DA-IICT

IT 314: Software Engineering

Software Testing: Introduction

Saurabh Tiwari

Spectacular Software Failures

- MARINER 1, the first U.S. attempt to send a spacecraft to Venus, failed minutes after launch in 1962. The guidance instructions from the ground stopped reaching the rocket due to a problem with its antenna, so the onboard computer took control. However, there turned out to be a bug in the guidance software, and the rocket promptly went off course, so the Range Safety Officer destroyed it. Although the bug is sometimes claimed to have been an incorrect FORTRAN DO statement, it was actually a transcription error in which the bar (indicating smoothing) was omitted from the expression "R-dot-bar sub n" (nth smoothed value of derivative of radius). This error led the software to treat normal minor variations of velocity as if they were serious, leading to incorrect compensation.

- Software problems in the automated baggage sorting system of a major airport in February 2008 prevented thousands of passengers from checking baggage for their flights. It was reported that the breakdown occurred during a software upgrade, despite pre-testing of the software. The system continued to have problems in subsequent months.

Spectacular Software Failures

- AT&T long distance network crash (January 15, 1990), in which the failure of one switching system would cause a message to be sent to nearby switching units to tell them that there was a problem. Unfortunately, the arrival of that message would cause those other systems to fail too – resulting in a 'wave' of failure that rapidly spread across the entire AT&T long distance network. Wrong BREAK statement in C-Code
- The Northeast blackout of 2003 was a widespread power outage that occurred throughout parts of the Northeastern and Midwestern United States and Ontario, Canada on Thursday, August 14, 2003, just before 4:10 p.m. The blackout affected an estimated 10 million people in Ontario and 45 million people in eight U.S. states. A software bug known as a race condition existed in General Electric Energy's Unix-based XA/21 energy management system.

Spectacular Software Failures

- The European Space Agency's Ariane 5 Flight 501 was destroyed 40 seconds after takeoff (June 4, 1996). The US$1 billion prototype rocket self-destructed due to a bug in the on-board guidance software.
- NASA Mars Polar Lander was destroyed because its flight software mistook vibrations due to atmospheric turbulence for evidence that the vehicle had landed and shut off the engines 40 meters from the Martian surface (December 3, 1999). Its sister spacecraft Mars Climate Orbiter was also destroyed, but due to human error and not, as is sometimes reported, due to a software bug.

Software bugs can potentially cause monetary and even loss of life

Today's Software Market

- ☐ is much bigger
- ☐ is more competitive
- ☐ more users

Android Market

Open Source Software

Emerging Software!!
Example…
- APPS
- API's
- Games
- Security
- Safety

Embedded Systems

Challenges

How to test??

Most Costly Software Failures…

2002 : NIST report, "The Economic Impacts of Inadequate Infrastructure for Software Testing" costs the US alone between $22 and $59 billion annually

Huge losses due to web application failures
- Financial services $6.5 million per hour (in USA)
- Credit card sales applications: $2.4 million per hour (in USA)

London stock exchange shut down
- Sept 08 - 7 hours
- 25 Feb 11 about 4 hours

Challenges

Industries are going through a revolution in what testing means to the success of software products

What Does This Mean?

Software testing is getting more important

Testing in the 21st Century

- More safety critical, real-time software
- Embedded software is ubiquitous … check your pockets
- Enterprise applications means bigger programs, more users
- Paradoxically, free software increases our expectations !
- Security is now all about software faults
  - Secure software is reliable software
- The web offers a new deployment platform / APPS too…
  - Very competitive and very available to more users
  - Web apps are distributed
  - Web apps must be highly reliable

Industry desperately needs our inventions !

Why Testing?

- Uncover as many as errors as possible in a given timeline.
- Demonstrate a given software product matching its requirements specification.
- Validate the quality of a software.

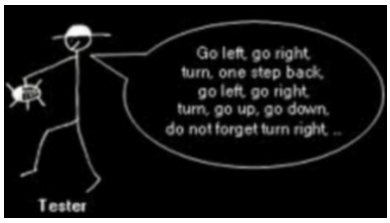
Software testing is arguably the least understood part of the development process. Through a four-phase approach, the author shows why eliminating bugs is tricky and why testing is a constant trade-off [Whittaker]

Who Tests the Software?

