RASHED KARIM

ISTQB Tester

mail4rashed@gmail.com
Mobile: 01711605286

Chapter-I

Fundamentals of Testing

I-Fundamentals of Testing
Agenda

Chapter-1- Fundamentals of software testing

-1/01 Understanding software testing

-1/02 The seven general principles of software testing

-1/03 Fundamental test process

-1/04 The psychology of testing

I-Fundamentals of Testing
01- Understanding software testing

The economic importance of software

- The functioning the machines and equipment depends largely on software
- We cannot imagine large systems in telecommunication, finance or traffic control running without software.

Software quality

- More and more, the quality of software has become the determining factor for the success of technical or commercial systems and products

Testing for quality improvement

- Testing and reviewing insure the improvement of the quality of software products as well as the quality of software development process itself.

Failure example 1: Ariane 5 launch

Flight 501, which took place on 4th June1996 was the first test flight of the Ariane 5 expendable launch system. It was not successful; the rocket tore itself apart 37 seconds after launch because of a malfunction in the control software, making the fault one of the most expensive computer bugs in history. The Ariane 5 software reused the specifications the Ariane 4, but the Ariane 5's flight path was considerably different and beyond the range for which the reused code had been designed. Specifically, the Ariane 5's greater acceleration caused the back-up and primary inertial guidance computers to crash, after which the launcher's nozzles were directed spurious data. Pre-flight tests had never been performed on the re-alignment code under simulated Ariane 5 flight conditions, so the error was not discovered before launch.

Source : Wikipedia.com

Failure example 2: Lethal X-Rays

Because of a software failure a number of patients received a lethal dose a gamma rays:

Therac-25 was a radiation therapy machine produced by atomic Energy of Canada Limited. It was involved with at least six known accidents between 1985 and 1987, in which patients were in some cases on the order of hundreds of gray. At least five patients died of the overdoses. These accidents highlighted the dangers of software control of safety-critical system.

Source: Wikipedia.com

I-Fundamentals of Testing
01- Understanding software testing

Causes of software failures

Human error

- A defect was introduced into the software code, the data or the configuration parameters

Causes of human error

- time pressure excessive demands because of complexity distractions

Environmental conditions

- changes of environmental conditions
- Causes of negative environmental conditions
- radiation, magnetism, electronic field on pollution sun spots, hard disk crashes , power functions.

Error, Defect, Failure

- **Error(IEEE 610):**

a human action that produces an incorrect result, e.g. a programming error

- **Defect:**

a flaw in a component or system to fail to perform its required function,
e.g. an incorrect statement of data definition.

- **Failure:**

the physical or functional manifestation of a defect.
A defect, if encountered during execution, may cause a failure.
➢ Deviation of the component or system from its expected delivery, service or result.

Defects cause failure

I-Fundamentals of Testing
01- Understanding software testing

Testing during software development, maintenance and operation

- **Increasing software quality:**

Testing helps to furnish the software with the desired attributes,
i.e. to remove defects leading to failures.

- **Reduction of the risk of encountering errors:**

Appropriate test activities will reduce the risk that errors are encountered during software operation

- **Meeting obligations:**

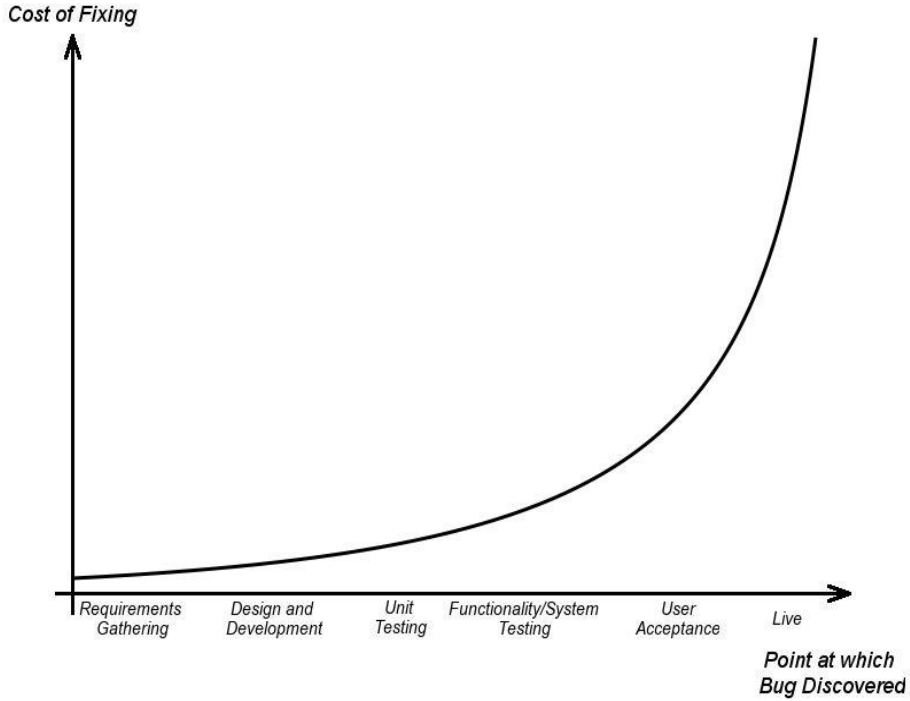
Tests might be mandatory because of client's or legal regulation as well as to meet industrial standards.

I-Fundamentals of Testing
01- Understanding software testing

Costs of defects

- The costs of fixing defect is increase with the time they remain in the system.

-Detecting errors at an early stage allows for error correction at reduced cost.



I-Fundamentals of Testing

01- Understanding software testing

Software quality

- **Definition Software (as per IEEE 610):**

  Computer programs, procedures and possibly associated documentation and data pertaining to the operation a computer system

- **Definition Software quality (as per IEEE std 610):**

  The totality of functionality and features of a software product that contribute to its ability to satisfy stated or implied needs.

