

Chapter 2

Software Testing Terminology and Methodology

Objectives

- Difference between error, fault and failure.
- Life Cycle of a bug.
- How does a bug affect economics of software testing?
- How does a bug classified?
- Testing Principles
- Software Testing Life Cycle (STLC) and its models.
- Difference between verification and validation.
- Development of software testing methodology

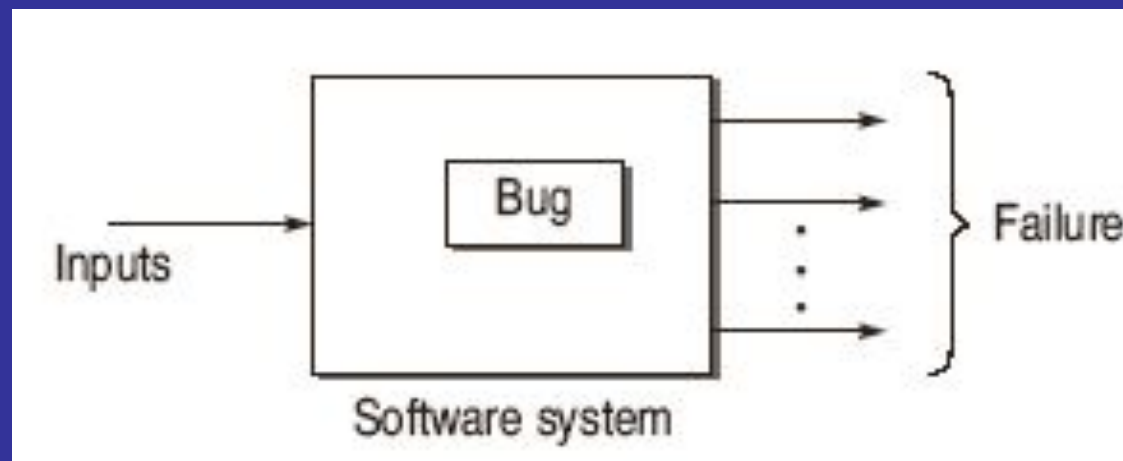
Software Testing Terminology

- **Failure**

The inability of a system or component to perform a required function according to its specification.

- **Fault / Defect / Bug**

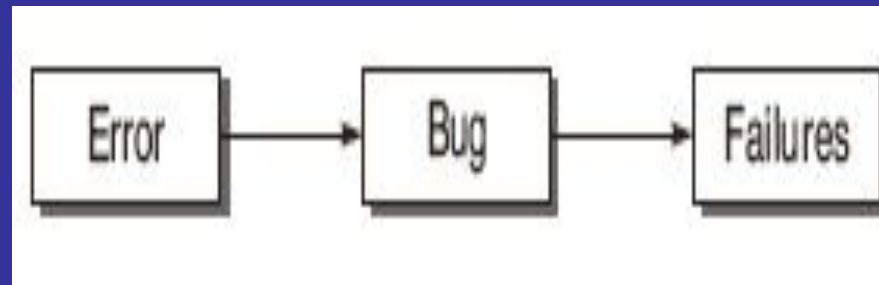
Fault is a condition that in actual causes a system to produce failure.



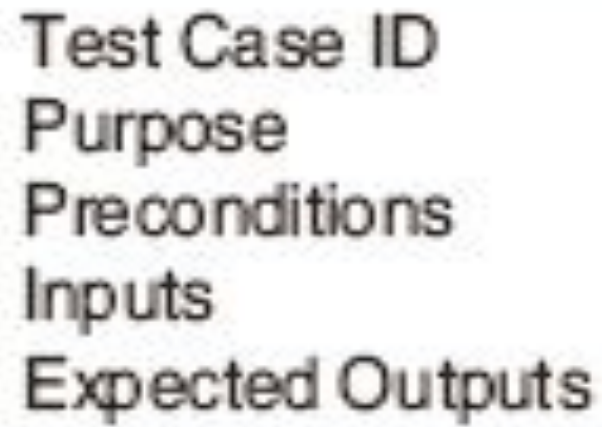
Software Testing Terminology

Error

Whenever a member of development team makes any mistake in any phase of SDLC, errors are produced. It might be a typographical error, a misleading of a specification, a misunderstanding of what a subroutine does and so on. Thus, error is a very general term used for human mistakes.



Software Testing Terminology



Test Case ID
Purpose
Preconditions
Inputs
Expected Outputs

- **Testware**

The documents created during the testing activities are known as Testware.

