene Burstein, "Practical Software Testing", 2003

Exhaustive Testing



- How many possible paths?
  - About 10^{14} paths
- Assume that one test case/millisecond
- How long it will take to execute all the test cases?
  - **3170 years** (24 hrs, 7days working)

Go with
SELECTIVE TESTING

What is S/W Testing Strategy?

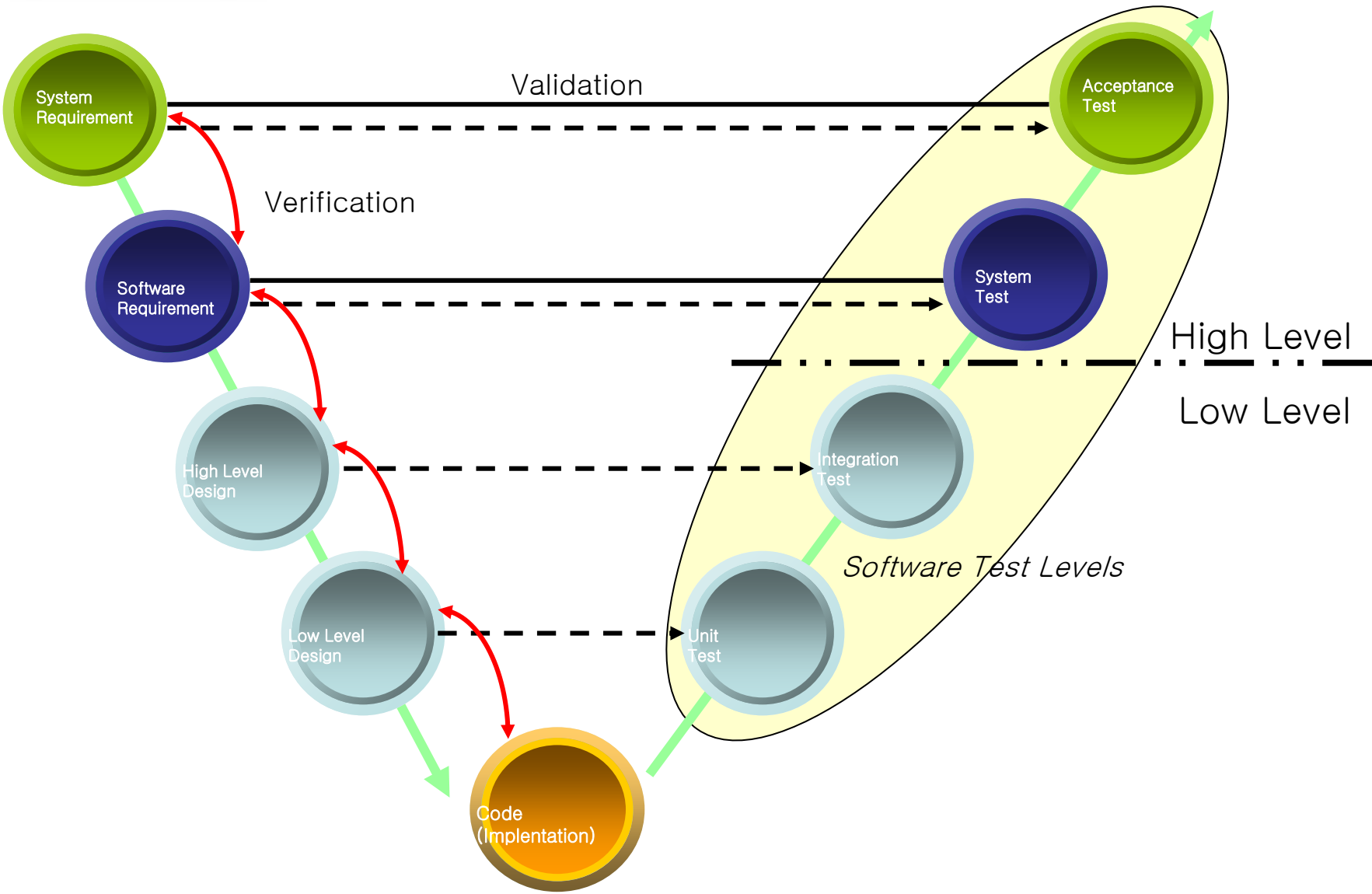
"An elaborate and systematic plan of series of actions to be taken for software testing to locate defects and remove them before the software system is released"