- **Definition Quality (as per IEEE std 610):**

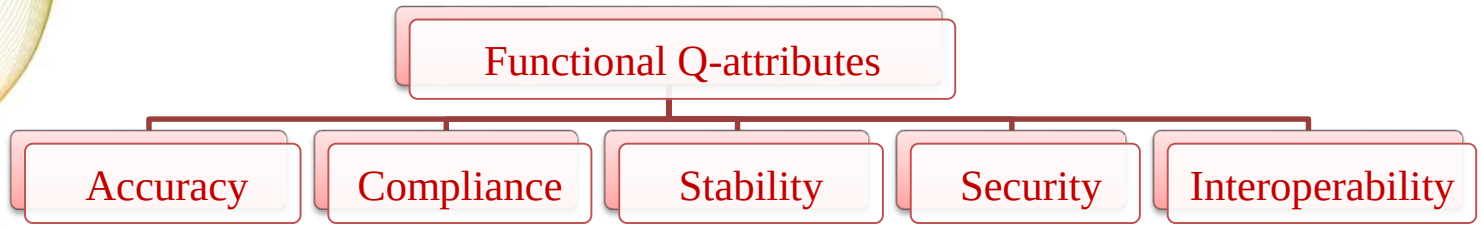
  The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations.

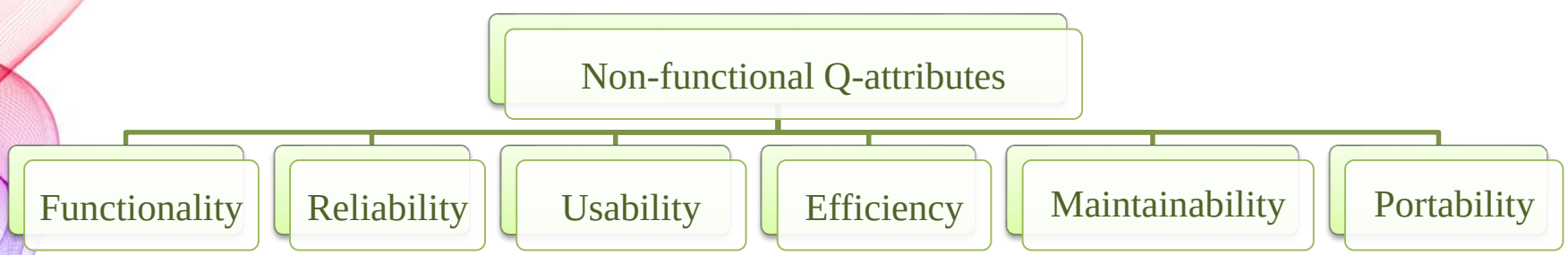
I-Fundamentals of Testing

01- Understanding software testing

Software quality

- **According to ISO/IEC 9126 software quality consists of:**





Types of Quality Assurance (QA):

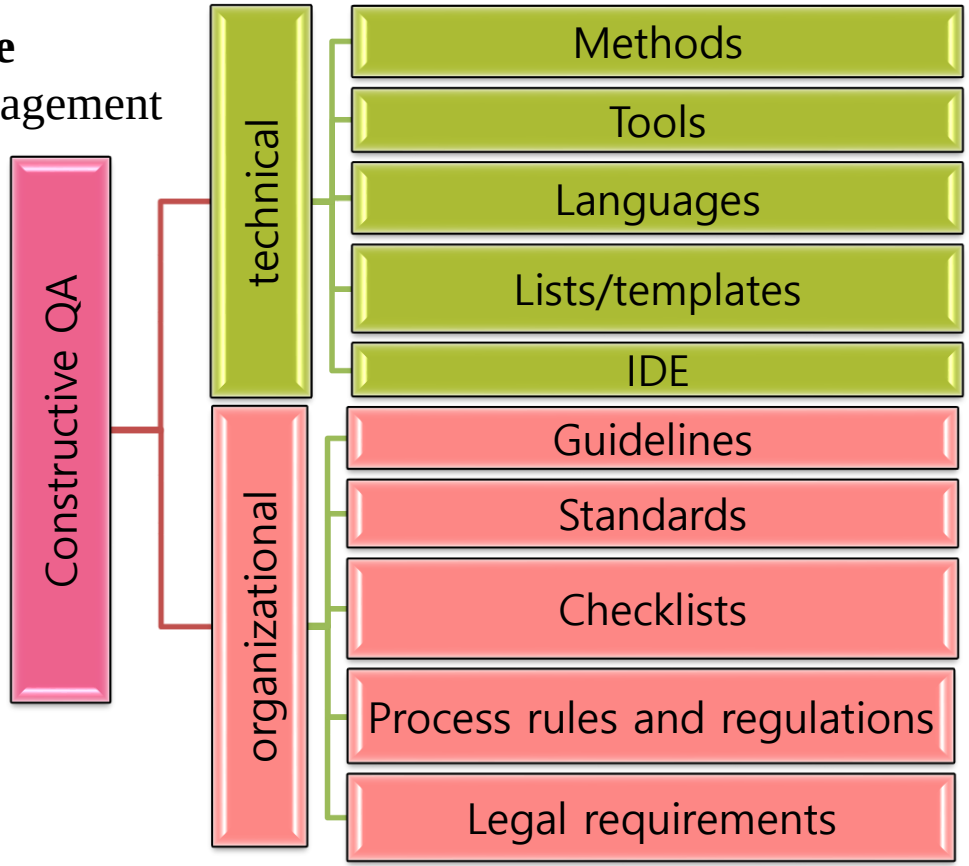
1. constructive activities to prevent defects, e.g. through appropriate methods of software engineering
2. analytical activities for finding defects, e.g. through testing leading to correcting defects and preventing failures, hence increasing the software quality.