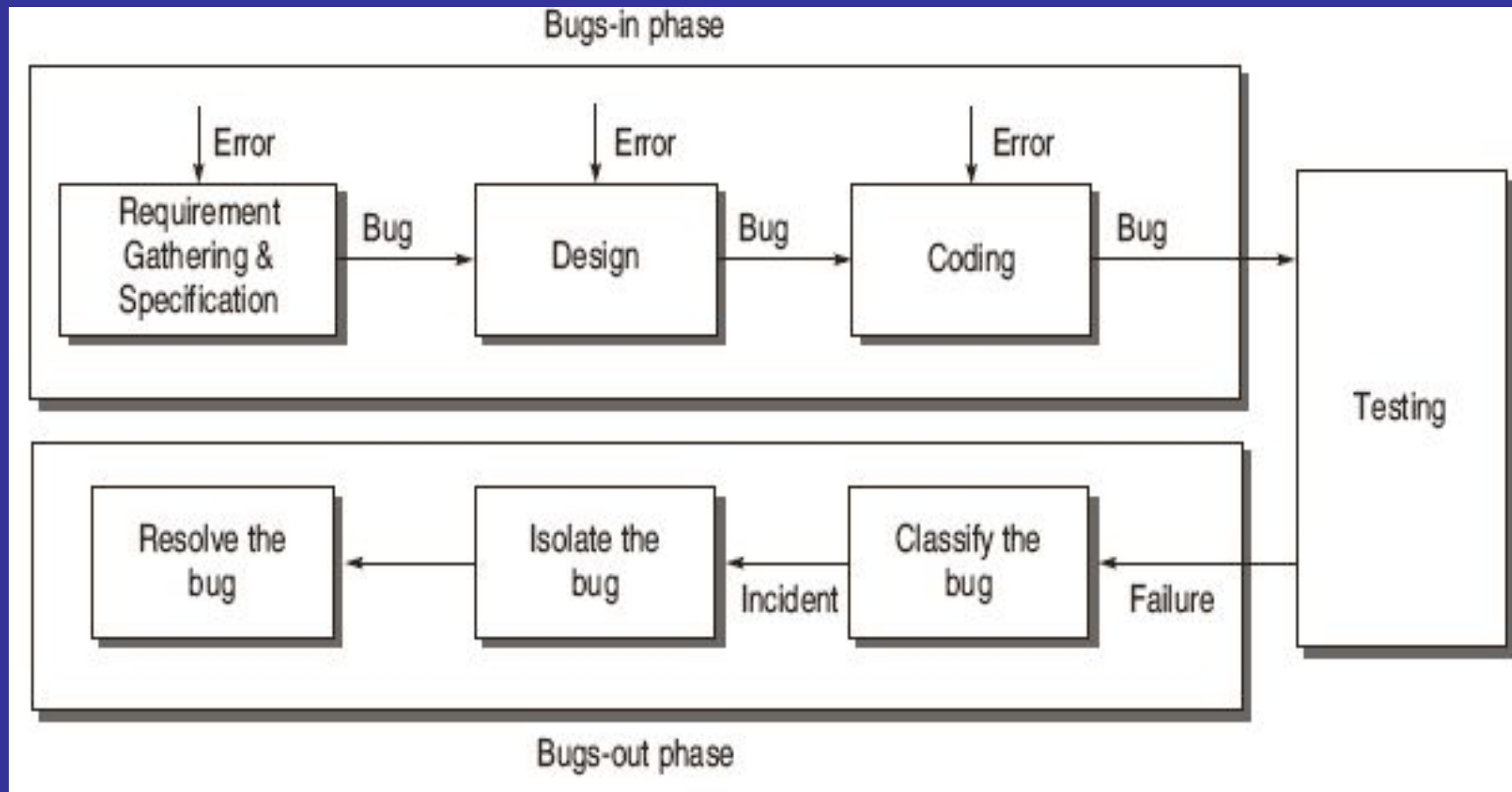
- **Incident**

the symptom(s) associated with a failure that alerts the user to the occurrence of a failure.

