UNIT 5 : System Design

5.1 Testing Fundamentals and principals.
5.2 Types of Testing.
- 5.2.1 Black Box & White Box
- 5.2.2 Unit Testing
- 5.2.3 Integration Testing
- 5.2.4 System Testing

5.3 Introduction to change Over
- 5.3.1 Types of change over

5.1 Software Testing Fundamental

- Software testing is the process of establishing confidence that a program or system does what it is supposed to.
- Testing is the process of executing a program or system with the intent of finding error.

- Software testing can be stated as the process of validating and verifying that a computer program/application/product:
  - Meets the requirements that guided its design and development,
  - Works as expected,
  - Can be implemented with the same characteristics,
  - Satisfies the needs of stakeholders.

PURPOSE OF TESTING

- To improve quality by make software error free.
- Testing is about verifying that what was specified is what was delivered: it verifies that the product (system) meets the functional, performance, design, and implementation requirements identified in the procurement specifications.

- The testing program is used to identify when the work has been “completed” so that the contract can be closed, the vendor paid.

ADVANTAGES

- Helps to improve the quality, reliability & performance of the system.
- Check what all functions software supposed to do & also check that Software is not doing what he not supposed to do.
- Help to identify the defects in the early stage & try to avoid the cost of fixing the bugs.

- Improve software quality by making software defect free.
- To ensure that product works as user expected.

DISADVANTAGES

- Require development time to test software internally and externally .
- Require skill developer for white box testing
- Increase cost of software development.
- Require more time to release software to the customer.
- Need various tool for white box testing.

TESTING PRINCIPLES

- **Testing shows presence of errors:**
  - Testing an application can only reveal that one or more defects exist in the application, however, testing alone cannot prove that the application is error free. Therefore, it is important to design test cases which find as many defects as possible. of existing, not discovered error conditions within the test object.

- **Exhaustive testing is not possible:**
  - Unless the application under test (UAT) has a very simple logical structure and limited input, it is not possible to test all possible combinations of data and scenarios. For this reason, risk and priorities are used to concentrate on the most important aspects to test.

- **Test early and regularly:**
  - Early testing helps detecting errors at an early stage of the development process which simplifies error correction.

- **Fading effectiveness**
  - The effectiveness of tests fades over time. If test-cases are only repeated, they do not expose new errors. Errors, remaining within untested functions may not be discovered. In order to prevent this effect, test-cases must be altered and reworked time by time.

- **Testing depends on context**
  - No two systems are the same and therefore can not be tested the same way. Testing intensity, the definition of end criteria etc. must be defined individually for each system depending on its testing context.

- □ **Testing Schedule.**
  - An overall testing schedule and resource planning must be made well in advance. Hence all test can be planned and designed before any code has been generated

- **The PARETO (80-20 Rule) Principle.**
  - This principle says that 20% of the problems lead to 80% of other problems. Hence, in order to concentrating on solving 80% of the problems rather, one can concentrate to solve 20 % of the problems which saves lot of troubles.
  - Applying Pareto principle to software testing, it would be correct to say that 80% of errors being concentrated in 20% of the developed product functionality.

5.2 Testing Methods/Types

- **Black Box Testing:**
  - This method is named Black box because the software program, in the eyes of the tester, is like a black box; inside which one cannot see.
  - It is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester.

- Test design techniques include:
    - Equivalence partitioning
    - Boundary Value Analysis
    - Cause Effect Graphing
- Also known as : functional testing.

- **White Box Testing:**
  - It is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester.
- Test design techniques include:
  - Control flow testing
  - Data flow testing
  - Branch testing
  - Path testing
  - Code coverage testing.

- Also known as : clear box testing, glass box testing, structural testing.

□ Unit Testing:

- It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.
- Example:

a) In a program we are checking if loop, method or function is working fine

b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization

□ Integration Testing:

- The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output.
- Integration testing is of four types:

(i) Top-down

(ii) Bottom-up

(iii) Sandwich

(iv) Big-Bang

- Example:
  (a) Black Box testing:- It is used for validation. In this we ignore internal working mechanism and focused on what is the output?.
  (b) White Box testing:- It is used for verification. In this we focus on internal mechanism i.e. how the output is achieved?

