

Testing Concepts

Lesson 1: Fundamentals of
Testing

Lesson Objectives

- To understand the following topics:
 - Some Facts
 - Introduction to Software Testing
 - Software Testing - Definitions
 - Need of Software Testing
 - Error-Failure-Defect
 - Causes of Software Defects
 - Cost of Software Defects
 - What does Software Testing reveal
 - Importance of Software Testing
 - Importance of Testing Early in SDLC Phases
 - Testing and Quality
 - Quality Perceptions
 - Seven Testing Principles
 - Economics of Testing
 - How Testing is conducted?




Lesson Objectives

- To understand the following topics:
 - Software Testing – Then (Past)
 - Software Testing – Now (Present)
 - Scope of Software Testing
 - Factors influencing the Scope of Testing
 - Risk Based Testing
 - Project Risks
 - Product Risks
 - Need of Independent Testing
 - Activities in Fundamental Test Process
 - Attributes of a good Tester
 - Psychology of Testing
 - Code of Ethics for Tester
 - FS SBU: Focus on Testing
 - Testing Roles in iTEAMS
 - Limitations of Software Testing



Some Facts !!

Ariane 5 explosion - data conversion of a 64-bit no to 16-bit	Patriot missile - rounding error....Kills 28, injures 100
Mars Climate Orbiter lost - (Mixture of pounds and kilograms, 1999)	F16 autopilot - flipped plane upside down whenever it crossed the equator
Microsoft's anti-Unix site crashes News: vnunet.com	Yahoo glitch strikes again 03/20/2002


Copyright © Capgemini 2015. All Rights Reserved 4

Some Facts

Ariane 5 explosion - data conversion of a 64-bit no to 16-bit

On 4 June 1996, the maiden flight of the Ariane 5 launcher ended in a failure.

Only about 40 seconds after initiation of the flight sequence, at an altitude of about 3700 m, the launcher veered off its flight path, broke up and exploded.

The internal SRI* software exception was caused during execution of a data conversion from 64-bit floating point to 16-bit signed integer value. The floating point number which was converted had a value greater than what could be represented by a 16-bit signed integer

Patriot missile - rounding error, kills 28, injures 100

A report of the General Accounting office, GAO/IMTEC-92-26, entitled Patriot Missile Defense: Software Problem Led to System Failure at Dhahran,

Saudi Arabia reported on the cause of the failure. It turns out that the cause was an inaccurate calculation of the time since boot due to computer arithmetic errors.

Mars Climate Orbiter lost - (Mixture of pounds and kilograms, 1999)

The peer review preliminary findings indicate that one team used English units

(e.g. inches, feet and pounds) while the other used metric units for a key spacecraft operation


F16 autopilot - flipped plane upside down whenever it crossed the equator

Some Facts !!

Excel gives $77.1 \times 850 = 100000$ instead of 65535

Y2K problem in Payroll systems designed in 1974

Disney's Lion King - 'Simba'

 CONSULTING TECHNOLOGY SERVICES

Copyright © Capgemini 2015. All Rights Reserved 5

Microsoft's anti-Unix site crashes - News: vnunet.com

The Web site launched by Microsoft and Unisys to lure customers away from

Unix has turned into a major embarrassment in more ways than one. First it was revealed that the site was powered by an open-source version of Unix and was running on the Apache Web server. So Microsoft switched to its Internet Information Server software and the site crashed.

Yahoo glitch strikes again - 03/20/2002

Users report irregular and missing content Parts of Yahoo were shut down on

Tuesday following software problems encountered in the merging of Yahoo Groups and Yahoo Clubs.

Excel gives $77.1 \times 850 = 100000$ instead of 65535

Excel gives $77.1 \times 850 = 100000$ instead of 65535 : While multiplying two numbers in MS-Excel and if the product equals 65535, it always gives the result 100000.

Y2K problem in Payroll systems designed in 1974

In 1974, when the first payroll was developed, to minimize the utilization of memory space, the year of the dates were stored in two digits instead of four digits i.e. 00 instead of 1900. But after 25 years in the yr. 2000, the question arose that how to store this. Will it be considered 1900 or 2000?



9. Disney's Lion King – Simba
- At the fall of 1994 Christmas, the Disney company came up with its first venture – a CDROM game for children with animated story of 'The Lion King Simba'. The sale was very huge. However it was a great loss to the Disney – it failed to work on most of the systems available in the market. The very next day on December 26th, Disney's customer care phones began to ring with calls from angry parents and crying children. Disney failed to properly test the game software on different PC models available in the market and as a result, the software worked only on few systems that were just like the one used by Disney programmers.