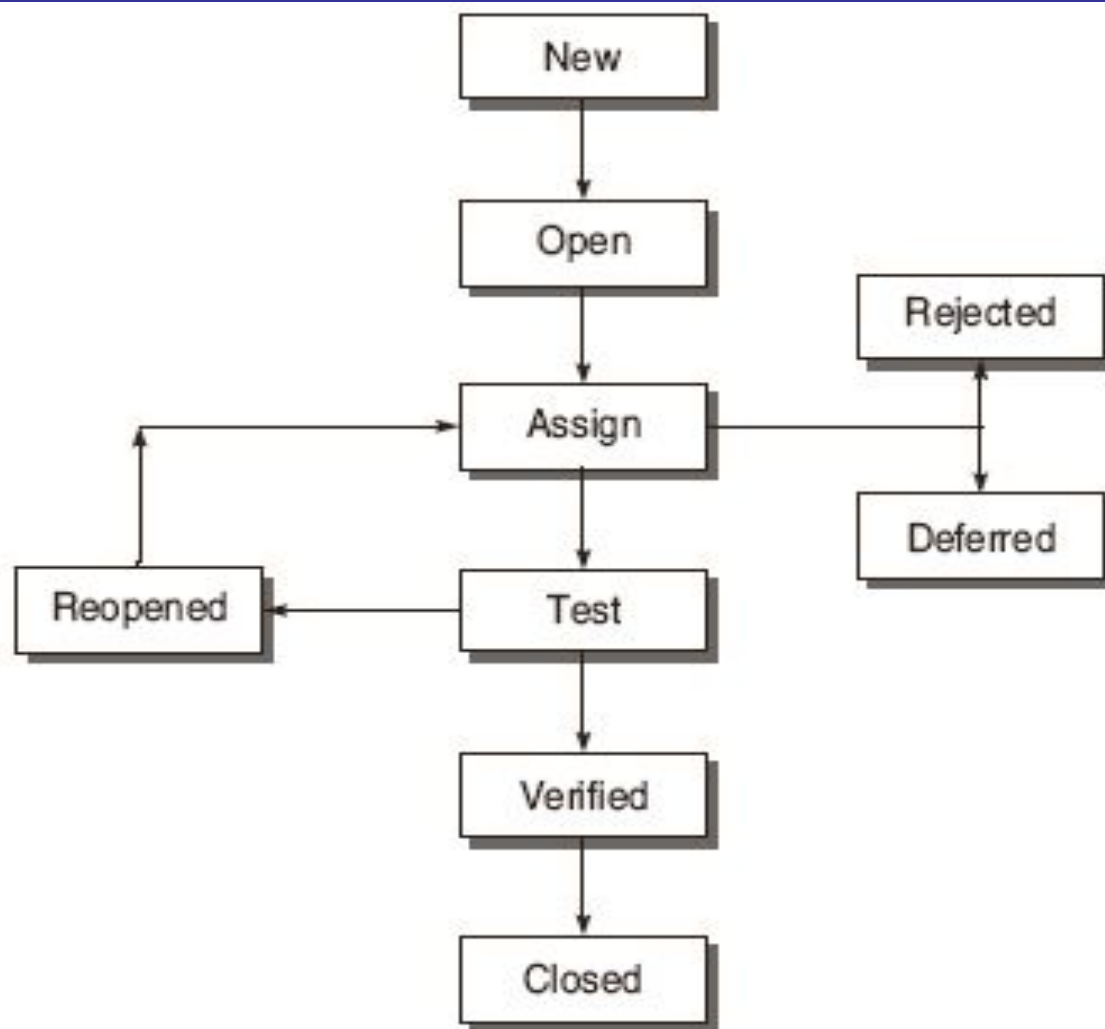
- **Test Oracle**

to judge the success or failure of a test,

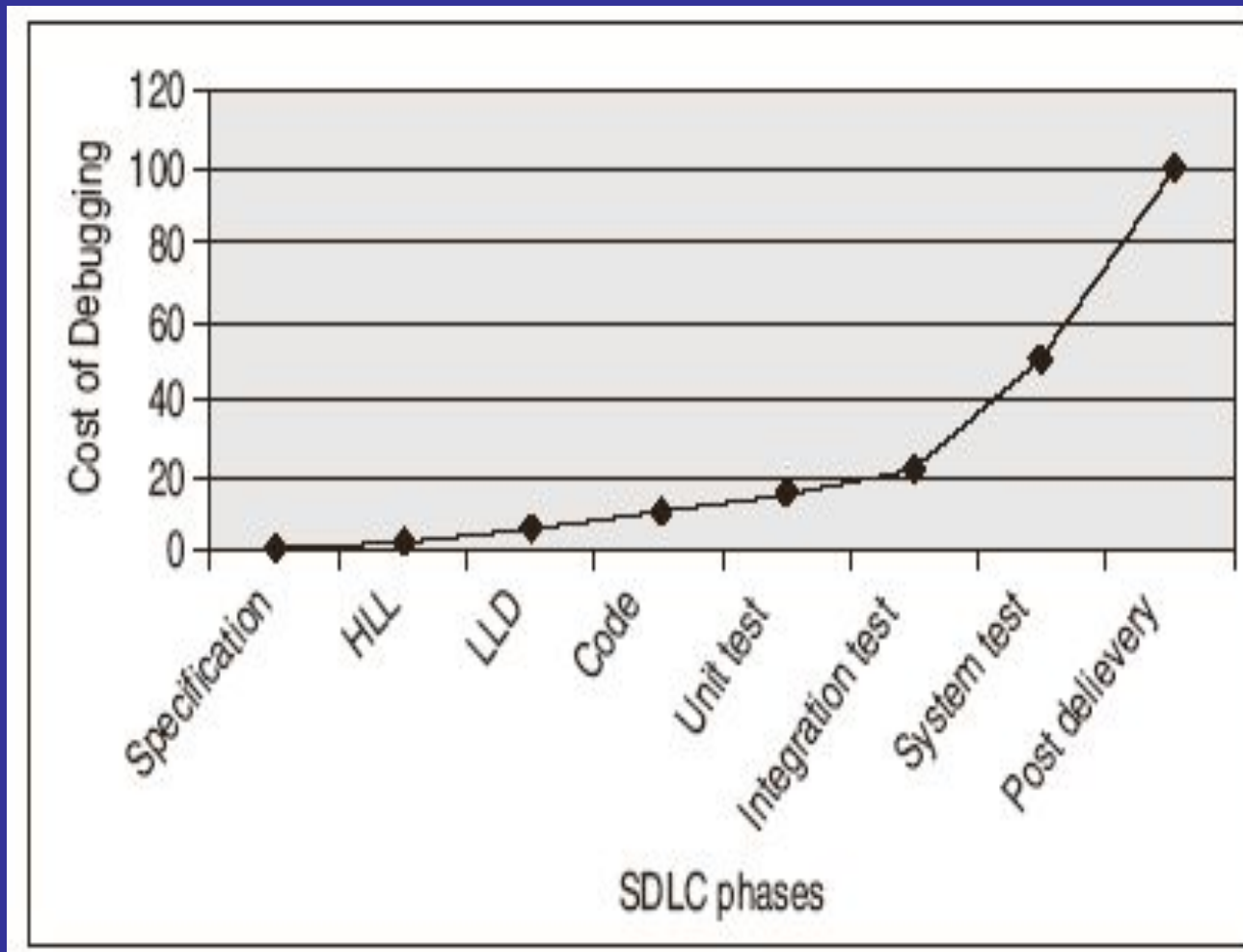
Life Cycle of a Bug



States of a Bug



Bug affects Economics of Software Testing



Bug Classification based on Criticality

- **Critical Bugs**

the worst effect on the functioning of software such that it stops or hangs the normal functioning of the software.

- **Major Bug**

This type of bug does not stop the functioning of the software but it causes a functionality to fail to meet its requirements as expected.

- **Medium Bugs**

Medium bugs are less critical in nature as compared to critical and major bugs.