"No Universal Testing Strategy"

Questions to be Answered

- How do you conduct the tests?
- Should you develop a formal plan for your tests? If then When?
- Should you test the entire program as whole or run tests only on a small part of it?
- Should you rerun tests you've already conducted as you add new components to a large system?
- When should you involve the customer?

A Testing Strategy



Testing Scope

- "Testing in the small"
  - Exercising the smallest executable unit of the software system
- "Testing in the large"
  - Putting the entire system to the test
- Begin with "Testing in the small" and then progress to "Testing in the large"

Characteristics for S/W Testing Strategies

- To perform effective testing, a software team should conduct effective formal technical reviews
- Testing begins at the components level and works “outward”
- Different testing techniques are appropriate at different point in time
- Testing is conducted by developer and an independent test group
- Testing and debugging are different activities, but debugging must be accommodated in any testing strategy

A Way of Thinking about Testing

- Constructive tasks – Software Analysis, Design, Coding, Documentation
- Destructive tasks – Testing
  - Considered to break the software
- Developers treats lightly, designing and executing tests that will demonstrate that the program works, rather than finding errors

Software Testing Organization

Software Developers

- Responsible for testing the individual unit of the program
- Also conduct the integration testing
- Correct uncovered errors

ITG (Independent Testing Group)

- Remove the inherent problems of developers' testing their programs
- Report to SQA organization – Independence

Close Collaboration b/w Development Group and ITG are required for successful tests throughout the life Cycle

Strategic Issues

- Specify product requirements in a quantifiable manner long before testing commences.
- State testing objectives explicitly.
- Understand the users of the software and develop a profile for each user category.
- Develop a testing plan that emphasizes “rapid cycle testing”.

Tom Gilb ’85

Strategic Issues (Cont.)

- Build “robust” software that is designed to test itself
- Use effective formal technical reviews as a filter prior to testing
- Conduct formal technical reviews to assess the test strategy and test cases themselves
- Develop a continuous improvement approach for testing process

Characteristics of "Testability"

- Operability—it operates cleanly
- Observability—the results of each test case are readily observed
- Controllability—the degree to which testing can be automated and optimized
- Decomposability—testing can be targeted
- Simplicity—reduce complex architecture and logic to simplify tests
- Stability—few changes are requested during testing
- Understandability—of the design

James Bach, 1994

What is a good test?

- A high probability of finding an error
- Not redundant
- “Best of breed”
- Neither too simple nor too complex

