Software Verification and Validation
SENG 22233

MISS K.THADSAYINI
BSc (Hons) in Sofware Engineering, UOK
Temporary Lecturer, UOK
Visiting Instructor, NAITA

Course Information

- SENG 22233
- Lecturer : Miss K.Thadsayini
- email : thkat201@kln.ac.lk
- Phone : 076 4348994

Thursdays 01:00 pm - 03:00 pm

Evaluation

- Attendance - 10%
- Assignments - 20%
- Final examination - 70%

Outline

- software validation and verification
- software inspections and reviews
- software testing principles
- software error proneness and testability
- specification-based testing, code-based testing, alpha, beta and acceptance testing, test prioritization, coverage criteria, test instrumentation
- software process and product quality, clean-room approach
- software quality standards
- software process certification

Software is everywhere. However, it’s written by people—so it’s not perfect

Famous Software Test Failures

Y2K (1999) - Cost: $500 billion Disaster:

Businesses spent billions on programmers to fix a glitch in legacy software. While no significant computer failures occurred, preparation for the Y2K bug had a significant cost and time impact on all industries that use computer technology.

Mars Climate Crasher (1998) - Cost: $125 million :

The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newton) as specified by NASA.

What is a ‘Defect’/ ‘Bug’ ?

What is a ‘Defect’/ ‘Bug’ ?

A human being can make an error (mistake), which produces a defect (fault, bug) in the program code or in a document. If a defect in code is executed, the system may fail to do what it should do (or do something it shouldn’t), causing a failure.

Defects in software, systems or documents may result in failures, but not all defects do so. Defects occur because human beings are fallible and because there is time pressure, complex code, complexity of infrastructure, changing technologies, and/or many system interactions

The bug had various names in different companies such as **error, issues, problem, fault, and mistake, etc**

1. **Defect** - The variation between the actual results and expected results

2. **Error** - If a developer unable to successfully compile or run a program

3. **Failure** - After release, if an end user finds an issue

4. **Wrong:** When requirements are implemented not in the right way. This defect is a variance from the given specification.

5. **Missing:** A requirement of the customer that was not fulfilled.

6. **Extra:** A requirement incorporated into the product that was not given by the end customer.

7. **Bug**: A bug is the result of a coding error. An Error found in the development environment before the product is shipped to the customer.