- **Minor Bugs**

Bug Classification based on SDLC

Requirements and Specifications Bugs

Design Bugs

Control Flow Bugs

Logic Bugs

Processing Bugs

Data Flow Bugs

Error Handling Bugs

Race Condition Bugs

Boundary Related Bugs

User Interface Bugs

Coding Bugs

Interface and Integration Bugs

System Bugs

Testing Bugs

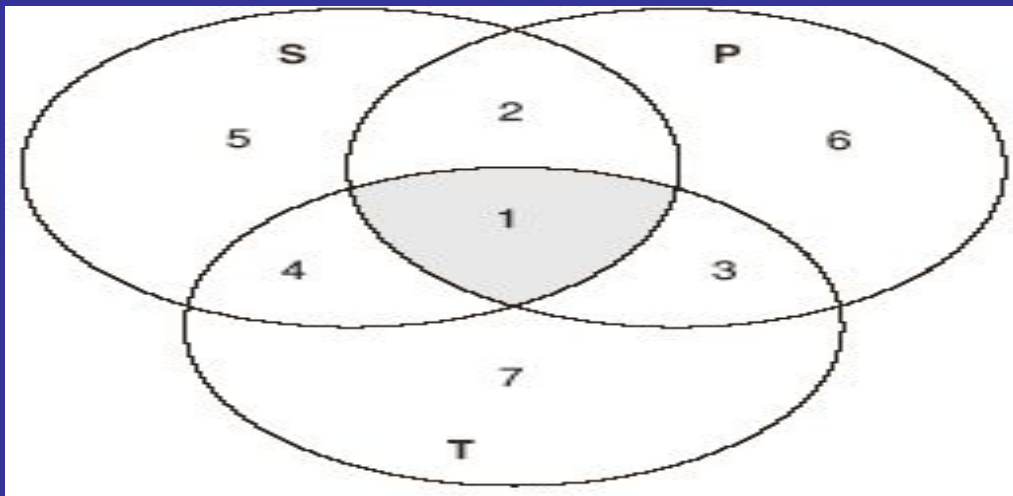
- **Effective Testing not Exhaustive Testing**
- **Testing is not a single phase performed in SDLC**
- **Destructive approach for constructive testing**
- **Early Testing is the best policy.**
- **The probability of the existence of an error in a section of a program is proportional to the number of errors already found in that section.**
- **Testing strategy should start at the smallest module level and expand toward the whole program.**

Testing Principles

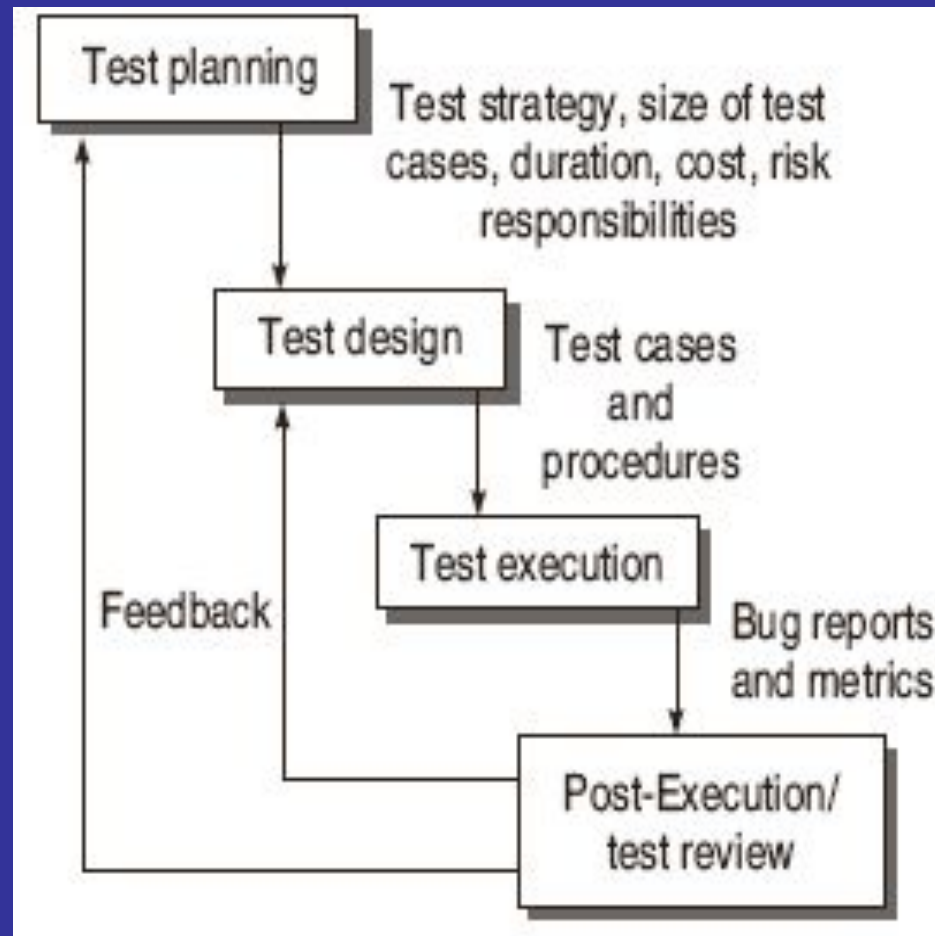
Testing should also be performed by an independent team.
Everything must be recorded in software testing.

Invalid inputs and unexpected behavior have a high probability of finding an error.

Testers must participate in specification and design reviews.