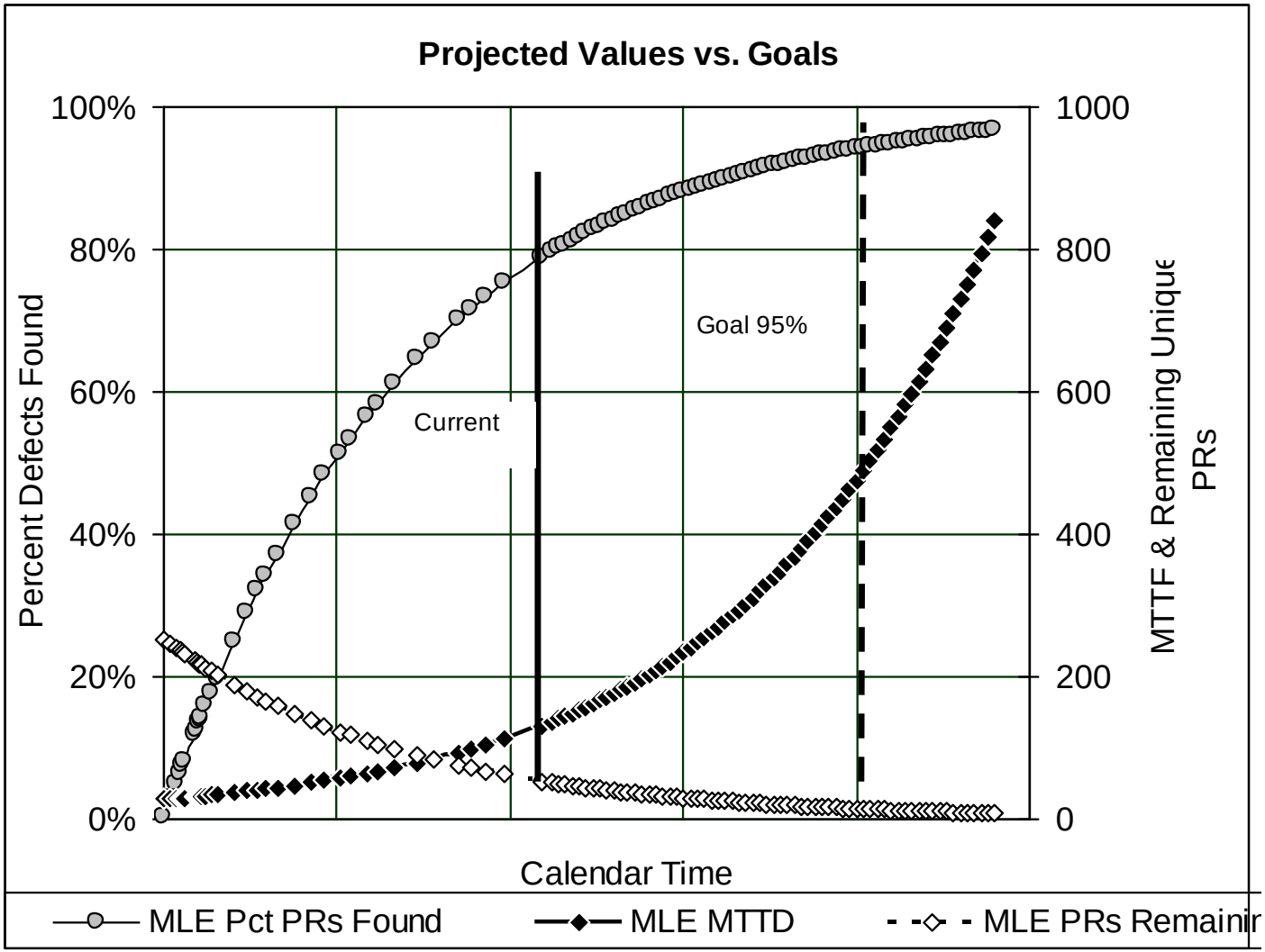
When to Stop Testing?

- No definitive answer to the question

- A few pragmatic answers
  - You're never done testing
  - Stop when you're out of budget, money, or schedule
  - Stop based on statistical criteria
    - e.g.: statistical modeling, software reliability theory

