Chapter 1: Fundamentals of Testing

**Outline**

* Understanding software testing
* The seven general principles of software testing
* Fundamental test process
* The psychology of testing

**Understanding Software Testing**

* **What is Software Testing?**
  + Software Testing is the process of executing a program or system with the intent of finding errors. It evaluates a system\software or its component(s) with the intent to find whether it satisfies the specified requirements or not.
  + Testing can have the following objectives:
    - * 1. Finding defects
        2. Gaining confidence about the level of quality
        3. Providing information for decision making
        4. Preventing defects
* **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
* **What is Software Quality?**
* **Software Quality (as per ISO/ IEC 9126):** 
  + The totality of functionality and features of a software product that contribute to its ability to satisfy stated or implied needs.
* **Software Quality (as IEEE Std 610):**
  + The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations.
* According to ISO/IEC 9126 **software quality consists of:**
  + **Functionality** 
    - Functional means correctness & completeness
  + **Reliability**
    - maturity, fault tolerance, recovery after failure
  + **Usability**
    - learn-ability, understandability, attractiveness
  + **Efficiency** 
    - min use of resource
  + **Maintainability**
    - Verifiability, changeability
  + **Portability**
    - Transfer, easy to install
* **Software Testing VS Quality**
* Testing helps us to measure the quality of software in terms of the number of defects found, the tests run, and the system covered by the tests.
* **Quality Assurance –** includes activities that ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.
  + Focuses on processes and procedures rather than conducting actual testing on the system.
  + Process- oriented
  + Preventive activities.
* **Quality Control –** includes activities that ensure the verification of a developed software with respect to documented (or not in some cases) requirements.
  + Focuses on actual testing by executing the software with an aim to identify bug/defect through implementation of procedures and process.
  + Product-oriented
  + Corrective process
* **Test goals:**
  1. **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
  2. **Poor of functionality**

System functionality should be implemented as specified

* 1. **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.
  2. **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.
* **Testing and Debugging**
* **Testing** : It involves identifying bug/error/defect in a software without correcting it. Normally professionals with a quality assurance background are involved in bugs identification. Testing is performed in the testing phase.
* **Debugging** : It involves identifying, isolating, and fixing the problems/bugs. Developers who code the software conduct debugging upon encountering an error in the code. Debugging is a part of White Box Testing or Unit Testing. Debugging can be performed in the development phase while conducting Unit Testing or in phases while fixing the reported bugs.

**The seven general principles of software testing**

**Software Testing Principles:**

**Principle 1: A necessary part of a test case is a definition of the expected output or result**- A test case must consist of two components:

* + 1. A description of the input data to the program
    2. A precise description of the correct output of the program for the set of input data.

**Principle 2: A Programmer should avoid attempting to test his or her own program**

-After a programmer has constructively designed and coded a program it is extremely difficult to suddenly change perspective to look at the program with a destructive eye.

**Principle 3: Test case must be written for input conditions that are invalid and unexpected, as well as for those that are valid and expected.**

**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: 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

**Principle 6: 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 and 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  
specification and design.

**Principle 7: Testing is content 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.

**Fundamental Test Process:**

**Software Testing Life Cycle (STLC):**

**Testing as a process within the software 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**
  + **Test Controlling (at all phases)**
* Each phase of the testing process takes place

concurrent to the phase of the software development process

1. **Test control-main tasks**

* Test control is an on going activity influencing test planning. The test plan may be modified according to the information  
  acquired from best controlling
  + The status of the test process is determined by

comparing the progress achieved against the last

plan. Necessary 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

1. **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
* **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:** 
  + 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)
  + 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 done 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 process to be officially completed. The purpose of exit criteria is to prevent a task from being considered 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.**

1. **Test analysis and Design**
   * 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 without 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)
   * 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
   * **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.
2. **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**.
* **Test suite/test sequence**  
  – a set of several test cases for a component or system , where post condition of one test is used 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 changes 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

1. **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
2. **Test Closure Activities**

* 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

**The Psychology of Testing:**

* The Success of testing is influenced by psychological factors:
  + clear objectives
  + a balance of self-testing and independent testing
  + recognition of courteous communication and feedback on defects.
* Contrast the mindset of a tester and that of a developer.
* Self-testing: While testing their own code they find many problems so the programmers, architect and the developers always test their own code before giving it to anyone. However we all know that it is difficult to find our own mistakes. So, programmers, architect, business analyst depend on others to help test their work.
* Independent testing
  + The separation of testing responsibilities support the independent evaluation of test results.
  + Independent testing enhances quality of testing:
    - instead of developer, use tester teams and teams with external personnel  
      for testing.