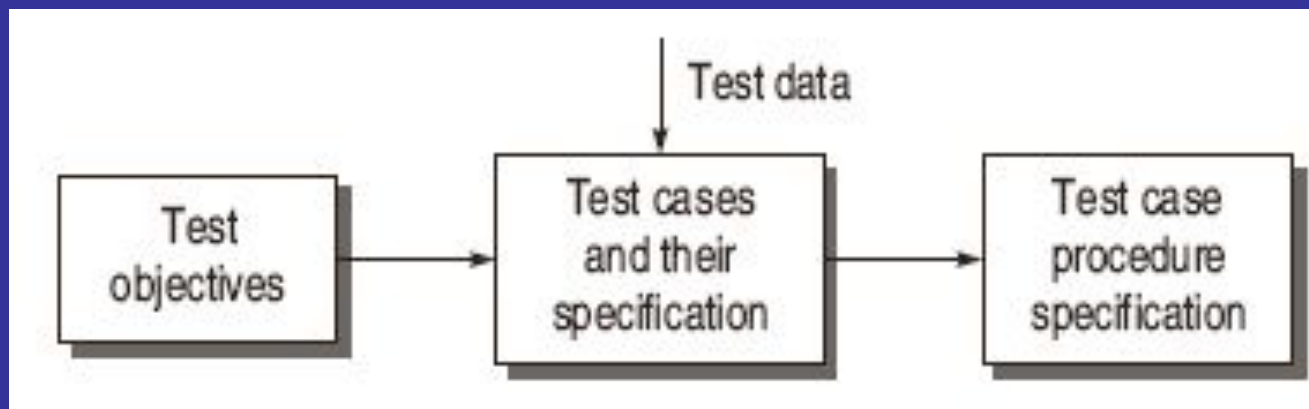


Software Testing Life Cycle (STLC)

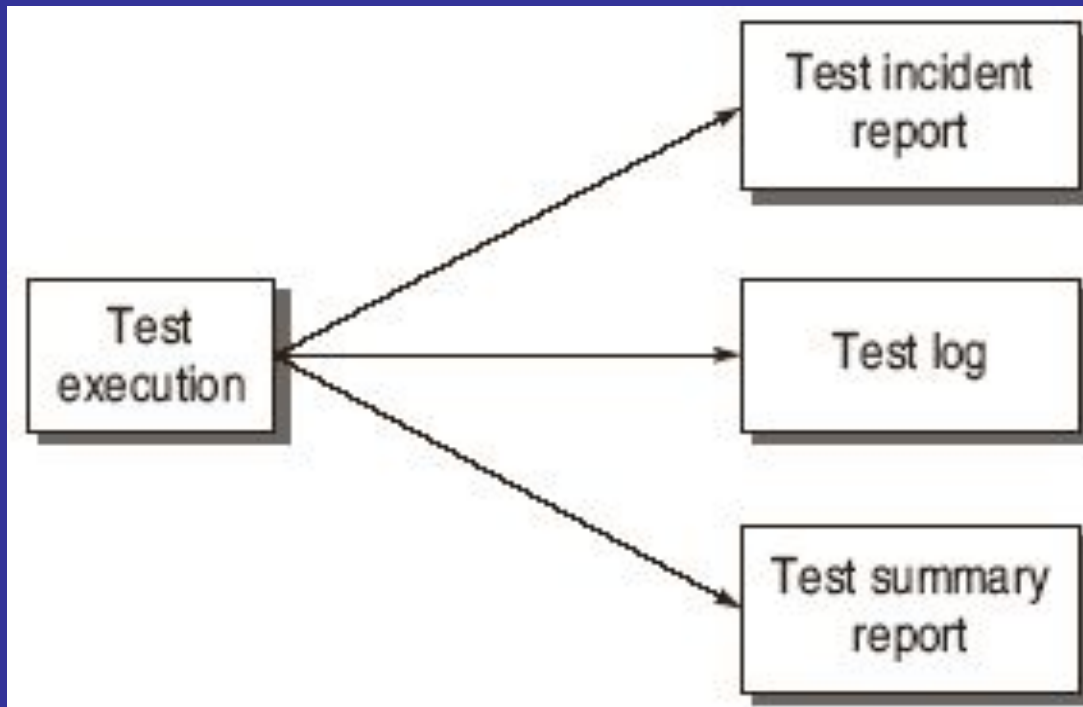


- Defining the Test Strategy
- Estimate of the number of test cases, their duration and cost.
- Plan the resources like the manpower to test, tools required, documents required.
- Identifying areas of risks.
- Defining the test completion criteria.
- Identification of methodologies, techniques and tools for various test cases.
- Identifying reporting procedures, bug classification, databases for testing, Bug Severity levels, project metrics