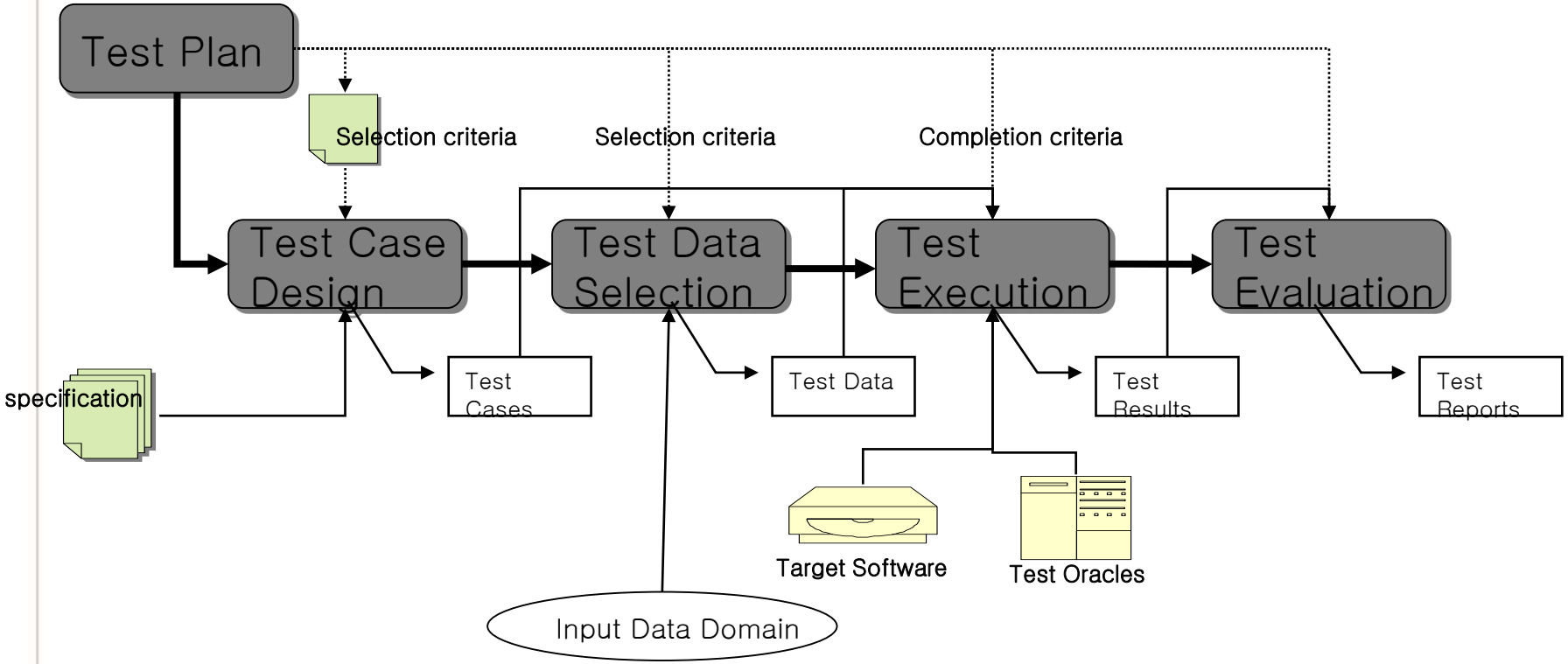
- Rule of Thumb
  - Stop when error detection rate drops

Example: Reliability Growth Model



Software Test Process

- Included in a entire software development process
- Test plan start with requirement analysis
- Test plan & Test procedure: systematic and performed in parallel with s/w development



Test Planning

- Start after requirement spec. and dev. Plan is made
- Goal of Test Plan
  - Define the scope, approach, resources, and schedule of the intended testing activities
- Activities of test planning
  - Definition of what will be tested and the approach that will be used
  - Mapping of tests to the specifications
  - Definition of the entry and exit criteria for each phase of testing
  - Organization of a team
  - Estimation of the time & effort needed
  - Schedule of major milestones
  - Definition of the test environment (hardware and software)
  - Definition of the work products for each phase of testing.
  - An assessment of test-related risks and a plan for their mitigation

Test Design

Test Design phase includes the following activities:

- Testability review of the test basis
- Define testing units and pass/fail criteria
- Allocating test techniques
- Specify the testing environment
- Generate test procedure
- Produce test cases/scripts/scenarios
- Preparing test specification

Test Execution

- Begins with the first delivery of testable component (s)
- Check the completeness of the test object & test environment
- Install the test object into the environment
- Execute (re) tests
- Compare and Analyze the test results

Test Evaluation

- Evaluate the test object
- Evaluate the test process
- Produce an evaluation report
- Preserve testware
- Discharge the test team

Software Test Management Practices

- Initial test planning activities
  - including planning for V&V activities at each lifecycle phase
- Defining and managing test objectives
  - test requirements
    - include the functions and logic to be tested, program constraints, software states, input and output data conditions, and usage scenarios
    - can be selected based on concerns, risks, and business logic
- Test progress monitoring activities.
- Product and test quality tracking activities.
- Test configuration control activities

CROSSTALK, June 1996

Q & A