Introduction to Software Testing

- Software Testing is the process of executing a program with the intent of finding errors as early as possible in SDLC
- It is a process used to help identify the correctness, completeness and quality of a developed computer software
- Software Testing helps in Verifying and Validating if the Software is working as it is intended to be working
- Testing is a process that helps in finding out how well the product works :
 - Aimed at finding defects
 - Aimed at demonstrating lack of quality
 - Aimed at demonstrating the gap between specifications and actual product
 - Aimed at building faith in the end product that gives advice on quality and risk



Copyright © Capgemini 2015. All Rights Reserved 7

Introduction to Software Testing?

Exercising (analyzing) a system or component with :
defined inputs
capturing monitored outputs
comparing outputs with specified or intended Requirements

To maximize the number of errors found by a finite no of test cases.
Testing is successful if you can prove that the product does what it should not do and does not do what it should do.

Quality: The degree to which a component system or process meets specified requirements and/or user/customer needs and expectation.

Software Testing - Definitions

- The process of executing a program (or part of a program) with the intention of finding errors - G. J. Myers
- Software testing is the process of testing the functionality and correctness of software by running it
- Testing is the process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements - IEEE 83a
- The process of analyzing a system to detect the difference between existing and required conditions and to evaluate the feature of the system - IEEE/ANSI, 1983 [Std 829-1983]

Need of Software Testing

- To find greatest possible number of errors with manageable amount of efforts applied over a realistic time span with a finite number of test cases



Cost effective

Time limited



- Because software is likely to have faults
- Because failures can be very expensive
- To contribute to the delivery of higher quality software product
- To detect the faults in the Software before User finds it
- **To know the reliability of the software**
- Undetected errors are costly to detect at a later stage
- To satisfy users and to lower the maintenance cost

Why is testing becoming such a crucial activity?

Because applications are becoming very complex with n-tiers in an application. When one tests a program one adds value to it through improved quality and reliability.

If not tested it can cause an unpleasant navigational error in case of a browsing applications or death or injury in case of safety critical applications.

End customers are becoming more demanding & conscious about quality

Why are company's outsourcing the testing phase?

It is being realized that testing is an extremely important phase, customers today are conscious of quality as they need to be more competitive in the market.

It is being realized that the best people to test an application are the ones who have not developed the application. The testers would have the approach of a user and have an unbiased mind.

Error-Failure-Defect

- Error(Mistake): A human action that produces an incorrect result
- Fault: A stage caused by an error which leads to unintended functionality of the program
- Bug: It is an evidence of the fault. It causes the program to perform in unintended manner. It is found before application goes into beta version
- Failure: Inability of the system to perform functionality according to its requirement
- Defect: It is a mismatch of the actual and expected result identified while testing the software in the beta version



Copyright © Capgemini 2015. All Rights Reserved 10

Error-Failure-Defect : Example

Consider the below program for addition of two integers;

```
#include<stdio.h>
```

```
1. int main ()
2. {
3.     int num1, num2, sum;
4.     num1 = 6;
5.     num2 = 4;
6.     sum = num1 – num2;
7.     printf("6 + 4 = %d", sum);
8. }
```

Output : 6 + 4 = 2

After compiling and running this program we see that the program has failed to do what it was supposed to do. The program was supposed to add two numbers but it did not. The result of 4 + 6 should be 10, but the result is 2. For now we have detected a **failure**. The **fault** in the program is the line 7 has '-' sign instead of '+' sign which led to a failure (deviation from the required functionality). In this case we can also say we have found the **bug**. **Error** is the mistake programmer made by typing '-' instead of '+' sign. The tester is testing the functionality of the program and realizes the output is faulty and will raise a **defect**.

Software that does not work correctly can lead to many problems including loss of money, time, business reputation & could even cause injury or death.