I-Fundamentals of Testing

01- Understanding software testing

Constructive quality assurance

Quality of process- Quality management



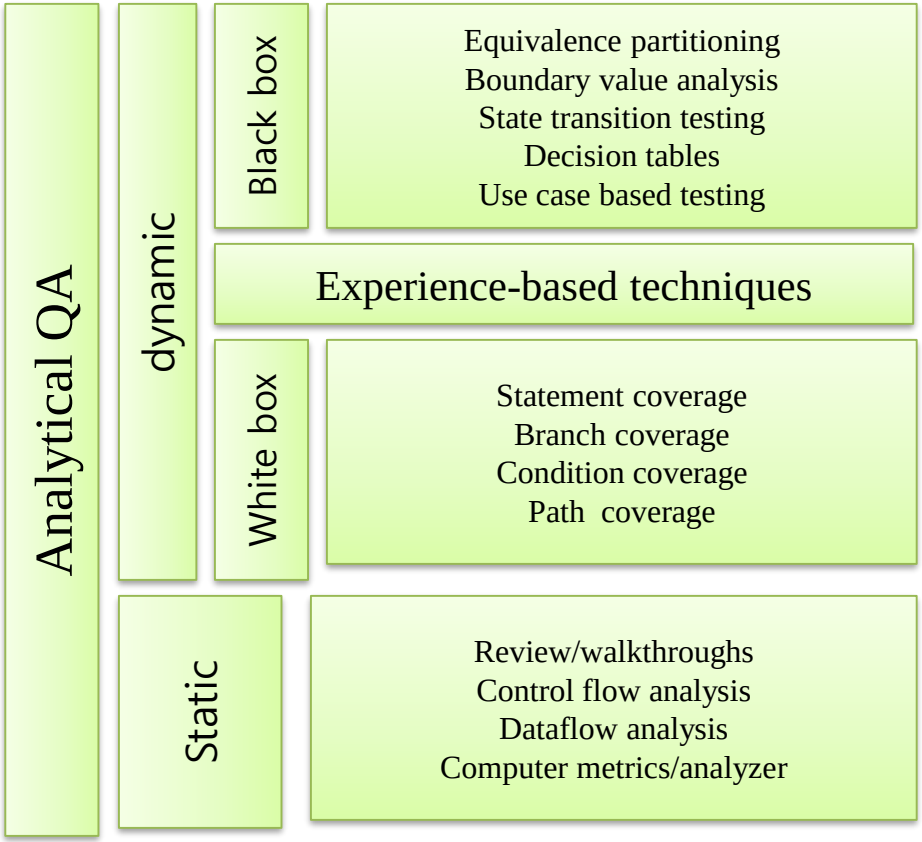
I-Fundamentals of Testing

01- Understanding software testing

Analytical quality assurance
Quality of product- Verification and test procedure

Motto

- defects should be detected early as possible in the process
- **Static testing** examination without excluding the program
- **Dynamic testing** includes executing the program



I-Fundamentals of Testing
01- Understanding software testing

Software quality- functional Q-attributes

- ***Functionality means:***

  Correctness: the functionality meets the required attributes / capabilities

  Completeness: the functionality meets all(functional) requirements.

- **Functionality includes (as per ISO/IEC 9126):**

* Stability
* Accuracy
* Interoperability
* Security

I-Fundamentals of Testing
01- Understanding software testing

Software quality- non-functional Q-attributes/1

Reliability

* maturity, fault tolerance, recovery after failure.
* characteristic: under given conditions, a software/ a system will keep its capabilities/ functionality over a period of time.
* reliability = quality/time

Usability

* Learn ability ,understanding ability, attractiveness.
* Characteristics: easy to learn, compliance with guidelines, intuitive handling.

I-Fundamentals of Testing

01- Understanding software testing

Software quality- Non-functional Q-attributes/2

Efficiency

- System behavior: functionality and time behavior.
- Characteristics: the system requires a minimal use of resources (e.g. CPU-time) for exec uting the given task.

Maintainability

- Verifiability, stability, analyzability, changeability.
- Characteristics: amount of effort needed to introduce changes in system components.

Portability

- Reparability, compliance, install ability.
- Ability to transfer the software to a new environment (software, hardware, organization)
- Characteristics: easy to install and uninstall, parameters.

I-Fundamentals of Testing
01- Understanding software testing

Quality attributes

- Some software quality attributes are influenced reciprocally, Because of this, depending on the test object, attributes must be prioritized, e.g. efficiency vs. portability.
- Different kinds of tests will be performed in order to measure the different kinds of attributes .

I-Fundamentals of Testing

01- Understanding software testing

Test goals

- **Gain knowledge about defects in the test objects**

  Defects contained in the test objects must be detected and be described in such a way as to facilitate their correction

- **Poor of functionality**

  System functionality should be implemented as specified

- **Generating information**

  before handling over a software system to the users, information about possible risks has to be provided. Gaining such information might be one of the test goals.

- **Gaining confidence**

  Software that has been well tested in trusted to meet the expected functionality and to have a high quality level.

How much testing is enough?

Exit criteria

not finding (any more) defects is not an appropriate criterion to stop testing activities. Other metrics are needed to adequately reflect the quality level reached.

Risk based testing

Levels of risk determine the extent of testing carried out, i.e. liability for damages in case of failures occurring, economic and project related aspects.

Time and budget testing

The amount of resources available (personal, time and budget) might determine the extent of testing efforts.

I-Fundamentals of Testing

01- Understanding software testing

Test case, test basis

Test case (as per IEEE 829):

test case definitions include at least the following information

- pre-condition
- set of input values
- set of expected results
- expected post conditions
- unique identifier
- dependence on other test cases
- reference to the requirement that will be tested
- how to execute the test and check results (optional)
- priority(optional)

Test basis: (also: test base)

Set of documents defining the requirements of a component or system. Used as the basis for the development of test cases

I-Fundamentals of Testing

01- Understanding software testing

Software development and reviews

Code, source code:

A computer program, written in a programming language, which can be read by human beings.

Debugging:

Locates and corrects defects in the source code.

Software-Development:

Is a complex process/sequence of activities aiming at implementing a computer system. It usually follows a software development model.

Requirement

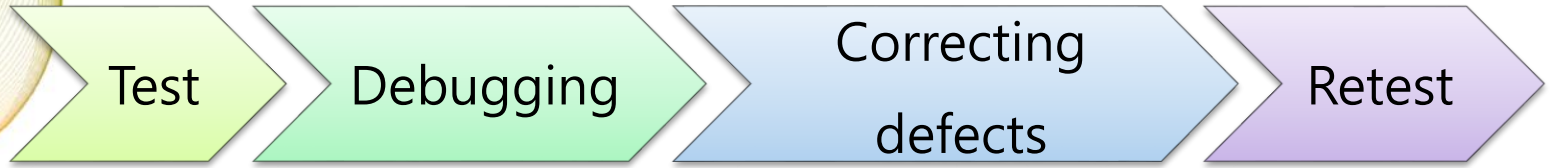
A requirement describes a functional attribute that is desired or seen as obligatory.

