Introduction to Software Testing

Dr. Hoger Mahmud | 2024

AUIS IT DEPARTMENT

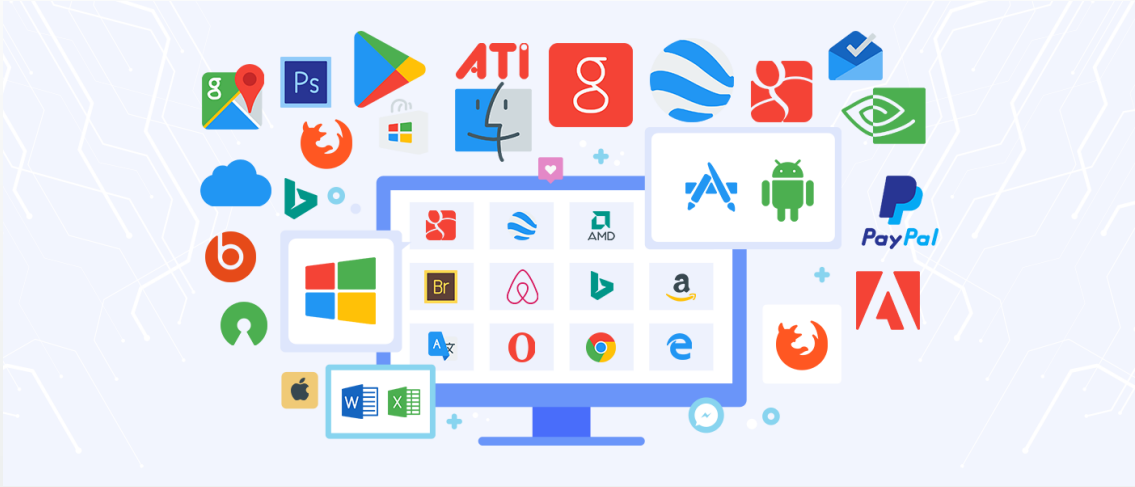
Content

- What is software and software testing
- Software failures
- Software verification and validation
- Why we test software
- Sources of software problems and cost of fixing

- https://www.youtube.com/watch?v=oLcggVM8FBM

Software is …

- Requirements specification documents
- Design documents
- Source code
- Test suites and test plans

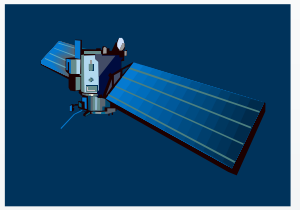


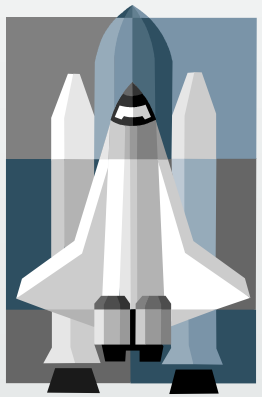
- Interfaces to hardware and software operating environment
- Internal and external documentation
- Executable programs and their persistent data

Testing in the 21st Century

- **Today's software market** :
  - is much bigger
  - is more competitive
  - has more users
- **Embedded Control Applications**
  - airplanes, air traffic control
  - spaceships
  - watches
  - remote controllers





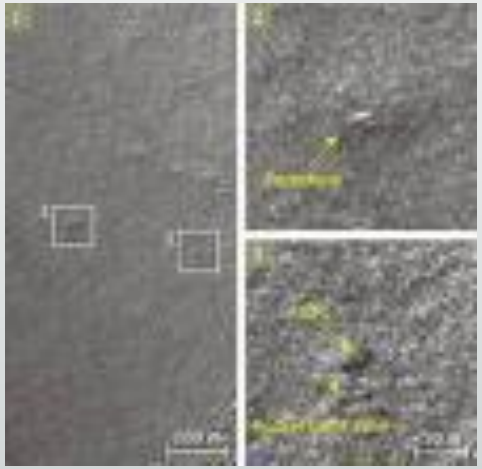

- **Agile processes put increased pressure on testers**
  - Programmers must unit test – with no training or education!
  - Tests are key to functional requirements – but who builds those tests ?