- *Determining the test objectives and their Prioritization*
- *Preparing List of Items to be Tested*
- *Mapping items to test cases*
- *Selection of Test case design techniques*
- *Creating Test Cases and Test Data*
- *Setting up the test environment and supporting tools*
- *Creating Test Procedure Specification*

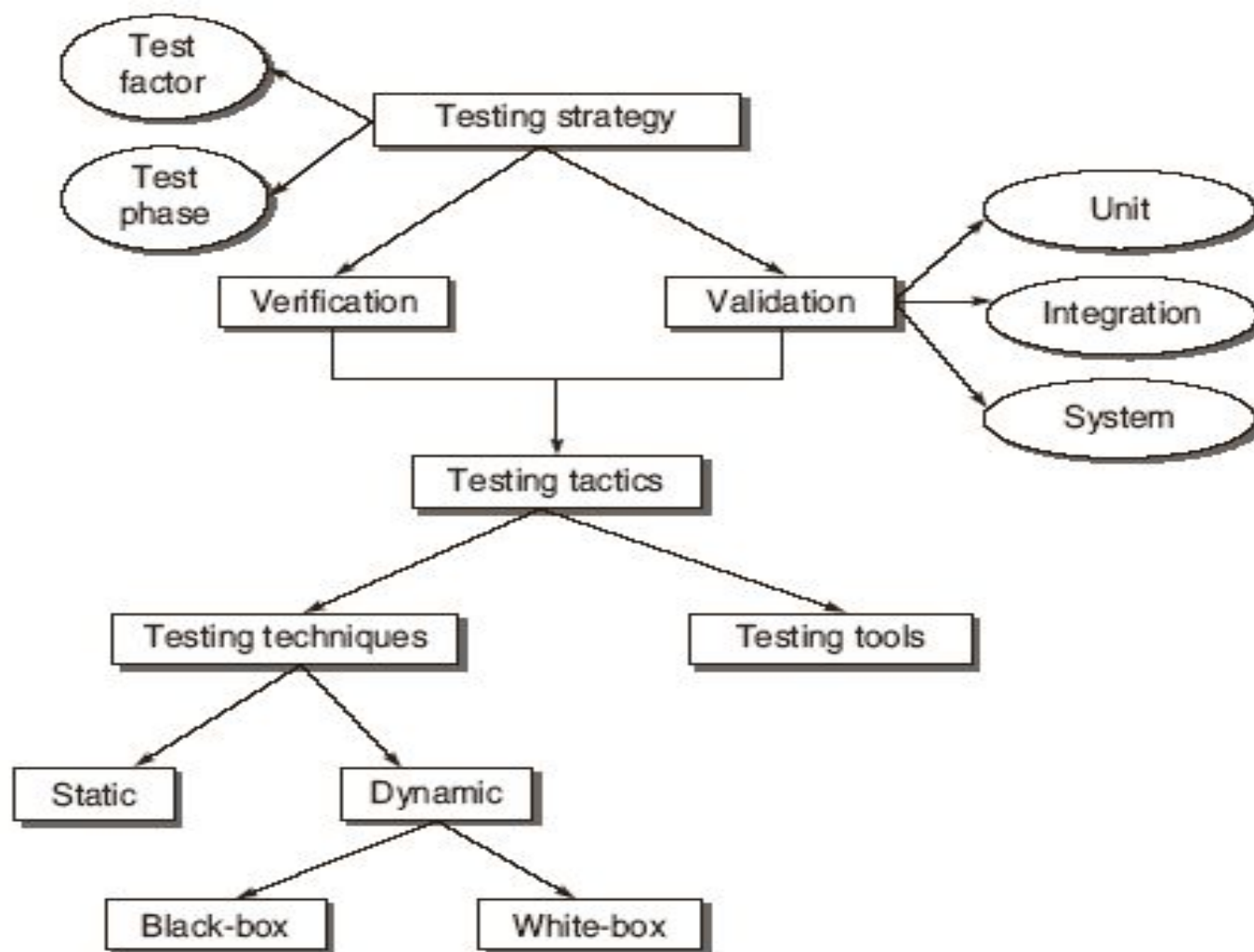


Test Execution



- Understanding the Bug
- Reproducing the bug
- Analyzing the nature and cause of the bug
- *Reliability analysis*
- *Coverage analysis*
- *Overall defect analysis*