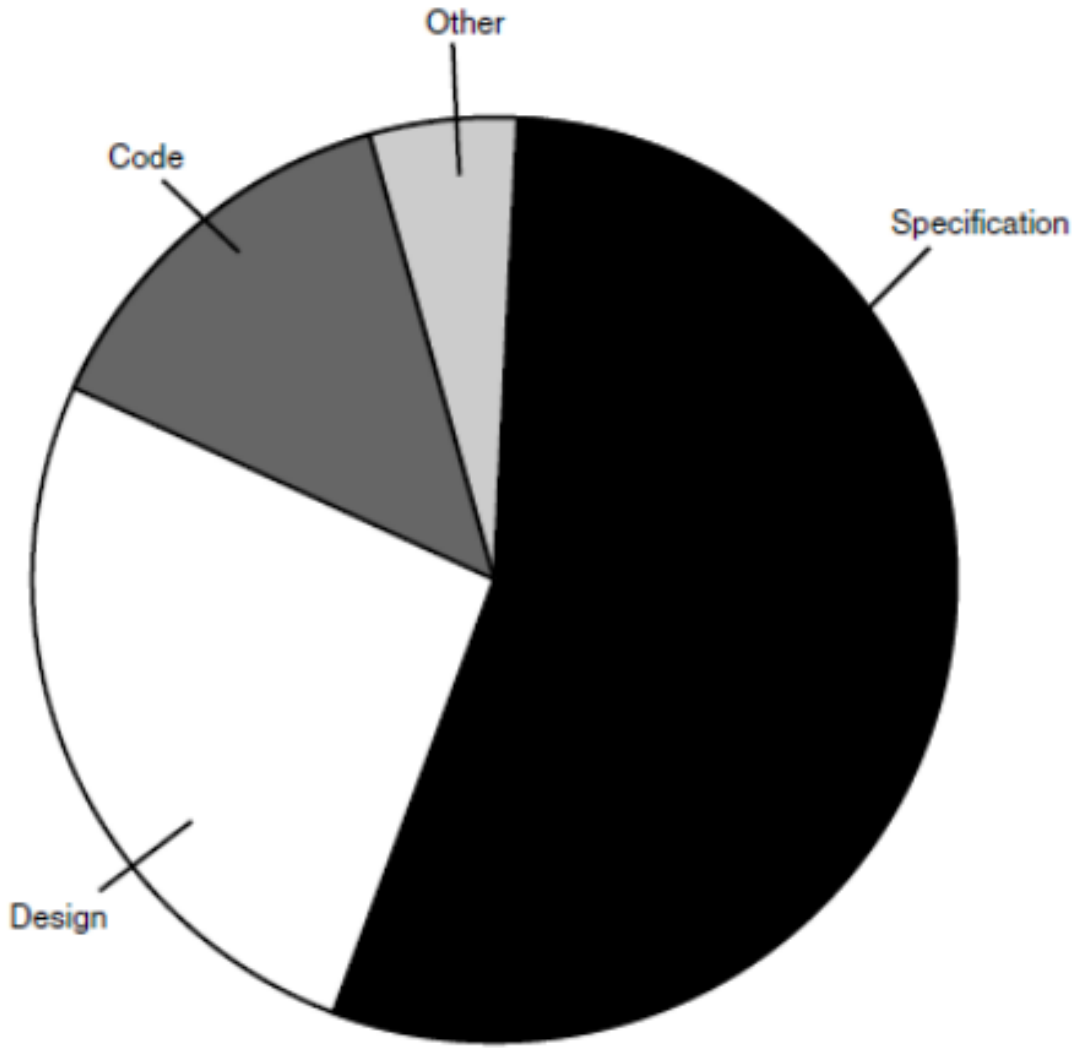
8. **Fault:** An incorrect step, process or data definition in a computer program which causes the program to perform in an unintended or unanticipated manner

Software Bug: A Formal Definition

A software bug occurs when one or more of the following five rules is true:

1. The software doesn't do something that the product specification says it should do.
2. The software does something that the product specification says it shouldn't do.
3. The software does something that the product specification doesn't mention.
4. The software doesn't do something that the product specification doesn't mention but should.
5. The software is difficult to understand, hard to use, slow, or—in the software tester's eyes—will be viewed by the end user as just plain not right