Ariane 5 – a spectacular failure

- 10 years and $7 billion to produce
- < 1 min to explode
- The error came from a piece of the software that was not needed during the crash
- Programmers thought that this particular value would never become large enough to cause trouble
- Removed the test present in Ariane 4 software
- 1 bug = 1 crash

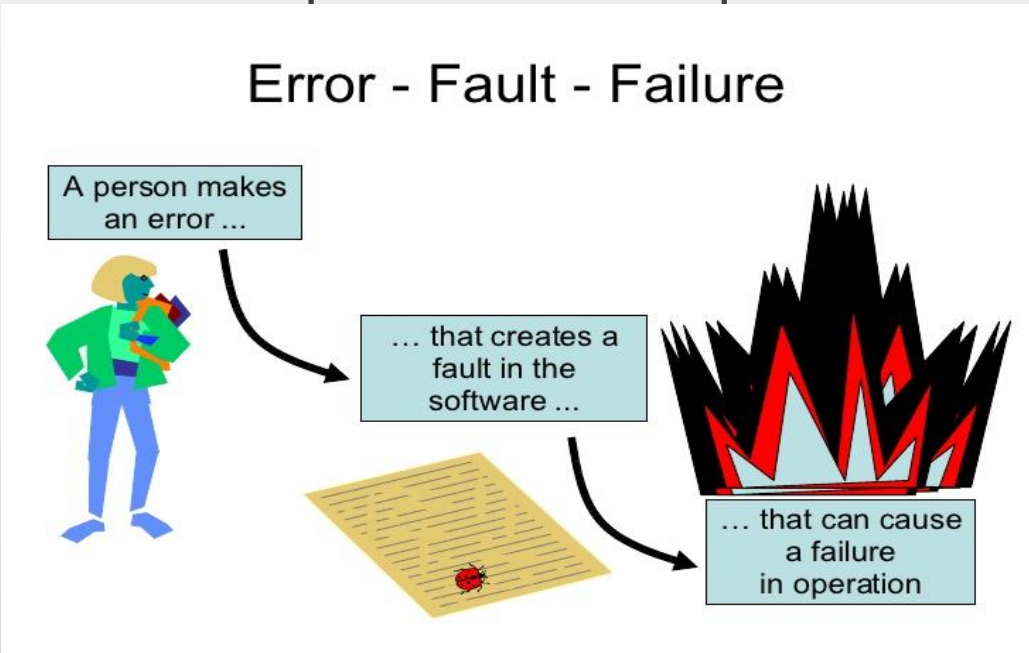
Other Software Failure Examples

- **Boeing A220** : Engines failed after software update allowed excessive vibrations
- **Healthcare website** : Crashed repeatedly on launch—never load tested
- **Toyota brakes** : Dozens dead, thousands of crashes
- **NASA's Mars lander:** September 1999, crashed due to a units integration fault



Software Faults, Errors & Failures

- **Software Error** : An incorrect internal state that is the manifestation of some fault
- **Software Fault** : A static defect in the software
- **Software Failure** : External, incorrect behavior with respect to the requirements or other description of the expected behavior.



A Concrete Example

```
public static int numZero (int [ ] arr)
{  // Effects: If arr is null throw NullPointerException
   // else return the number of occurrences of 0 in arr
   int count = 0;
   for (int i = 1; i < arr.length; i++)
   {
      if (arr [ i ] == 0)
      {
         count++;
      }
   }
   return count;
}
```