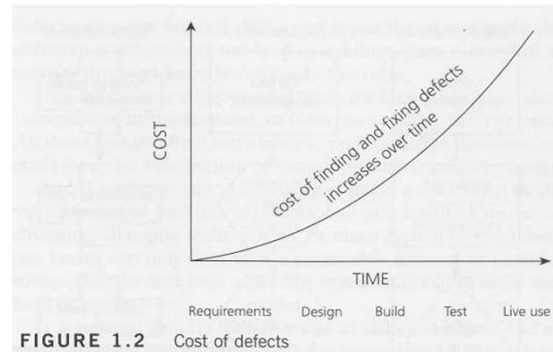
Few examples : Ariane 5 Space Program (\$7billion), Mariner space probe to Venus (\$250m), American Airlines (\$50m)

Causes of Software Defects

- Software is written by human beings
 - Who know something, but not everything
 - Who have skills, but aren't perfect
 - Who do make mistakes (errors)
- Under increasing pressure to deliver to strict deadlines
 - No time to check but assumptions may be wrong
 - Systems may be incomplete
- Environmental conditions
 - Radiations, Magnetism, Electronic fields and pollution can cause faults in firmware or influence execution of software by changing hardware conditions
- Minimal or no proper documentation of Business Requirements
- Insufficient time window for development
- Lack of domain knowledge
- Programming Language constraints

Cost of Software Defects

- It is Easy to find and fix defect in early stages rather than in the later phases of software.



Cost of Software Defects

The cost of fixing a bug (defect) and making the required changes in early phases of software development is less as compared to the same detected in later phases since cost is spent at four different times in a SDLC;

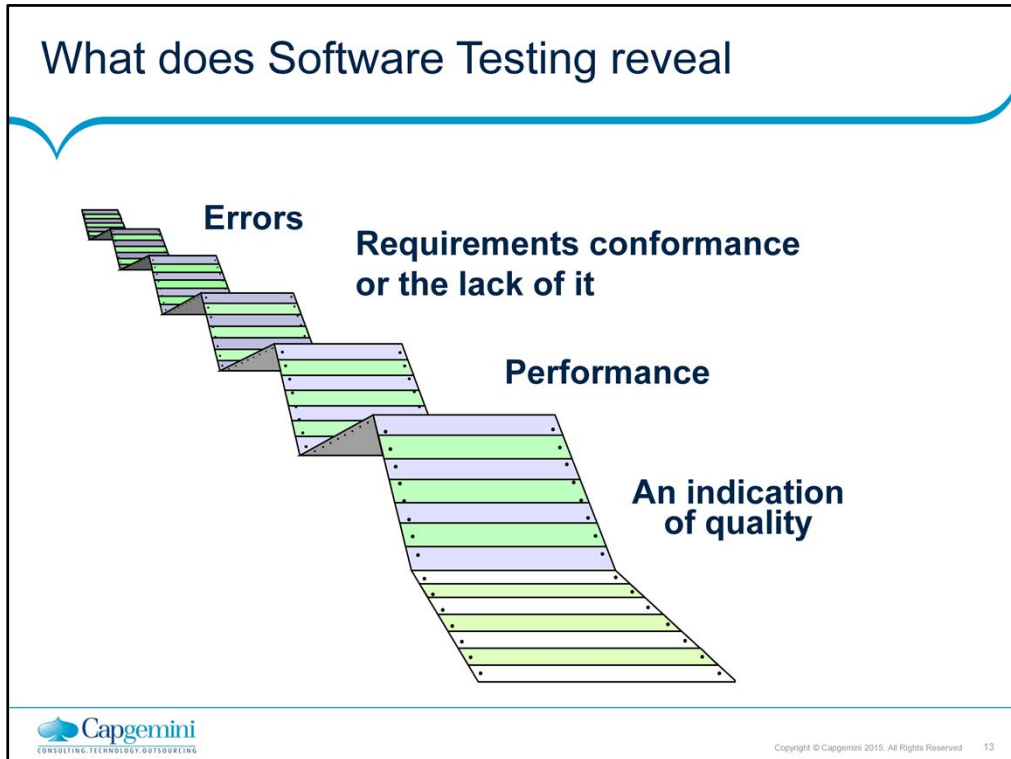
First, cost is spent in writing wrong specifications, erroneous coding and incorrectly documenting the system

Second, cost is spent on detecting the errors rather on preventing errors

Third, cost is spent on removing discovered errors in specifications, coding, and documentation

Fourth, cost is spent on re-testing the system to determine whether the errors have been fixed.

Therefore, to achieve lower cost and high quality systems, testing must not be considered as single phase activity; it must be incorporated in all SDLC phases.