Review (after IEEE 1028)

Evaluation of a product or project status to find discrepancies from planned result and to recommend improvements.

I-Fundamentals of Testing
01- Understanding software testing

Testing and debugging



Test and re-test are test activities

- testing shows system failures.
- Re-testing proves, that the defect has been corrected.

Debugging and correcting defects are developer activities

- Through debugging, developers can reproduce failures, investigate the st ate of programs and find the corresponding defect in order to correct it.

I-Fundamentals of Testing

01- Understanding software testing

Summary:

- **Software failures** may cause enormous damage
- **Software quality (SQ)** is the sum of attributes that refer to the capability of the software to meet given requirements
- **Constructive** software quality, assurance deals with preventing defects
- **Analytical** software quality, assurance deals with finding defects and correcting them
- Functional and non-functional **quality attributes** define the total quality of the system
- Each test has to have predefined exit criteria reaching the **exit criteria** will conc lude testing activities.
- Testers look for failures in the system and report them **(testing)** .
  Developers look for detects and correct them (**debugging**)

I-Fundamentals of Testing

02- The seven general principles of software testing

Principle 1: Testing shows the presence of defects

- ✓ Testing can prove the presence of defects.
- ✓ Deviations discovered while testing show a failure.
- ✓ The cause of the failure might not be obvious.
- ✓ Testing reduces the probability of defects remaining undiscovered. The absence of failure does not prove the correctness of the software.
- ✓ The test procedure itself might contain error
- ✓ The test conditions might be unsuitable for finding errors.

Principle 2: Exhaustive testing is not possible

Exhaustive testing

A test approach in which the test suite comprises all combinations of input values and preconditions

Test case explosion

defines the exponential increase of efforts and costs when testing exhaustively

Sample test

The test includes only a (systematically or randomly derived) subset of all possible input values

Under real life conditions, sample tests are generally used. Testing all combination of inputs and preconditions is only economically feasible in trivial cases

I-Fundamentals of Testing

02- The seven general principles of software testing

Principle 3: Early testing

- ✓ The earlier a defect is discovered, the less costly is its correction.
- ✓ Highest cost effectiveness when errors are corrected before implementation
- ✓ Concepts and specifications may already be tested.
- ✓ Defects discovered at the conception phase are corrected with the least effort a nd costs.
- ✓ Preparing a test is time consuming as well.
- ✓ Testing involves more than just test execution.
- ✓ Test activities can be prepared before software development is completed.
- ✓ Testing activities (including reviews) should run in parallel to software specific ation and design.

I-Fundamentals of Testing

02- The seven general principles of software testing

Principle 4: Defect clustering

- ✓ Find a defect and you will find more defects nearby!
- ✓ Defects often appear clustered like mushrooms or cockroaches
- ✓ It is worth screening the same module where one defect was found
- ✓ Testers must be flexible
- ✓ Once a defect is found it is a good idea to reconsider the direction of further testing.
- ✓ The location of a defect might be screened at higher detail level, e.g. starting additional tests or modifying existing tests.

Principle 5: Pesticide paradox

- ✓ **Repeating tests under the same conditions is ineffective.**
- ✓ Each test case should contain a unique combination of input parameters for a signal test objects, otherwise no additional information can be gained.
- ✓ If the same tests are repeated over and over again, no new bugs can be found
- ✓ **Tests must be revised regularly for different code modules.**
- ✓ It is necessary to repeat a test after changes have been made in the code (bug fixing, new functionality).
- ✓ Automating tests can be an advantage if a group of tests cases is used regularly.

I-Fundamentals of Testing

02- The seven general principles of software testing

Principle 6: Testing is context dependent

- ✓ **Testing is done differently in different contexts**
- ✓ **Different test objects are tested differently.**
- ✓ The engine controller of a car requires tests **different** than those of an ecommerce application
- ✓ **Test environmental (test bed) vs. production environment**
- ✓ Test take place on an environment other than the production environment. The test environment should be **very similar** to the production environment.
- ✓ There will always be **deviations** between test environment and the production environment. These deviations impeach the **conclusions** drawn after testing.

Principle 7: Absence of errors fallacy

- ✓ **Successful testing find the most serious failure**
- ✓ In most cases, testing will find all defects of the system (see principle 2), but the most serious defects should be found.
- ✓ **This alone does not prove the quality of the software.**
- ✓ The functionality of the software may not meet the needs and expectations of the users.
- ✓ You can not test quality into the product, it must be built in from the very beginning!

I-Fundamentals of Testing

02- The seven general principles of software testing

Summary:

- ✓ **Tests** can help finding defects in the software, however; they can not give proof of the **absence of defects**
- ✓ For non-trivial system, **exhaustive testing** is impossible, sample testing necessary.
- ✓ **Early testing** helps reduce costs because defects discovered early on are fixed with less effort
- ✓ Defects show up **clustered**. Finding a defect at one place will mean you probably find another defect nearby.
- ✓ **Repeating** identical tests produces no new information
- ✓ Each **particular environmental** determines the way in which tests will run
- ✓ Error-free software does not imply **suitability for use.**

I-Fundamentals of Testing

03-Fundamental of Test process

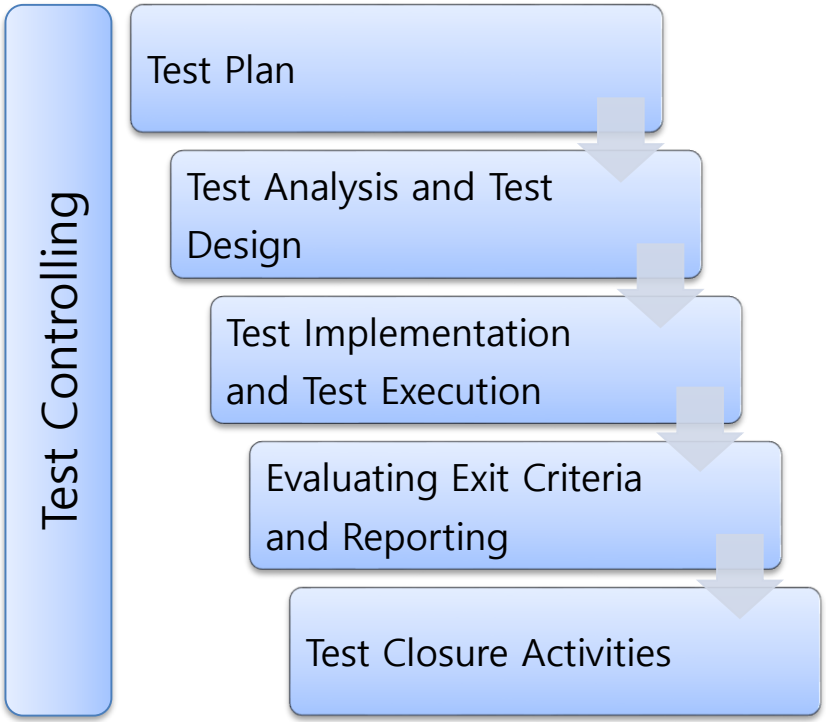
Testing as a process within the SW development process