Fault: Should start searching at 0, not 1

Error: i is 1, not 0, on the first iteration
Failure: none

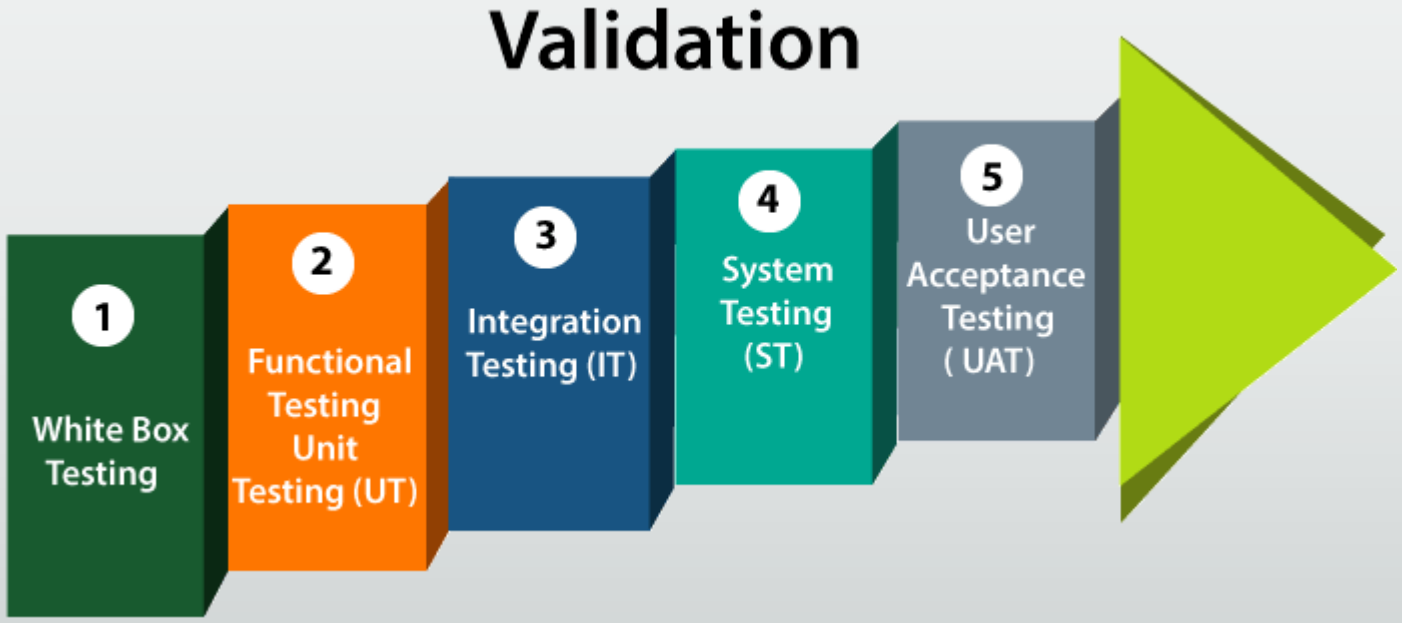
Error: i is 1, not 0
Error propagates to the variable count
Failure: count is 0 at the return statement

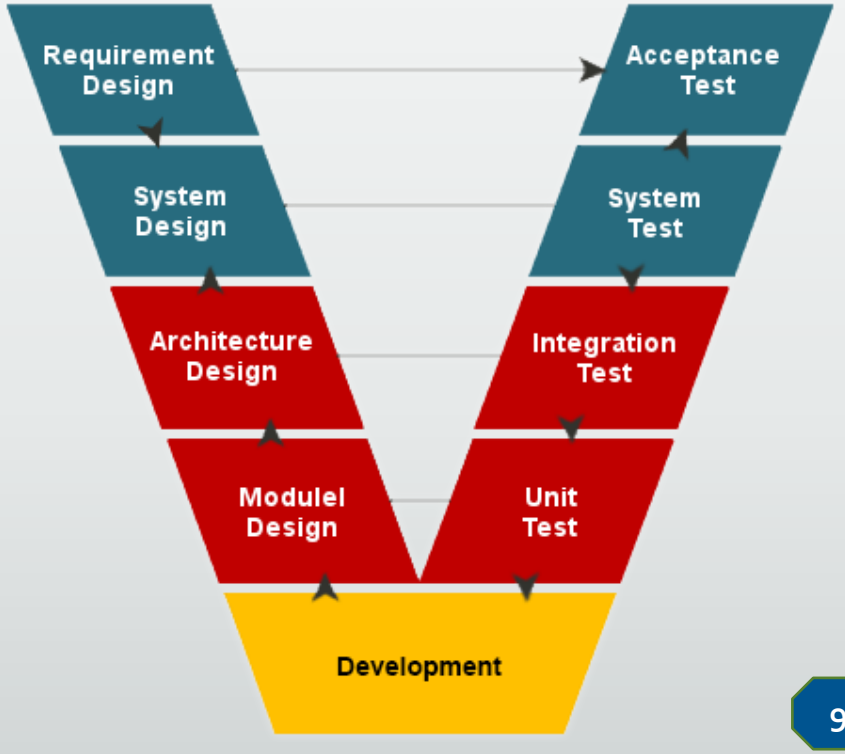
Test 1
[2, 7, 0]
Expected: 1
Actual: 1

Test 2
[0, 2, 7]
Expected: 1
Actual: 0

Validation & Verification (IEEE)

- **Validation** : The process of evaluating software at the end of software development to ensure compliance with intended usage
- **Verification** : The process of determining whether the products of a given phase of the software development process fulfill the requirements established during the previous phase





What is software testing?