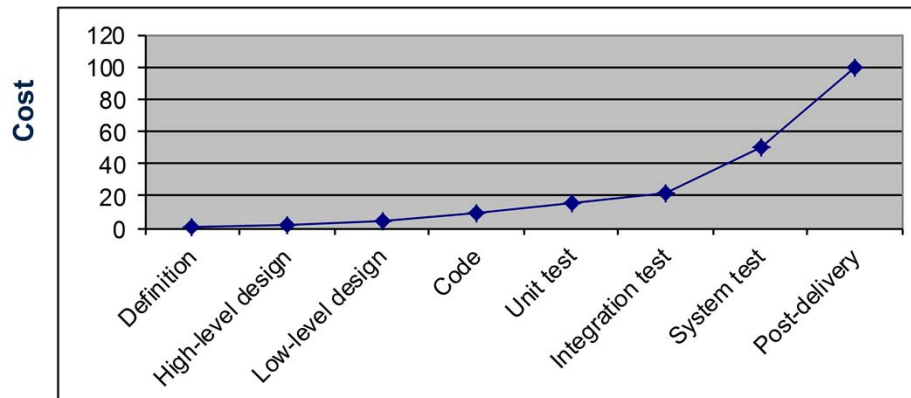
Software testing exposes all the possible, noticed & unnoticed Errors.
It also takes care of the requirements conformance with respect to the end product.
It checks for the performance test.
The very crucial aim of all of these activities is Quality. It ensures the product in terms of quality.

Importance of Software Testing

- Ensures that the product is usable
- Ensures that Customer's Objectives are met
- Early detection of errors to prevents breakdown at a later stage
- Ensures that the software is reliable
- Builds Confidence in software
- Increases Customer Satisfaction
- Ensures effective execution in the given environment
- Reduces overall cost of software
- Reduces time for going live (production)

Importance of Testing Early in SDLC Phases

■ Error removal cost over SDLC



Importance of Testing Early in SDLC Phases

- Prevents future Problems thus lowering the cost
- Testing will not be a bottleneck anymore
- Testers become more familiar with the software, as they are more involved with the evolution of the product.
- Reduces the chances of failure
- The test environment can be prepared in advance
- The risk of having a short time for testing is greatly reduced
- Maintains “quality culture” in the organization



Copyright © Capgemini 2015. All Rights Reserved 16

Importance of Testing Early in SDLC Phases

Many problems arise during planning or design. Requirements testing can prevent future problems thus lowering the cost.

Since the testing process is involved with all phases of the SDLC, Management will not feel as if testing is a bottleneck to release the product.

Test cases written during requirements and shared with the Dev. team before the construction phase can help developers to reduce the chances of failure

The test environment can be prepared in advance

The risk of having a short time for testing is greatly reduced

Involving quality assurance in all phases of the SDLC helps creating a 'quality culture' inside the organization.

Testing and Quality

- Testing helps to measure the Quality of software in terms of
 - Number of defects found
 - important information regarding the Non functional attributes like Reliability, Security, Performance, etc.
- Quality Definition
 - Conformance to explicitly stated functional and performance requirements, explicitly documented development standards and implicit characteristics that are expected from all professionally developed software.



Copyright © Capgemini 2015. All Rights Reserved 17

Importance of Software Quality
Aim : Customer & User Satisfaction
Quality is a competitive issue
Quality is must for survival
Quality gives entry into International market.
Quality is cost effective
Quality helps retain customers

Quality Perceptions

- Engineer may judge :
 - User satisfaction
 - Portability
 - Maintainability
 - Robustness & Efficiency
- Customer may judge :
 - Cost
- User may judge :
 - Reliability
 - usability

Seven Testing Principles

- Principle 1 - Testing shows presence of defects but cannot prove that there are no defects
- Principle 2 - Exhaustive testing is impossible
- Principle 3 - Early testing
- Principle 4 - Defect Clustering
- Principle 5 - Pesticide Paradox
- Principle 6 - Testing is context dependent
- Principle 7 - Absence of Errors fallacy



Copyright © Capgemini 2015. All Rights Reserved 19

Testing Principles

Although testing is itself an expensive activity, the cost of not testing is potentially much higher. The most damaging errors are those which are not discovered during the testing process and therefore remain when the system goes live.

Testing shows presence of defects but cannot prove that there are no defects
Exhaustive testing is impossible. Risk analysis & prioritisation are used to focus testing effort.

Early testing : Testing activity is started as early as possible in the SDLC to find defects early

Defect Clustering : Testing effort should be focused proportionally to the expected or later observed defect density of modules. A small number of modules usually contains most of the defects.

Pesticide Paradox : If same set of tests are repeated, then NO defects found. To overcome this problem, test cases need to be reviewed and revised regularly to assess different parts of the software or systems.