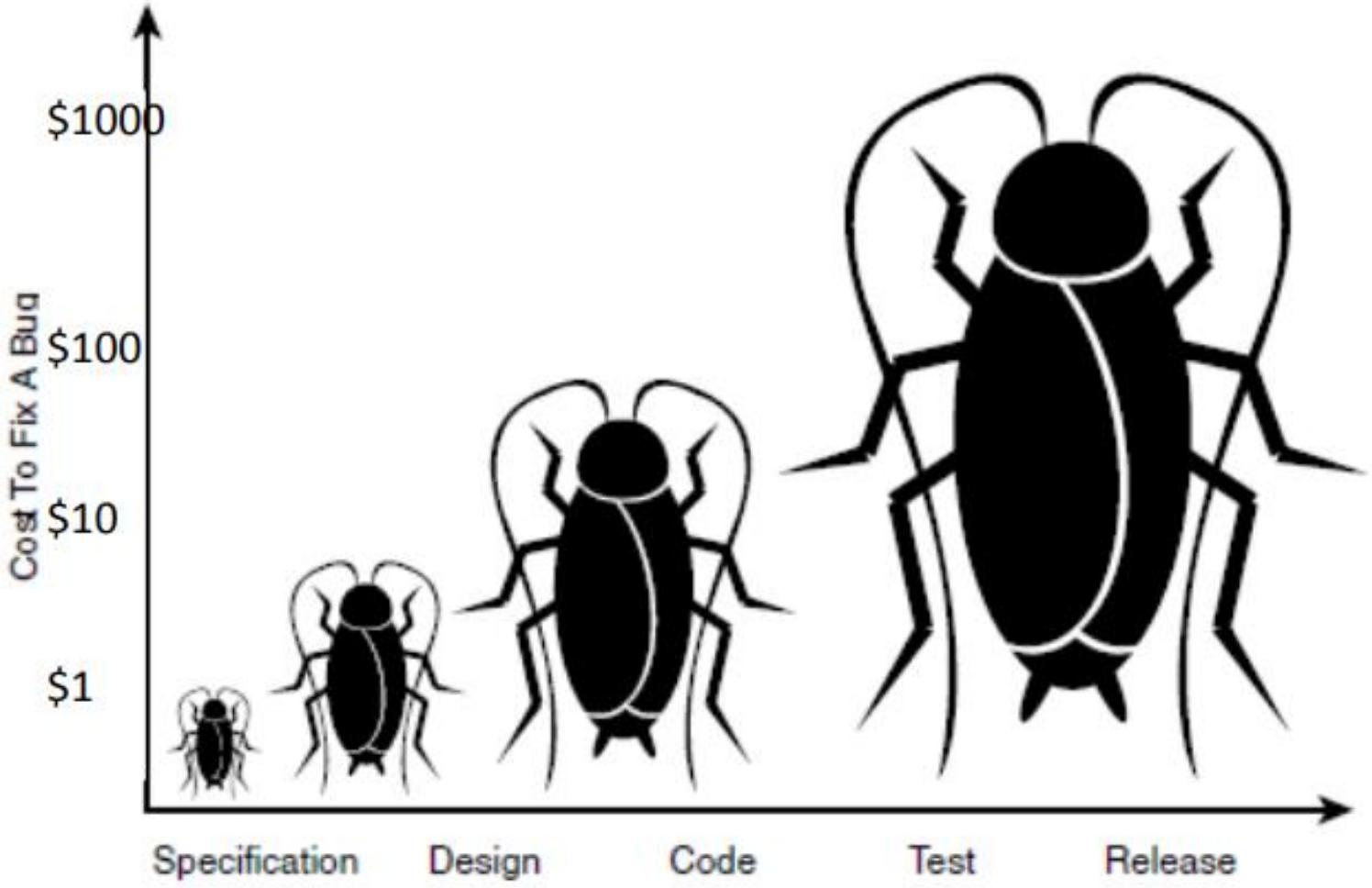
Why Do Bugs Occur?



Why Do Bugs Occur?

- Programming Error
- Incomplete Requirements
- Changes in the Requirements
- Poor Design
- Miscommunication
- Environment Factors

The Cost of Bugs



What Exactly Does a Software Tester Do?

The goal of a **software tester** is to,

- Find bugs.
- Find them as early as possible.
- Make sure they get fixed.

Definition of Testing

- IEEE 610 (Software Engineering Terminology):
  "The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component."

- IEEE 829 (Test Documentation):
  "The process of analyzing a software item to detect the difference between existing and required conditions (that is, bugs) and to evaluate the features of the software items."

Definition of Testing

- BS 7925-1 (Software Testing–Vocabulary):
  "Process of exercising software to verify that it satisfies requirements and to detect errors."

- ISTQB Glossary of Terms used in Software Testing V 1.0:
  "The process consisting of all life cycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects."

Qualities of Testers

Qualities of Testers

1. **They are explorers.** Software testers aren't afraid to venture into unknown situations. They love to get a new piece of software, install it on their PC, and see what happens.
2. **They are troubleshooters.** Software testers are good at figuring out why something doesn't work. They love puzzles.
3. **They are relentless.** Software testers keep trying. They may see a bug that quickly vanishes or is difficult to re-create. Rather than dismiss it as a fluke, they will try every way possible to find it.

Qualities of Testers

4. **They are creative.** Testing the obvious isn't sufficient for software testers. Their job is to think up creative and even off the-wall approaches to find bugs.

5. **They exercise good judgment.** Software testers need to make decisions about what they will test, how long it will take, and if the problem they're looking at is really a bug.

Qualities of Testers

6. **They are tactful and diplomatic.** Software testers are always the bearers of bad news.