IEEE defines software testing as:

- A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.
- **Note that…**

…one needs to know the required conditions

…one needs to be able to observe the existing conditions

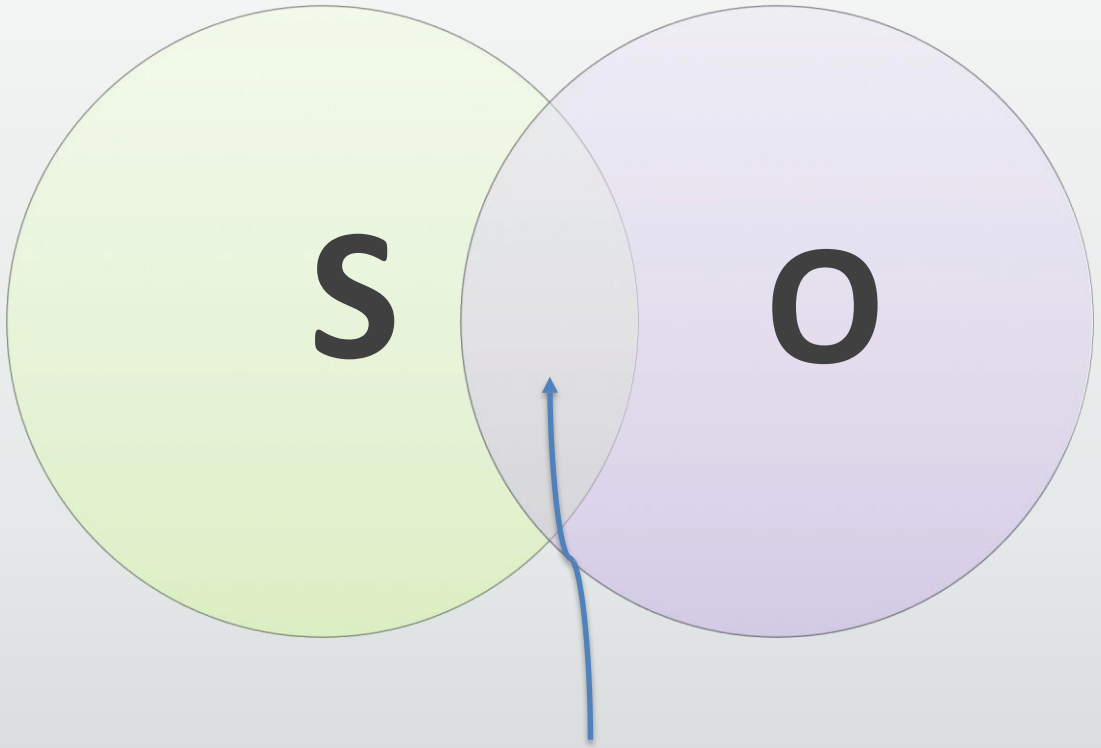
- **Testing focuses on behavioral (what the program does) and structural (how the program is) aspects**

Software Specification

- You have to know what your product is before you can say if it has a bug.
- A specification defines the product being created and includes:
- **Functional requirements** that describes the features the product will support. E.g., on a word processor: Save, print, check spelling, change font, …
- **Non-functional requirements** are constraints on the product. E.g, Security, reliability, user friendliness, platform, …

Program Behavior

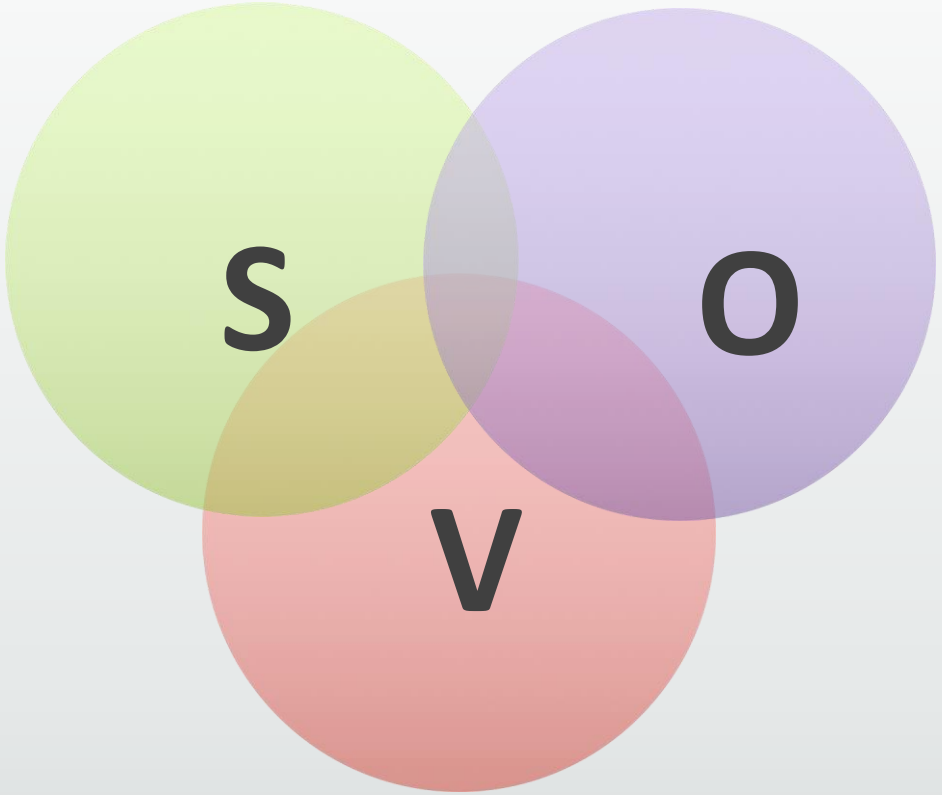
Specification
(expected behavior)