Testing is context dependent : Safety Critical Software is tested differently compared to an e-commerce application

Absence of Errors fallacy: Finding & fixing of errors does not help if the system built is unusable and does not fulfil the users' needs and expectations.

Economics of Testing

- Economics of Testing
 - It is both the driving force and the limiting factor
- Driving - Earlier the errors are discovered and removed in the lifecycle, lowers the cost of their removal.
- Limiting - Testing must end when the economic returns cease to make it worth while i.e. the costs of testing process significantly outweigh the returns



Copyright © Capgemini 2015. All Rights Reserved 20

Although testing is itself an expensive activity, the cost of not testing is potentially much higher. The most damaging errors are those which are not discovered during the testing process and therefore remain when the system goes live.

Limiting - It is infeasible to test exhaustively all but the most simple or the most vital of s/w.

Here we discuss how exhaustive testing is impossible to achieve.

Consider the following example:

Exhaustive input testing (Black box)

Lets assume that we have written a function say $ax^2 + bx + c = 0$

Assume 16 bit numbers

So each input is 2^{16}

And so total test cases is $2^{16} \times 2^{16} \times 2^{16} = 2^{48}$ test cases which is impractical.

How Testing is conducted?

- By examining the users' requirements
- By reviewing the artifacts like design documents
- By examining the design objectives
- By examining the functionality
- By examining the internal structures and design
- By executing code

How testing is conducted?

Testing is conducted with the help of users requirements, design documents, functionality, internal structures & design, by executing code.

Software Testing – Then (Past)

- Testing led to blame each other and giving faulty justifications and excuses which would arise quarrels among the team members
- Not much re-work was welcomed
- Developers misunderstood that increased testing would increase the project cost unnecessarily
- It was understood that software has quality if it is just easy to maintain, reusable and flexible.
- Testing was conducted only at project execution phase
- Ad-hoc & need driven
- Totally manual
- Testing jobs perceived as low scale as compared to other discipline

Software Testing – Now (Present)

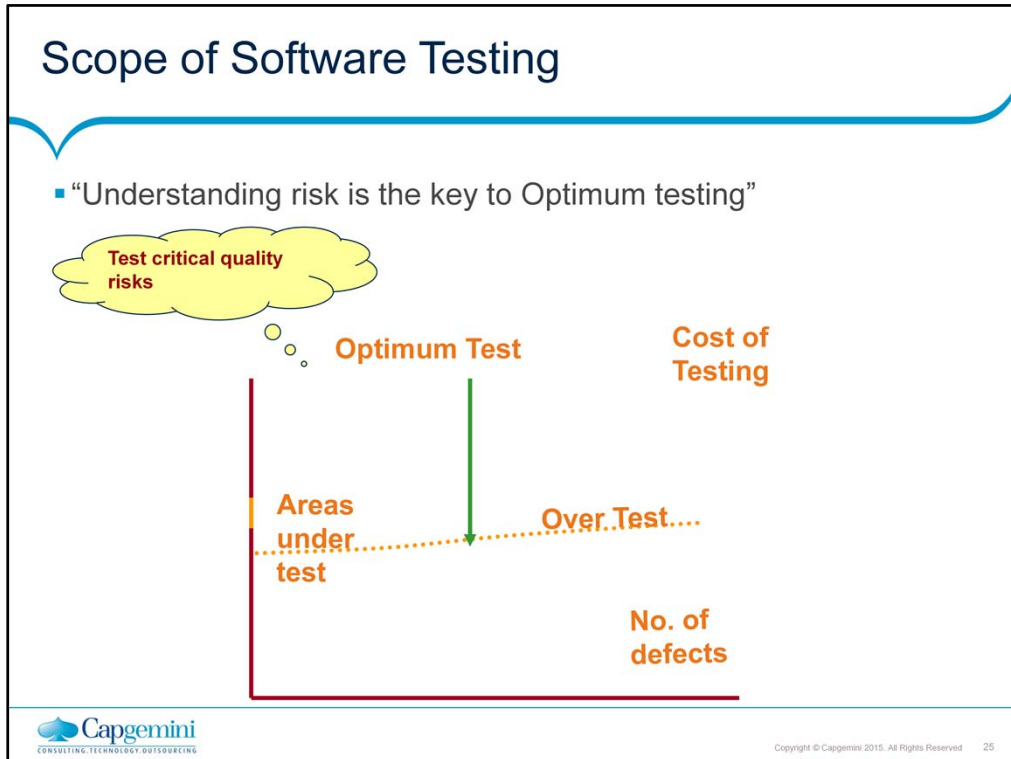
- Organized body of knowledge
 - Independent Testing teams
 - Testing Models
 - Testing knowledge Groups
- A specialized engineering discipline in great demand
- Defects are highlighted and brought to surface so as to conduct corrective measures and achieve user satisfaction
- Testing lead to cooperative solutions so as to prevent the failures
- Documentation is considered as essential to note down the lessons learnt so that mistakes are not repeated
- Developers know that testing increases the business profit
- Describes software quality in terms of integrity, reliability, usability and accuracy
- Testing is conducted as the project initiates