Developer

Understands the system but, will test "gently“ and, is driven by "delivery”

- Unit Testing (Unit level)
- Integration (Unit1, 2, 3....)



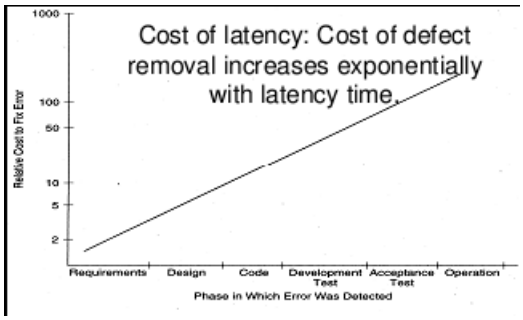
Independent Tester

Must learn about the system, but, will attempt to break it and, is driven by quality

- System level (System Testing)

Testing Challenges

- When to start testing? (cost of late testing)
- When to stop testing? How much testing is enough?
- Which and how many testing techniques/types?
- Test automation
- How will you know the program passed or failed test? (test output evaluation)



Testing Purpose

1. Avoid Redundancy
   - Source Code
   - Requirement/Design/Maintenance/Change Management

2. Reducing Cost
   - Cost of detection, Cost of Prevention
   - Internal Failure (at the time of development or before you release the software product)
   - External Failure (finding bugs after deploying the software product)

3. Correctness CONFIDENCE

4. Quality Assurance

5. Verification (design models, requirements...)

Suggested books

- Main text:
  "Software Testing and Quality Assurance: Theory and Practice", Kshirasagar Naik And Priyadarshi Tripathy, A John wiley & Sons, Inc., Publication
- *"Software Quality Assurance: From theory to implementation"*, Daniel Galin, Pearson Education Limited 2004
- *"Software Quality Engineering Testing, Quality Assurance, and Quantifiable Improvement"*, Jeff Tian, IEEE Computer Society
- *"The Art of Software Testing"*, G.J. Myers, Second Edition, John Wiley & Sons, New York, 1976.
- *"Lessons Learned in Software Testing"*, C Kaner, J. Bach, B. Pettichord Wiley, 2001.

Taxonomy of Bugs

Verification?

Error?

Validation?

Defects ? Doesn't work as defined

Fault?

Failure?- Non functioning of system

Testing?

Bugs? – error in the system

Debugging?

Quality?

Error, Defects, Fault and Failure

Example:
You are driving a car and you are on road while on driving now there is two way on the road

1. left--> Mumbai
2. right--> Delhi

Now you have to go to Delhi, it means you have to turn the steering to the right, but by mistake you turn the steering to the left, from that position that is called as "Error"

and now Fault is there till you will not reach the Mumbai, but when you reach Mumbai that is a final stage which is called "Failure" because you had to reach Delhi but now you are in Mumbai.

"A mistake in coding is called Error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it Is failure."

Coding: Error, Defects, Fault and Failure

The program is required to add two numbers

```
1   #include<stdio.h>
2
3   int main ()
4   {
5   int value1, value2, ans;
6
7   value1 = 5;
8   value2 = 3;
9
10  ans = value1 - value2;
11
12  printf("The addition of 5 + 3 = %d.", ans);
13
14  return 0;
15  }
```

5 + 3 should be 8, but the result is 2. There could be various reasons as to why the program displays the answer 2 instead of 8. For now we have detected a *failure*.

As the failure has been detected a *defect* can be raised.

The program is required to add two numbers

```
1   #include<stdio.h>
2
3   int main ()
4   {
5   int value1, value2, ans;
6
7   value1 = 5;
8   value2 = 3;
9
10  ans = value1 - value2;  // Bug, Defect
11
12  printf("The addition of 5 + 3 = %d.", ans);
13
14  return 0;
15  }
```

Now lets go back to the program and analyze what was the fault in the program.

There is a '-' sign present instead of '+' sign. So the fault in the program is the '-' sign.

Error is the mistake I made by typing '-' instead of '+' sign.

Error, Defects, Fault and Failure

A tester does not necessarily have access to the code and may be just testing the functionality of the program. In that case the tester will realize the output is faulty and will raise a defect.