Program
(observed behavior)

Program Behavior and Testing

Specification
(expected behavior)



Program
(observed behavior)

Test Cased
(verified)

Why We Test Software?

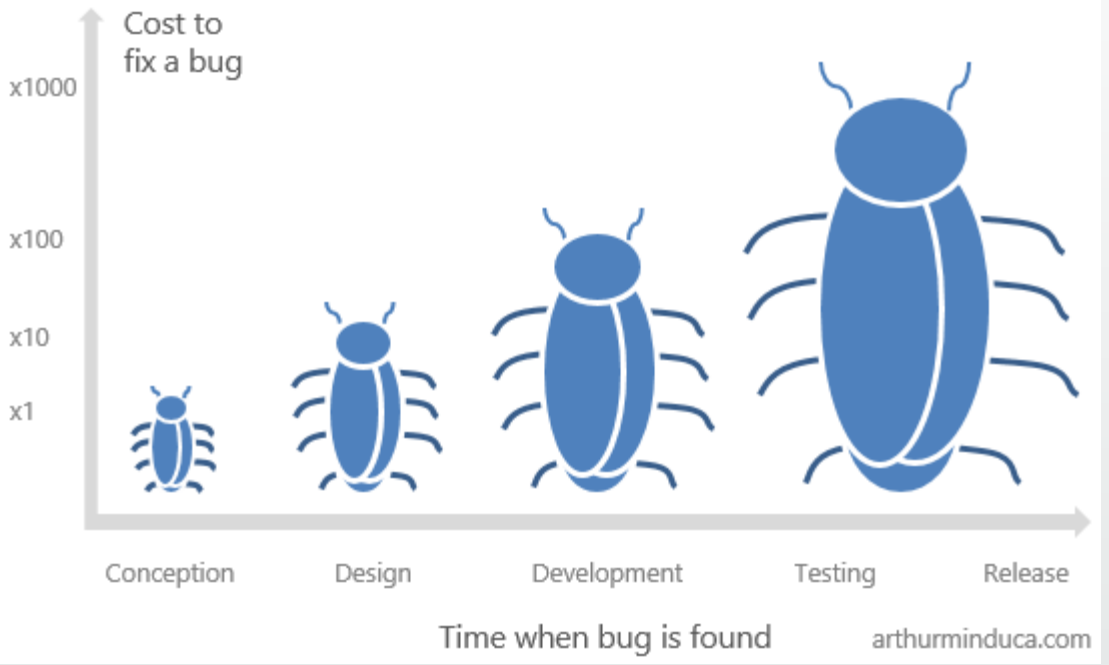
- To isolate and fix bugs in the program
- To demonstrate that the program works
- To demonstrate that the program doesn't work
- To reduce the risk involved in using the program
- To have a methodology for producing better quality software

How would you test a ballpoint pen?

- Does the pen write?
- Does it work upside down?
- Does it write in the correct color?
- Do the lines have the correct thickness?
- Does the click-mechanism work? Does it work after 100,000 clicks?
- Is it safe to chew on the pen?
- Is the logo on the pen according to company standards?
- Does the pen write in -40 degree temperature?
- Does the pen write underwater?
- Does the pen write after being run over by a car?
- **Which are relevant? Which are not relevant? Why (not)?**

Sources of Problems and Cost of Fixing

- **Requirements Definition**: Erroneous, incomplete, inconsistent requirements.
- **Design:** Fundamental design flaws in the software.
- **Implementation**: Mistakes in chip fabrication, wiring, programming faults, malicious code.
- **Support Systems:** Poor programming languages, faulty compilers and debuggers misleading development tools.
- **Inadequate Testing of Software**: Incomplete testing, poor verification, mistakes in debugging.



References

- As specified in the syllabus