Scope of Software Testing

- Bad news : You can't test everything
- Good news : There is such a thing as "good enough"
- Bad news : Good enough may cost too much

- What do we do : Increase focus via systematic process of elimination

- What you might test?
 - Those areas which are within the scope of your project
- What you should test?
 - The critical system functionality which effects the customers & users experience of quality
- What you can test?
 - Estimate time & resource for risk driven effort



Factors influencing the Scope of Testing

- Contractual requirements
- Legal requirements
 - Privacy related laws
 - Non-disclosure of identity
- Industry-specific requirements
 - Aircraft safety equipment
- Scope of Testing is about identifying the correct test cases for automation
- The steps involved are:
 - Identify various factors that form the basis of identifying the candidate test cases
 - Apply 'Divide & rule' strategy : Break the application into smaller modules
 - Analyze each module to identify the candidate test cases
 - Calculate ROI
- Factors influencing the scope of testing :
 - In small projects
 - Test case writing
 - Test case execution
 - Regression testing
 - In Large projects
 - Setting up test bed
 - Generating test data, test scripts, etc.

Risk Based Testing

- Risk:
 - A factor that could result in negative consequences; usually expressed as impact and like hood
 - Risks are used to decide where to start testing and where to test more.
 - Risk Based testing:
 - Testing oriented towards exploring and providing information about product risks
- Risk based Testing is used to reduce risk of adverse effect occurring or to reduce the impact of adverse effect
 - It draws on the collective knowledge and insight of the project stakeholders to determine the risk and the level of testing required to address those risks.



Copyright © Capgemini 2015. All Rights Reserved 27

Risk identified in the Risk Based testing is used to:

Determine the test techniques to be employed

Determine the extent of testing to be carried out

Prioritize testing in an attempt to find the critical defect as early as possible

Determine whether any non testing activities could be employed to reduce risk

Risk Management activities provide a discipline approach to minimize the product failure:

Assess and reassess what can go wrong (risks)

Determine what risks are important to deal with

Implement action to deal with those risks

Project Risks

- Project Risk
 - A risk related to management and control of the (test) project is called as Project Risk
- Project risk are:
- Organizational factor
 - Skill, training and staff shortage
 - Personnel issues
 - Political issues
 - Improper attitude toward or expectation of testing
- Technical issues
 - Problem in defining right requirements and quality of the design code, test data and test
 - The extent to which requirement cannot be met given existing constraints
 - Test environment not ready on time or late data conversion, migration planning and development and testing data conversion/migration tools
- Supplier issues
 - Failure of a third party
 - Contractual issues

Product Risks

- Product Risk: it is directly related to the test object
- Risks related to quality of a product:
 - Failure-prone software delivered
 - The Potential that the software/hardware could cause harm to an individual or company
 - Poor software characteristics
 - Poor data integrity and quality
 - Software that does not perform its intended functions

Need of Independent Testing

- Unbiased testing is necessary to objectively evaluate quality of a software
- Developer carrying out testing would not like to expose defects
- Assumptions made are carried into testing
- People see what they want to see.
- More effective in terms of Quality & Cost
- It is conducted by an independent test team other than developer to avoid author bias and is more effective in finding defects and failures
- The tester sees each defect in a neutral perspective
- The tester is totally unbiased
- The tester sees what has been built rather than what the developer thought
- The tester makes no assumptions regarding quality

Activities in Fundamental Test Process

- Test planning and control
 - It defines the objectives and specification of test activities
 - Test control is the on going activity of comparing actual progress against the plan
- Test analysis and design
 - Testing objectives are transformed into tangible test conditions and test cases
 - Reviewing the test basis
 - Evaluating testability of the test basis and test objects
 - Identifying, Designing and Prioritizing test conditions based on analysis of test item, the specification, behaviour and structure of the software
 - Designing and Prioritizing high level test cases
 - Identifying necessary test data
 - Designing test environment setup and identifying required infrastructure and tools
 - Creating bi-directional traceability between test basis and test cases