- Depending on the approach chosen, testing will take place at different points within the development process
- Testing is a process itself
- The testing process is determined by the following phases
  - **Test planning**
  - **Test analysis and test design**
  - **Test implementation and test execution**
  - **Evaluating Exit Criteria and reporting**
  - **Test closure activities**
- as well as
  - **Test Controlling (at all phases)**
- Test phases may overlap

I-Fundamentals of Testing
03-Fundamental of Test process

Testing throughout the software development process

- ✓ Testing is more than test execution!
- ✓ Includes **overlapping** and **backtracking**
- ✓ Each phase of the **testing process** takes p lace **concurrent** to the phase of the softw are development process

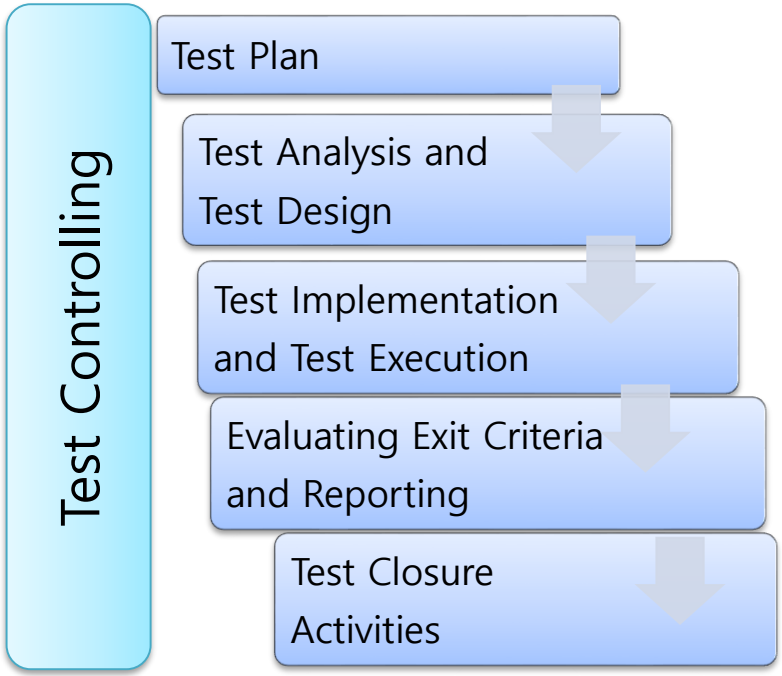


I-Fundamentals of Testing
03-Fundamental of Test process

Test control-main tasks

Test control is an on going activity **influencing** test plan ning. The test plan May be modified according to the info rmation Acquired from best controlling

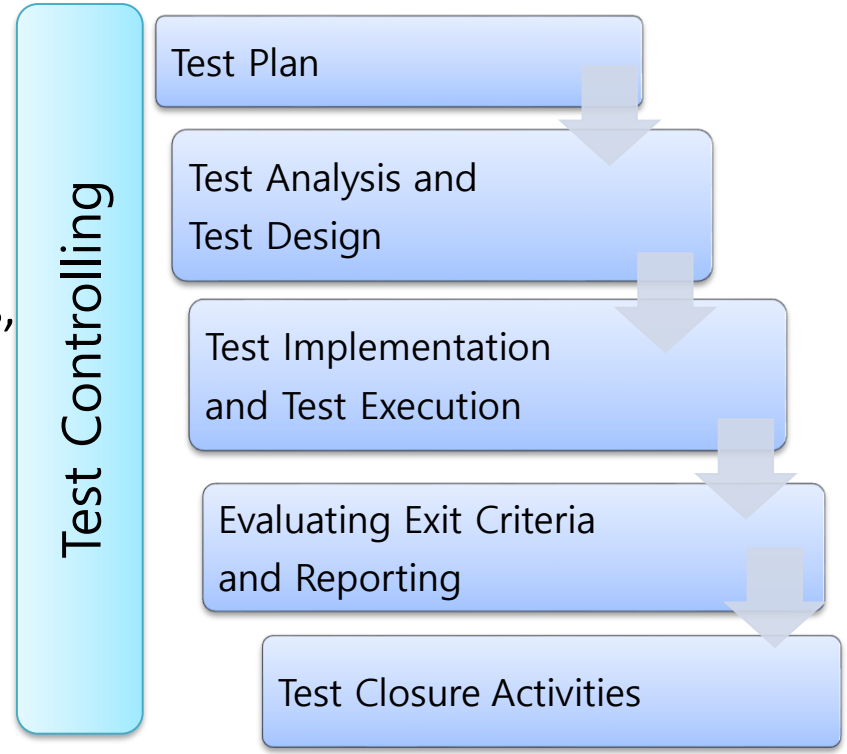
- The status of the test process is determined by compar ing the **progress** achieved against the last plan. Neces sary activities will be started accordingly.
- **Measure and analyze** results
- The test **progress** test **coverage** and the **exit criteria** are monitored and documented
- Start **correcting measures**
- Prepare and make **decisions**



I-Fundamentals of Testing
03-Fundamental of Test process

Test Planning-main tasks

- Determining the scope and risk
- Identifying the objectives of testing and exit criteria
- Determining the approach: test techniques, test coverage, testing teams
- Implement testing method/test strategy, plan time span for actives following
- Acquiring and scheduling test resources: people, test environment, test budget



I-Fundamentals of Testing

03-Fundamental of Test process

Test plan(German: Testkonzept):

A document describing the scope, approach, resources and schedule of intended test activities. It includes, but is not limited to, the test items, the features to be tested, resources and contingency planning.