- Error: A mistake made by a programmer
  - Example: Misunderstood the requirements.
- Defect/fault/bug: Manifestation of an error in a program.

Example:	
Incorrect code:	*if (a<b) {foo(a,b);}*
Correct code:	*if (a>b) {foo(a,b);}*

- Failure: Manifestation of one or more faults in the observed program behavior

Testing & Debugging

Testing: Finding inputs that cause the software to fail
- Finding and localization of a defect
- Done by Testing team
- Intention behind is to find as many as defects possible

Debugging: The process of finding a fault given a failure
- Fixing that defect
- Done by development team
- Intention is to remove those defects

"...testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence. The only effective way to raise the confidence level of a program significantly is to give a convincing proof of its correctness."

http://www.cs.utexas.edu/users/EWD/transcriptions/EWD03xx/EWD340.html Edsger Dijkstra

Trivial example

- Try to move a file from folder A to another folder B
- What are the possible scenarios?

Possible Solutions

- Trying to move the file when it is open
- You do not have the security rights to paste the file in folder B
- Folder B is on a shared drive and storage capacity is full
- Folder B already has a file with the same name

Another Scenario

- Suppose you have a 15 input fields each one having 5 possible values.
- How many combinations to be tested?

5^15 = 30517578125!!!

Software Testing is a process, or a series of processes, designed to make sure computer code does what it is designed to do and that it does not do anything unintended.

A Self Assessment Test

Write a set of test cases – specific set of data – to properly test a relatively Simple program.

Create a set of test data for the program .

The program reads three integer values from an input dialog. The three value represent the lengths of the sides of a triangle. The program displays a message that states whether the triangle is scalene, isosceles, or equilateral.

Test Action and Data	Expected Result
2, 5, 10	??
…	…

A Self Assessment Test

The function triangle takes three integer parameters that are interpreted as the lengths of the sides of a triangle. It returns whether the triangle is equilateral (three lengths equal), isosceles (two lengths equal), scalene (no lengths equal), or invalid (impossible lengths).

```
final int EQUILATERAL = 0;
final int ISOSCELES = 1;
final int SCALENE = 2;
final int INVALID = 3;
int triangle(int a, int b, int c)
{
   if (a >= b+c || b >= a+c || c >= a+b)
      return(INVALID);
   if (a == b && b == c)
      return(EQUILATERAL);
   if (a == b || a == c || b == c)
      return(ISOSCELES);
   return(SCALENE);
}
```

Solution?

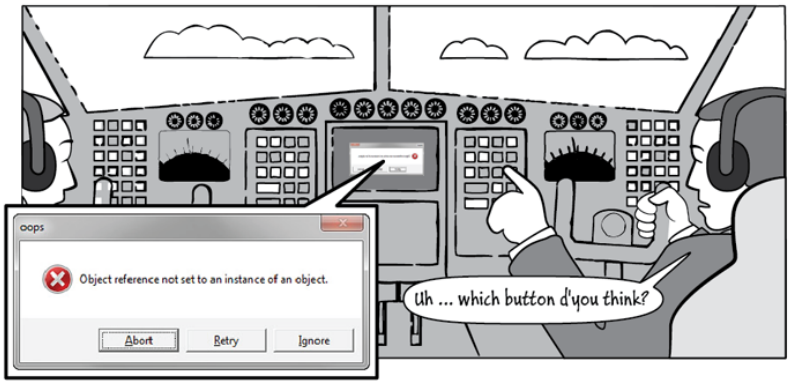
Tester Action and Data	Expected Result
Erroneous input partition	
a, b, or c a non-number	an error message
Scalene triangle partition	
2, 3, 4	scalene
1, 2, 3	an error message
Isosceles triangle partition	
2, 2, 1	isosceles
2, 2, 0	an error message
Equilateral triangle partition	
1, 1, 1	equilateral
1, 1, -1	an error message
User interface tests	
Try a leading space for a, b, c	an error message

Many more…

Summary: Why Do We Test Software ?