Activities in Fundamental Test Process

- Test implementation and execution
 - Test procedures (scripts) are specified by combining test cases in a particular order and the environment is set up and tests are run
 - Finalizing, implementing and prioritizing test cases
 - Developing and prioritizing test procedures, creating test data and writing automated test scripts
 - Creating test suites from test procedures for efficient test execution
 - Verification if Test environment is setup correctly
 - Verifying and updating bi-directional traceability between test basis and test cases
 - Executing test procedure using tool or manually according to the planned sequence
 - Logging the outcome of the test execution and recording the identities and version of the software under test tools and test ware
 - Comparing actual result with expected result
 - Reporting discrepancies and analysing their root cause
 - Repeating test activities as a result of action taken for each discrepancy

Activities in Fundamental Test Process

- Evaluating exit criteria and reporting
 - Test execution is assessed against the defined objectives
 - Test logs and checked against the exit criteria specified in test planning
 - Assessment is done if more tests are needed
 - Test summary report is written
- Test closure activities
 - Data from completed test activity is collected to consolidate experience, facts and numbers
 - This is done at milestones

Testware: Artifacts produced during the test process required to plan, design and execute test and any additional software or utilities used in testing.

Attributes of a good Tester

- A good test engineer has a 'test to break' attitude
- Need to take a different view, a different mindset (“What if it isn’t?”, “What could go wrong?”)
- An ability to understand the point of view of the customer
- A passion for quality and attention to detail
- Notice little things that others miss/ignore (See symptom not bug)
- Ability to communicate fault information to both technical (developers) and non-technical (managers and customers)
- Tact and diplomacy for maintaining a cooperative relationship with developers
- Work under worst time pressure (at the end)
- “Patience”

Psychology of Testing

- Test Engineers pursue defects not people
- Don't assume that no error(s) will be found
- Test for Valid and Expected as well as Invalid and Unexpected
- The probability of the existence of more errors in a section of a program is proportional to the number of errors already found in that section
- Testing is extremely creative and intellectually challenging



Copyright © Capgemini 2015. All Rights Reserved 35

Testing is in a way a destructive process and a successful test case is one that brings out an error in the program . Detection of an error/failure is a success as far as a test engineer is concerned.

The mindset to be used while testing and reviewing is different from that used while developing software.

Separation of responsibilities of development and testing are done to help focus efforts and provide an independent / unbiased view.

While a certain level of independence often makes the tester more effective at finding defects and failures, this independence can however not replace familiarity which developers possess.

People and projects are driven by objectives. For example, to find defects and confirm that the software meets its objectives. It is therefore important to clearly state the objectives of testing.

Identifying failures during testing may be perceived as criticism against the product and its author. Testing is therefore often viewed as a destructive activity.

Communication of errors in the product in a constructive way therefore assumes particular importance in order to make testing appear constructive and supportive.

The Test Leaders and Testers therefore need to have good interpersonal and communication skills to overcome this difference of perception.

Some ways of improving communication and relationships between testers and others:

- Start with collaboration rather than battles – Common goal of better quality system

- Communicate findings in a neutral, fact-focussed way, without criticising individuals who created the tested products

- Try to understand what the other person feels and why they react in the way they do

- Confirm that the other person has understood what you conveyed.

Read the following example:

A doctor asks a patient to get all the laboratory tests and if the test could not locate any problem then it is not a successful test as it did not find out the cause of the patient's problems and only wasted the laboratory fees and efforts in getting the tests done.

The psychology of people towards treating testing as the process of demonstrating that errors are not present and treating testing as the process of uncovering errors needs be discussed here

Retesting is required when errors are found. So the test cases need to be documented and retained. This gives us indication on where the errors are found most and make more investments and investigation on those areas.

Writing automated tests is harder than writing the code itself, in many cases. The most expert programmers are the best testers. When faced with seemingly mundane coding tasks, coming up with creative tests provides an intellectual challenge that expert programmers thrive on.

Beginners typically need expert assistance when writing tests. This is where pair-programming helps, because experts work side-by-side with beginners as they learn the art of testing.

Just like coding is an art, testing is an art.

In many organizations, testing is relegated to the least-experienced programmers. We often encounter the misconception that testing consists of people completing written checklists as they manually execute the application. This approach is completely not scalable, because it takes longer and longer for humans (monkeys?) to test every feature as the application grows.