Test Strategy:

(1) a high level description of the test levels to be performed and the testing within those levels for an organization or program (one or more projects)

(2) according to the overall approach, the test efforts are divided among the test objects and the different test objectives: the choice of test methods, how and when the test activities should be do ne and when to stop testing (exit criteria)

Exit criteria (after Glib and Graham):

The set of generic specific conditions, agreed upon with the stakeholders, for permitting a proc ess to be officially completed. The purpose of exit criteria is to prevent a task from being consider ed completed when there are still parts of the task outstanding which have not been finished. **Exit criteria** are used to report against and to plan **when to stop testing**. This should **be done for each test level.**

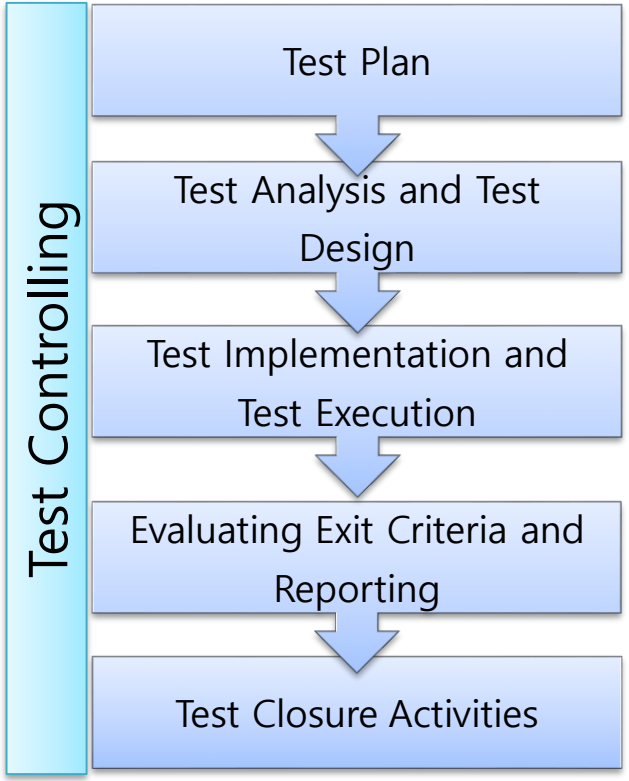
I-Fundamentals of Testing
03-Fundamental of Test process

Test analysis and Design-main tasks/1

- Reviewing the test basis (requirements, system architecture, design, interfaces)
  Analyze system architecture,
  system design including interfaces among test objects
- Identify specific test conditions and required test data
  evaluate the availability of test data and/or the feasibility of generating test data
- Designing the test/test cases
- Create and prioritize logical test causes (test causes witho ut specific values for test data)

✓ Positive tests give proof of the functionality, negative tests check the handling of error situations

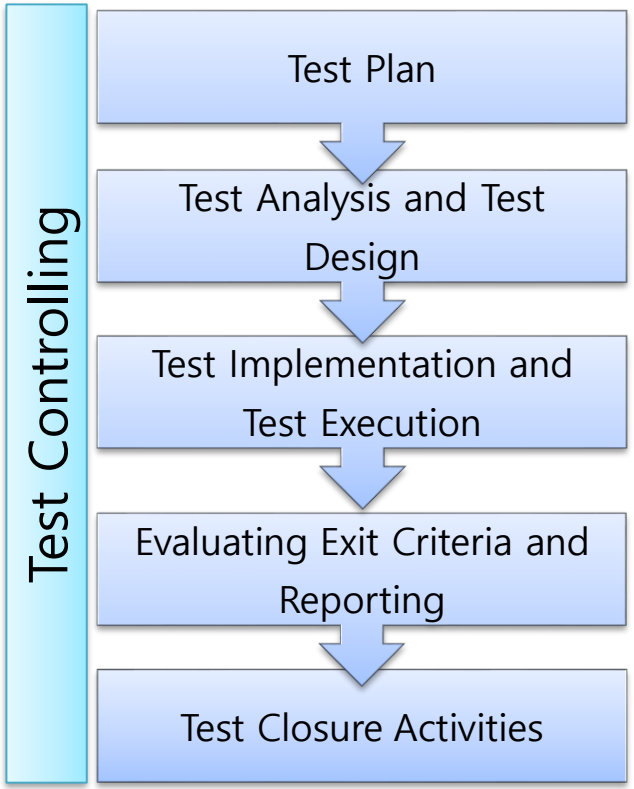
✓ Testability analysis (more about this following)



I-Fundamentals of Testing
03-Fundamental of Test process

Test Analysis and Design- main tasks/2

- Organizing the **test environment** (test bed)
  - (Exclusive) availability of the test environment, time windows, etc.
  - Define the operation of the test environment, including user administration
- Loading data sets and system parameters
- Connecting the test environment to adjacent systems
- Test infrastructure and **test tools,** if needed
  - Processes, procedures and responsibilities
  - choosing, provisioning, installation and operations of test tools



I-Fundamentals of Testing

03-Fundamental of Test process

Test data:

Data that exists in the system before a test is executed and affects or is affected by the component or system under test.

Input data:

A variable that is read by a component (whether stored within the system or outside)

Test coverage:

The degree of which a specified item has been exercised by a test suite (expressed as a percentage). Used mostly on white box tests to determine code coverage.