Software Testing Methodology



Test Strategy Matrix

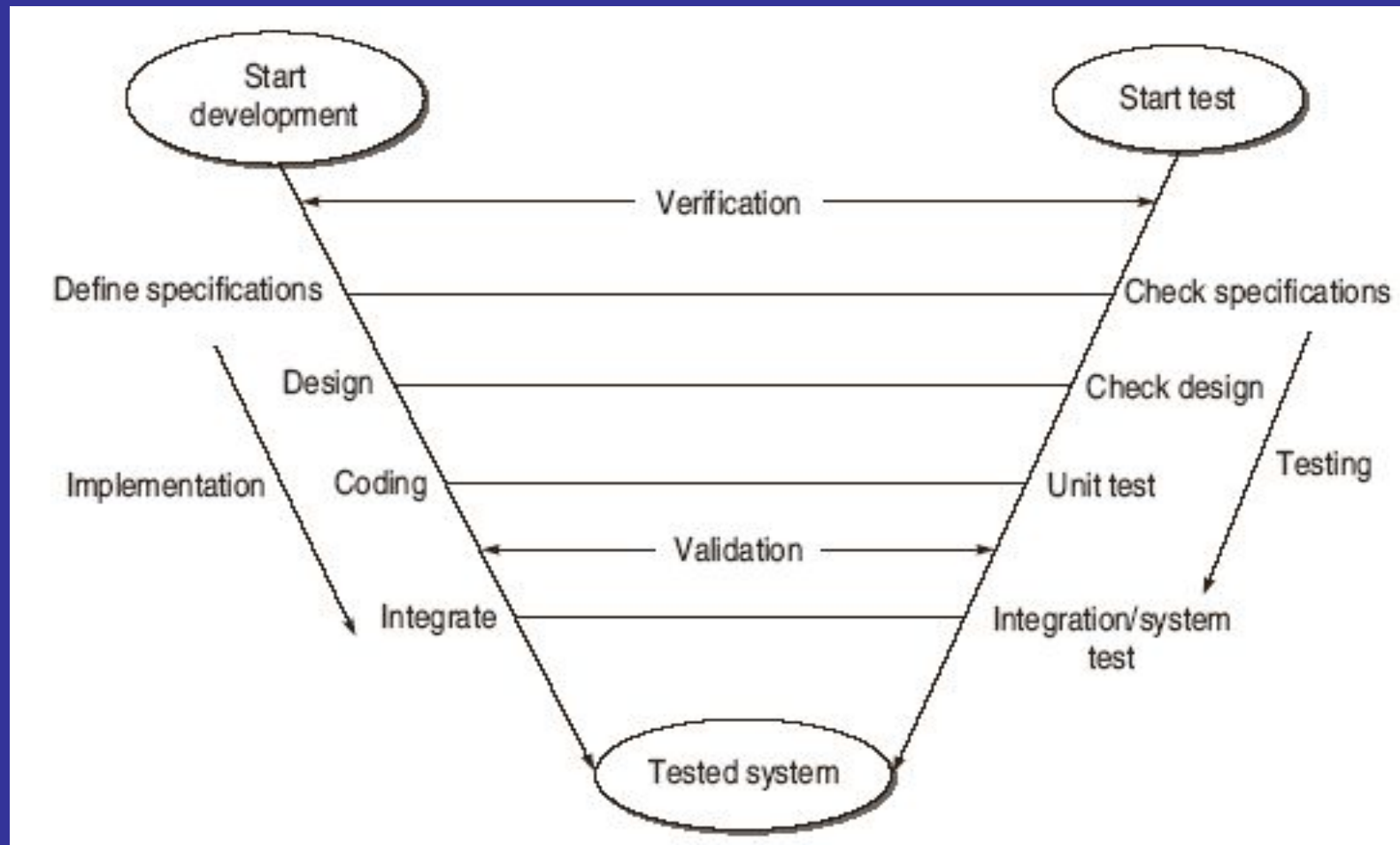
- **Select and Rank Test Factors**
- **Identify the System Development Phases**
- **Identify the Risks associated with System under Development**

Test Factors	Test Phase					
	Requirements	Design	Code	Unit test	Integration test	System test
Portability	Is portability feature mentioned in specifications according to different hardware?					Is system testing performed on MIPS and INTEL platforms?
Service Level	Is time frame for booting mentioned?	Is time frame incorporated in design of the module?				

Verification: “Are we building the product right?”

Validation: “Are we building the right product?”

V Testing Life Cycle Model



- Unit Testing
- Integration Testing
- Function Testing
- System Testing
- Acceptance Testing