A tester's goal is to eliminate faults as early as possible

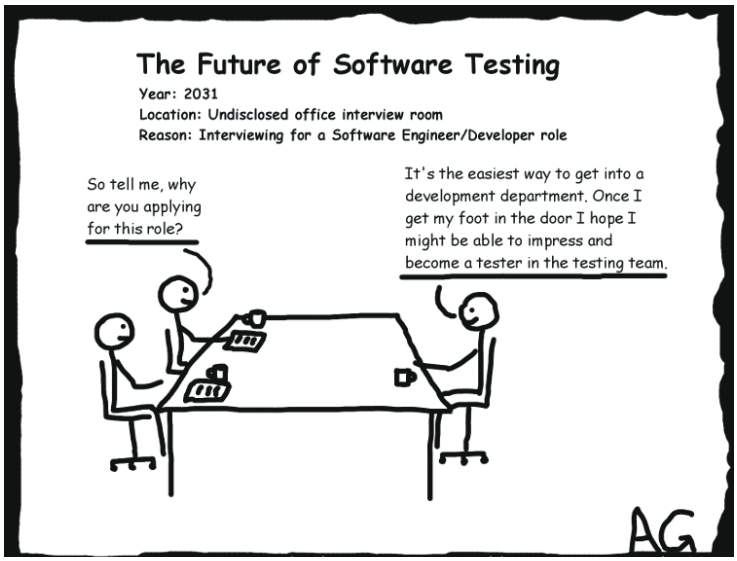
- Improve quality
- Reduce cost
- Preserve customer satisfaction



Source: Microsoft Academia

Software should be predictable and consistent, offerings no surprises to users.

Software should be predictable and consistent, offerings no surprises to users.



The Mind of a TESTER

Four different kinds of thinking exhibited by a GOOD TESTER *[Kaner, Bach, Pettichord]*

1. Technical Thinking: *the ability to model technology and understand causes and effects*
2. Creative Thinking: *the ability to generate ideas and see possibilities*
3. Critical Thinking: *the ability to evaluate ideas and make inferences*
4. Practical Thinking: *the ability to put ideas into practice*

Example to TEST YOUR Mind

An example of these kinds of thinking is found in a fable called "The King's Challenge"

The King's Challenge (a fable)

Once upon a time, a mighty king wanted to determine which of his three court wizards was the most powerful.

So,

He put the three court wizards in the castle dungeon and declared whoever escaped from his respective dungeon first was the most powerful wizard in all the kingdom.

(If you are one of three wizard, what you would do.)

Example to TEST YOUR Mind…

The first wizard immediately started chanting musical poems to open his cell door.

The second wizard immediately started casting small polished stones and bits of bone on the floor to learn how he might open his cell door

The third wizard sat down across from his cell door and thought about the situation for a minute. Then he got up, walked over to the cell door and pulled on the door handle. The cell door swung open because it was closed but not locked.

Thus, the third wizard escaped his cell first and became known as the most powerful wizard in all the kingdom.

Example to TEST YOUR Mind…

What kind of "tester" thinking did the third wizard exercise in solving the king's puzzle?

- Creative thinking: *the ability to see the possibility that the door was not locked in the first place.*
- Practical thinking: *the ability to decide to try the simplest solution first.*

Another Example to TEST YOUR Mind…

BUYING A CAR

-Go for a test drive & What you are supposed to do?
- Take test drive to BREAK THE CAR
OR
- To improve the CAR's DESIGN

Objectives of TEST DRIVE are:

- To validate affordability
- To validate attractiveness
- To validate comfort
- To validate usefulness
- To validate performance

When you have clear testing objectives, the we choose approaches that best validates the car against those Objectives.

Testing Approaches include:

- *Examine the sticker price and sale contract*
- *Trying out the radio, the air conditioner and the lights*
- *Trying acceleration, stopping and cornering*

These testing approaches are referred to by fairly common terminology in the testing industry.

How??

Examine = Static Testing
(observe, read, review without actually driving the car)

Try out = Functional and structural testing
(work different features of the car without actually driving the car)

Try = Performance Testing
(work different features of the car by actually driving the car)

The Psychology of Testing

One of the primary causes of POOR program testing is – most of the programmers begin with a FALSE definition of the term:

Such as,

- *"Testing is the process of demonstrating that error are not present"*
- *"The purpose of testing is to show that a program performs its intended functions correctly"*
- *"Testing is the process of establishing confidence that a program does what it is supposed to do"*

Testing is the process of executing a program with the intent of finding errors.

Questions??