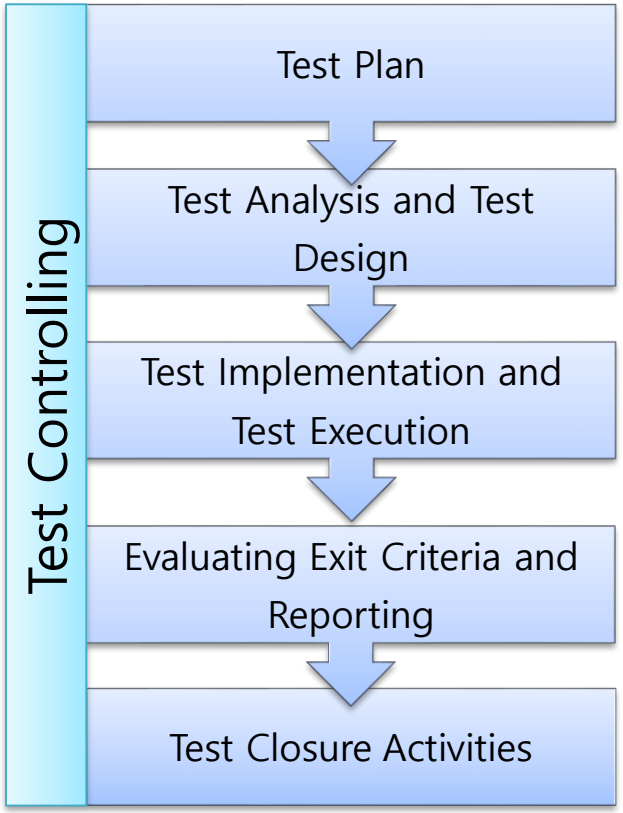
Test oracle:

A source to determine the expected results of the software under test: benchmarks (also the results of earlier tests). User’s manual or specialized knowledge. It should be the code.

I-Fundamentals of Testing
03-Fundamental of Test process

Test Implementation & Execution

- **developing and prioritizing** test cases
  - creating test data , writing test procedure
  - creating test sequences (test suites)
- creating test **automation scripts**, if necessary
- configuring the **test environment(test bed)**
- **executing** test(manually or automatically)
  - follow test sequence state in the test
- plan(test suites, order of test cases)
- test **result recording** and analysis
- retest(after defect correction)
- regression test
  - ensure that changes(after installing a new release, or error fixing) did not uncover other or **introduce new defects**.



I-Fundamentals of Testing
03-Fundamental of Test process

Test suite/test sequence

- a set of several test cases for a component or system , where post condition of one test is u sed as the precondition foe the next one

Test procedure specification(test scenario)

- a document specifying a sequence of action for the execution of a test. Also known as test script or manual test script.(After IEEE 829)

Test execution

- The process of running a test, producing actual results.

Test log (test protocol, test report)

- A chronological record of relevant details about the execution of tests:

when the test was done, what result was produced.

Regression tests:

- tasting of a previously tasted program following modification of ensure that defected have not been introduced or uncovered in unchanged areas of the software , as a result of the ch anges made. It is performed when the software or its environment is changed.

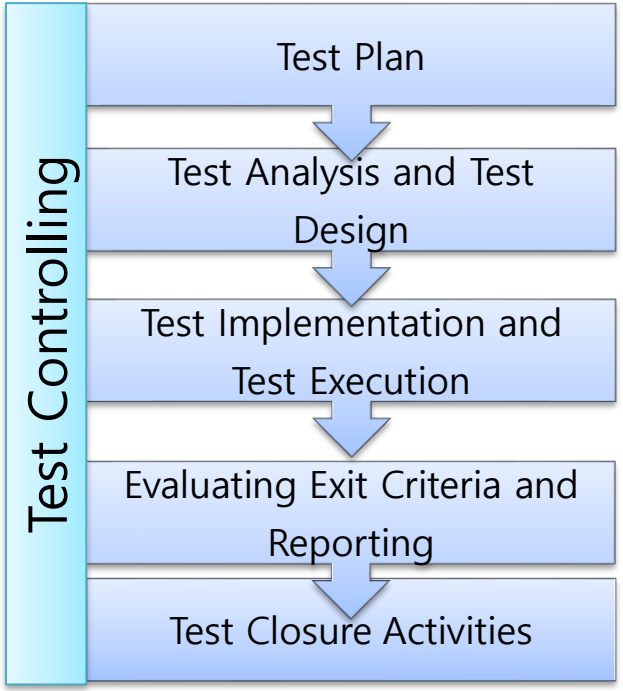
Confirmation testing retest:

- repeating a test after a defect has been fixed in order to confirm that the original defect has been successfully removed

I-Fundamentals of Testing
03-Fundamental of Test process

Evaluating Exit Criteria-main tasks

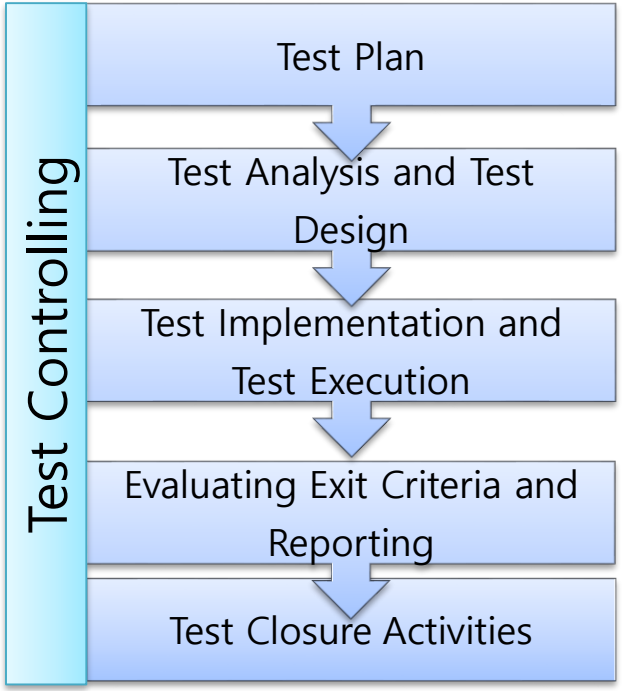
- Assessing test execution against the defined objectives (e.g. test and criteria)
- Evaluating **test logs** (summary of test activities, test result , communication exit criteria)
- Provide information to allow the decision, whether **more test** should take place



I-Fundamentals of Testing
03-Fundamental of Test process

Test Closure Activities - main task

- collection data from completed test activities to consolidate experience , test ware , facts and numbers.
- **Closure of incident reports** or raising change requests for any remaining open points
- Checking which **planned deliverables** have been delivered and tested.
- Documenting the acceptance of the system
- Finalizing and **archiving test ware,** the test environment and the test infrastructure for later reuse, hand over to operations
- Analyzing "**lessons learned**" for future project



I-Fundamentals of Testing
04-The psychology of testing

Roles and responsibilities

Developer role	Tester role
Implements requirements	Plans testing activities
Developers structures	Design test case
Design and programs the software	Is concerned only with finding defects
Creating a product is his success	Finding an error made by a developer is his success

Perception:

Wrong!