Modern OO languages like Java are complex, particularly when it comes to dependencies between classes. One change can easily introduce bugs in seemingly unrelated classes. Gone are the days when each character-based screen is a standalone program. OO apps are far more complex and demand automated tests.

Code of Ethics for Tester

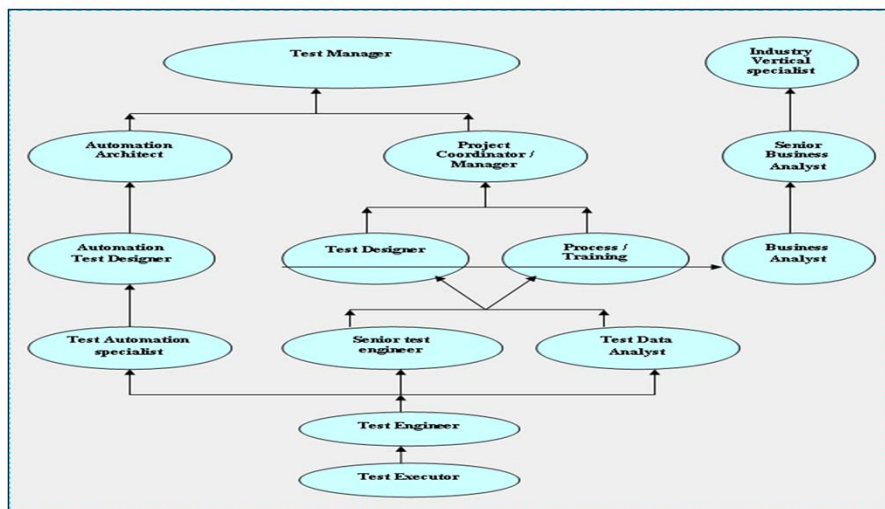
Involvement in software testing enables individuals to learn confidential and privileged information. A code of ethics is therefore necessary, among other reasons to ensure that the information is not put to inappropriate use.

- PUBLIC - Tester shall act consistently with public interest
- CLIENT AND EMPLOYER - Tester shall act in best interests of their client and employer
- PRODUCT - Tester shall ensure that the deliverables they provide meet highest professional standards possible
- JUDGMENT - Tester shall maintain integrity and independence in their professional judgment
- MANAGEMENT - Tester managers and leaders shall subscribe to and promote an ethical approach to manage software testing
- PROFESSION - Tester shall advance the integrity and reputation of the profession consistent with the public interest
- COLLEAGUES - Tester shall be fair to and supportive of their colleagues, and promote cooperation with software developers
- SELF - Tester shall participate in lifelong learning and shall promote an ethical approach to the practice of the profession

FS SBU: Focus on Testing

- Fastest growing service line is FS SBU (Financial Services Strategic Business Unit)
- Team of 1600+ test engineers
- Was CMM level 5 assessed within a year of being established
- Presence in most of the existing FS SBU accounts
- High Focus on competency building
- Do almost all types of testing including Performance monitoring and Public Website testing
- Offering Security Testing

Testing Roles in iTEAMS



Limitations of Software Testing

- Even if we could generate the input, run the tests, and evaluate the output, we would not detect all faults
- Correctness is not checked
 - The programmer may have misinterpreted the specs, the specs may have misinterpreted the requirements
- There is no way to find missing paths due to coding errors

Summary

- In this lesson, you have learnt:

- Testing is an extremely creative & intellectually challenging task
- No software exists without bug
- Testing is conducted with the help of users requirements, design documents, functionality, internal structures & design, by executing code
- Scope of testing
- The cost of not testing is potentially much higher
- Testing is in a way a destructive process
- A successful test case is one that brings out an error in program
- Various principles of testing



Review Question

- Question 1: What is visible to end-users is a deviation from the specific or expected behavior is called as
 - Defect
 - Bug
 - Failure
 - fault
- Question 2: _____ is a planned sequence of actions
- Question 3: Pick the best definition of Quality :
 - Quality is job done
 - Zero defects
 - Conformance to requirements
 - Work as designed
- Question 4: One cannot test a program completely to guarantee that it is error free
 - Option: True / False
- Question 5: One can find missing paths due to coding errors
 - Option: True / False



Review Question: Match the Following

1. Economics of limiting

2. Testing

3. A good test case

4. Use every possible input condition as a test case

A. Driving

B. Exhaustive testing

C. Limiting

D. Test cycle

E. Comparing outputs with specified or intended

F. Maximize bug count

