Testing

Instructor: Mehroze Khan

Software Faults and Failures

Why Does Software Fail?

- Wrong requirement: not what the customer wants
- Missing requirement
- Requirement impossible to implement
- Faulty design
- Faulty code
- Improperly implemented design

Objective of Testing

- Objective of testing: discover faults
- A test is successful only when a fault is discovered
  - Fault identification is the process of determining what fault caused the failure
  - Fault correction is the process of making changes to the system so that the faults are removed

Elements of a Test Case

- Purpose
- Input
- Expected Output
- Actual Output
- Sample Format:

Test Case ID	What To Test?	How to Test?	Input Data	Expected Result	Actual Result	Pass/Fail
.	.	.	.	.	.	.

Types of Faults

- **Algorithmic fault**
  - When a component's algorithm or logic does not produce the proper output for a given input because something is wrong with the processing steps
  - These faults are sometimes easy to spot just by reading through the program (called **desk checking**) or by submitting input data from each of the different classes of data that we expect the program to receive during its regular working
  - Typical algorithmic faults include:
    - branching too soon
    - branching too late
    - testing for the wrong condition
    - forgetting to initialize variables or set loop invariants
    - forgetting to test for a particular condition (e.g., when division by zero might occur)
    - comparing variables of inappropriate data types

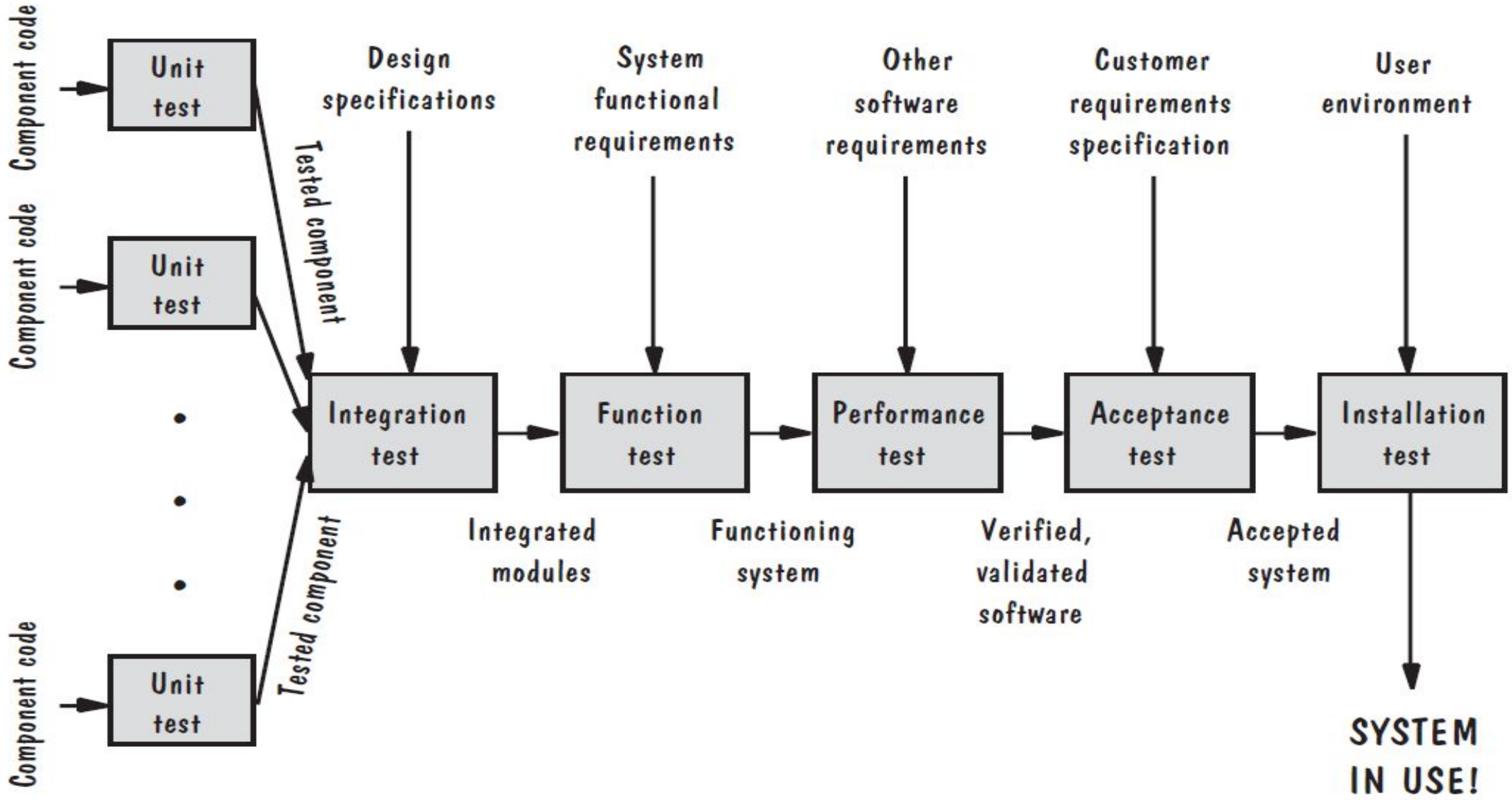
Types of Faults

- **Syntax fault**
- **Computation and precision fault**
  - A formula's implementation is wrong or does not compute result to required number of decimal places
- **Documentation fault**
  - Documentation doesn't match what program does
- **Capacity or boundary faults**
  - System's performance not acceptable when certain limits are reached
- **Timing or coordination faults**
  - Occur when the code coordinating several processes executing simultaneously or in a carefully defined sequence is inadequate
- **Throughput or performance faults**
  - System does not perform at the speed prescribed by requirements

Different Level of Failure Severity

- Catastrophic: causes death or system loss
- Critical: causes severe injury or major system damage
- Marginal: causes minor injury or minor system damage
- Minor: causes no injury or system damage

Testing Steps



Testing Issues

Attitude Toward Testing

- Programs are viewed as components of a larger system, not as the property of those who wrote them

Testing Issues

Who Performs the Test?

- Independent test team
  - avoid conflict
    - personal responsibility vs need to discover faults
  - improve objectivity
  - allow testing and coding concurrently

Testing Issues

Views of the Test Objects

- If we view the test object from the outside as a **closed box** or **black box** whose contents are unknown, our testing feeds input to the closed box and notes what output is produced
  - Test's goal is to be sure that every kind of input is submitted, and that the output observed matches the output expected
- **Open box** (sometimes called **clear box** or **white box**); can use the structure of the test object to test in different ways
  - Devise test cases that execute all the statements or all the control paths within the component(s) to be sure the test object is working properly