Testing is a constructive activity as well,

It aims eliminating defects from a product !

Personal attributes of a good tester /1

Curious , perceptive attentive to detail- not all error show up great manor

- To **comprehend** the practical scenarios of the **customer**
- To be able to analysis the structure of the test
- To **discover details**, where failure might show

Skepticism and has a critical eye

- Test object contain defects- you just have to find them
- Do not believed everything you are told by the developers
- One must not get frightened by the fact that serious defects may often be found which will have impact on the course of the project.

I-Fundamentals of Testing
04-The psychology of testing

Personal attributes of a good tester /2

- Good communication skills
  - ✓ To bring **bad news** to the developers
  - ✓ To overbear frustration state of minds
  - ✓ Both technical as well as issue of the practical use of the system must be understood and communicated
  - ✓ Positive communication can help to avoid or to ease difficult situations.
  - ✓ To quickly establish a working relationship with the developers

    Experiences
  - ✓ **Personal factors** influencing error occurrence
  - ✓ Experience helps identifying where **errors** might **accumulate**

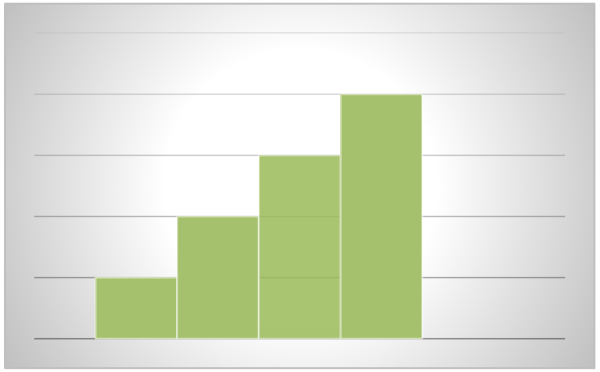
I-Fundamentals of Testing
04-The psychology of testing

Differences: to design- to develop – to test

- Testing requires a different mindset from designing developing new computer systems
  - **Common goal**: to provide good software
  - **Design** mission: help the customer to supply the right requirements
  - **Developer's** mission: convert the **requirements** into functions
  - **Tester's** mission: examine the **correct** implementation of the customer's requirements
- In principle, one person can be given all three roles to work at.
  - Differences in goal and role models must be taken into account
  - This is difficult but possible
  - Other solution (independent testers) are often easier and produce better results

Independent testing

- The separation of testing responsibilities support the independent evaluation of test results.
- The diagram below show the degree of independent as a bar chart.



I-Fundamentals of Testing
04-The psychology of testing

Types of test organization /1

- Developer test
  - The developer will never examine his ”creation” unbiased(**emotional attachment**)
  - Extra costs result for the orientation of other person on the test object
- Human being tend to **overlook their own faults.**
  - The developer run the risks of not recognizing even self-evident defects.
- Error made because of **misinterpretation** of the requirements will remain undetected.
  - Setting up test teams where developers test each other’s products helps to avoid or at least lessen this short coming.

I-Fundamentals of Testing
04-The psychology of testing

Types of test organization /2

- Teams of developers
  - Developers speak the same language
  - Costs for orientation in the test object are kept moderate especially when the teams exchange test objects.
  - Danger of generation of conflicts among developing teams
    - One developer who looks for and finds a defect will not be the other developer's best friend
  - Mingling development and test activities
    - Frequent switching of ways of thinking
    - Makes difficult to control project budget

Types of test organization /3

❑Test teams

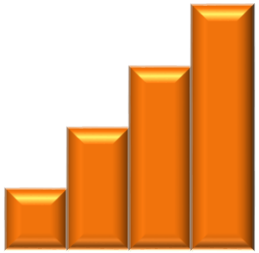
- Creating test teams converting different project areas enhances the quality of testing.
- It is important that test teams of different areas in the project work independently

I-Fundamentals of Testing
04-The psychology of testing

Types of test organization /4

❑ Outsourcing tests

- The separation of testing activities and development activities offers best independence between test object and tester.
- Outsourced test activities are performed by persons having relatively little knowledge about the test object and the project background
  - **Learning curve** bring high costs, therefore unbiased party experts should be involved at the early stages of the project
- External expert have a high level of **testing know how**:
  - An appropriate test design is ensured
  - Methods and tools find optimal



❑ Designing test cases automatically

- Computer aided generation of test cases, e.g. based on the formal specification documents, is also independent

I-Fundamentals of Testing
04-The psychology of testing

Difficulties /1

- **Unable to understand** each other
  - Developers should have basic knowledge of testing
  - Tester should have basic knowledge of software development
- Especially in **stress situations**, discovering errors that someone has made often leads to conflicts.
  - The way of document defects and the way of the defects is described will decide how the situation will develop.
  - Persons should not be criticized, the defects must be stated **faculty**
  - Defect description should help the developers find the error
  - **Common objectives** must always be the main issue.

I-Fundamentals of Testing
04-The psychology of testing

Difficulties /2

- Communication between tester and developers missing or insufficient. This can make impossible to work together.
  - Tester seen as “only messenger of **bed news** ”
  - Improvement: try to see yourself in the other person’s role. Did my message come through? Did the answer reach me?

- A solid test requires the appropriate **distance to the test object**
  - An independent and non-biased position is acquired through distance from the development
  - However, too large a distance between the test object and the development team will lead to more effort and time for testing.

I-Fundamentals of Testing
04-The psychology of testing

Summary

- People make **mistakes**, every implementation has defects.
- Human nature makes it difficult to stand in front of one's own defects(**error blindness**)
- Developer and tester means two different worlds meet each other.
  - **Developing is constructive**- something is created that was not there before
  - **Testing** seems destructive at first glance-defect will found
  - Together , **development and test** are constructive in their objective to ensure software with the least defects possible.
- **Independent testing enhances quality of testing**:

  instead of developer, use tester teams and teams with external personnel for testing.

Sffdsadf
Asdfs