System Testing:

- This software is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.
- In this, we have security testing, recovery testing, stress testing, and performance testing
- Example:
  - This include functional as well as non functional testing

5.3 Introduction to Change Over

- □As technologies change, many businesses find themselves needing to change over their computer information systems.
- □Upgrading these systems helps them optimize their efficiency and remain competitive.
- □Common changeover areas include security systems, database systems, accounting systems and managerial information systems.
- □Deciding which changeover technique will work best for a particular company depends on the type of changeover and degree of risk for the company.

5.3.1 Types of change over
1.Data Conversion / Change over

- It is a Important part of system installation process.
- During data conversion, existing data is loaded into new system.
- It can be done before, after or during operational environment is complete.
- We should develop data conversion plan as early as possible and the conversion process should be tested when tested environment is developed.

- □ When new system replace an existing system, you should automate the data conversion process.
- □ The old system might be capable of exporting data in an acceptable format of the new system.
- □ If a standard format is not available, you must develop a program to extract the data and convert it into acceptable format.
- □ Data conversion is more difficult when the new system replace manual system because all data must be entered manually.

2. Application / System Changeover

- □ Definition: Changeover or Conversion is the process of changing from the old system, which is currently running in the organization, to the newly built system.

- There are four methods of handling the system conversion which are:-
    1. Parallel system method.
    2. Dual system method.
    3. Direct cutover method.
    4. Pilot Approach method

3. Parallel Changeover

- It require both old and new system operate fully for a specified period.
- Data is input in both system and output generated by new system is compared with the equivalent output from the old system.
- When user, management and IT group are satisfied that the new system operates correctly , old system is terminated.

Advantages

- □ Lower risk: If the new system does not work correctly, company can use old system as backup .
- □ Easier to verify new system: Output of new system is compare with old system and verified during parallel operation.

Disadvantages

- Costly: Company pay for both system because both system are in full operation.
- Processing Delay: Running both system might place a burden on operating environment and cause processing delay.
- Work and data are duplicated.
- Time consuming.

4.Direct Changeover

- Direct changeover, also referred to as immediate replacement, tends to be the least favorite of the changeover techniques.
- In a direct changeover, the entire system is replaced in an instant.
- This involves taking the old system offline and putting the new system online within a day or over a weekend or holiday period.
- It is least expensive because IT group has to operate and maintain only one system at a time.

- There are no parallel activities and there is on falling back to old system.
- It is only choice if the operating environment cannot support both the old and new system or if the old and new system are incompatible.
- Most organization use direct cutover only for noncritical situations.
- Timing is important when using direct changeover strategy.

Advantages

- Minimal Cost : IT group has to operate and maintain only one system at a time.
- Minimize Workload

Disadvantage

- More Risk : Because if Changeover fail stop entire organization.
- Require careful planning: If something goes wrong, reverting back to the old system usually is impossible.

5. Phase In / Dual Changeover

- The phased changeover technique is considered a compromise between parallel and direct changeovers.
- In a phased changeover, the new system is implemented one stage at a time.
- Phased operation works in different phases or stages.
- Implementation of new system in modules or stages is phased operation.

- In this method, new system is implemented in many phases over a period of time. So, the old system is gradually phased out while the new one is being phased in.
- In phase operation the risk of errors or failures is limited to the implemented module only and also phased operation is less expensive than the full parallel operation.

- The actually conversion from the old parts of the system to the new parts may be either parallel or direct.
- As an example, consider a company working toward installing a new financial system. Implementing the new system one department at a time, the company converts accounts receivable, accounts payable, payroll, and so on .
- This method is used when it is not possible to install a system throughout an organization all at once.

ADVANTAGE

- Less risky because implemented in phase so no danger of total breakdown.
- Any problem should be in one area other area operations are unaffected.

DISADVANTAGE

- Can take long time to achieve total changeover.
- Interface between parts of the system may make this impractical.

6. Pilot Changeover

- In the Pilot Approach method, a complete new version of the system is implemented in just one location or Site of the organization.
- For example, a bank may first test the system at one of its branches. This branch is referred to as the pilot, or beta, site for the program.

- The old system continues to operate for the entire organization including the pilot site.
- After the system proves successful at the pilot site, it is implemented in the rest of the organization, usually using direct cutover method.
- Pilot operation is combination of parallel operation and direct cutover methods.

ADVANTAGE:

- Less risky
- Less costly

DISADVANTAGE

- Can take a long time to achieve total changeover.