7. **They are persuasive.** Bugs that testers find won't always be viewed as severe enough to be fixed. Testers need to be good at making their points clear, demonstrating why the bug does indeed need to be fixed, and following through on making it happen

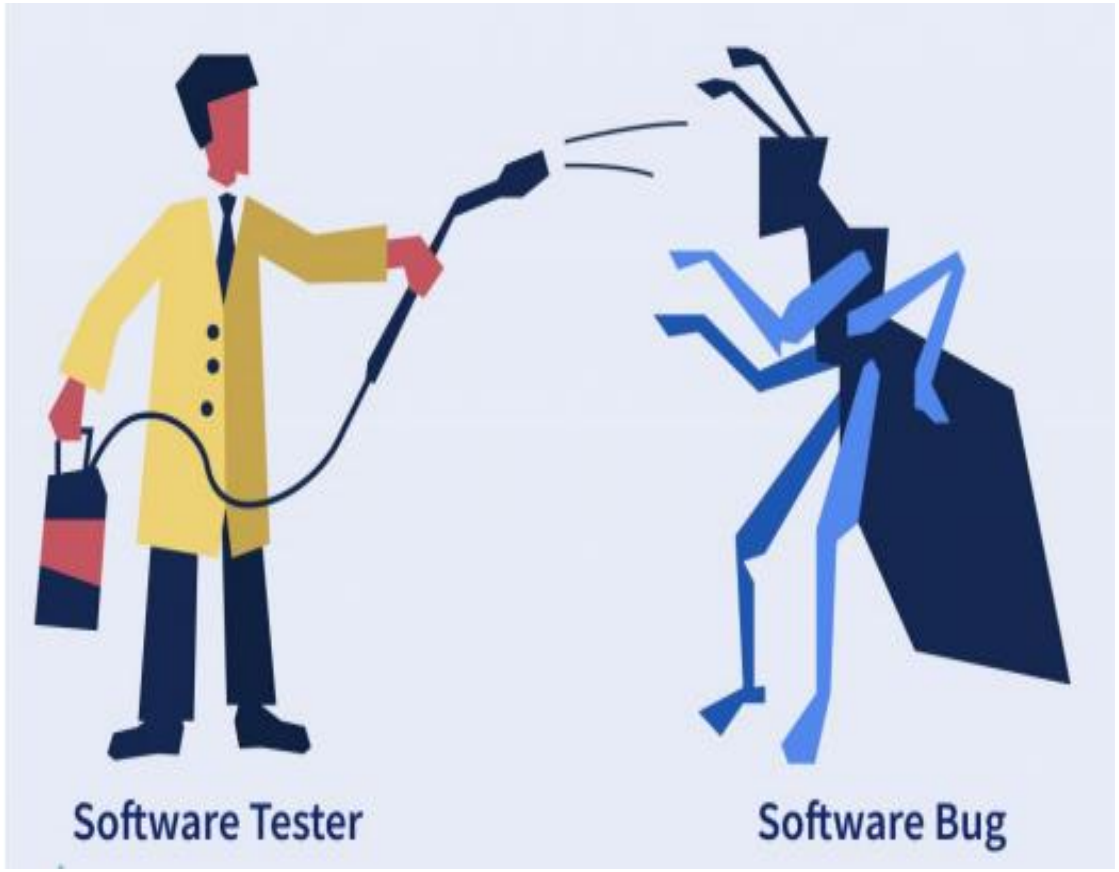
Impossible to Test a Program Completely

- The number of possible inputs is very large.
- The number of possible outputs is very large.
- The number of paths through the software is very large.

The More Bugs You Find, the More Bugs There Are

- Programmers have bad days
- Programmers often make the same mistake
- Some bugs are really just the tip of the iceberg

The Pesticide Paradox



If the same kinds of tests are repeated again and again, eventually the same set of test cases will no longer be able to find any new defects. To overcome this "pesticide paradox", test cases must be regularly reviewed and revised and new and different tests need to be written to cover different paths in the software or system to enable the discovery of new defects. (ISTQB)

Not All the Bugs You Find Will Be Fixed

- There’s not enough time
- It’s really not a bug
- It’s too risky to fix
- It’s just not worth it

Software Testing Terms and Definitions

Precision and Accuracy

Verification and Validation

Quality and Reliability

Testing and Quality Assurance

THANK YOU