UNIT-III

Performance Testing:

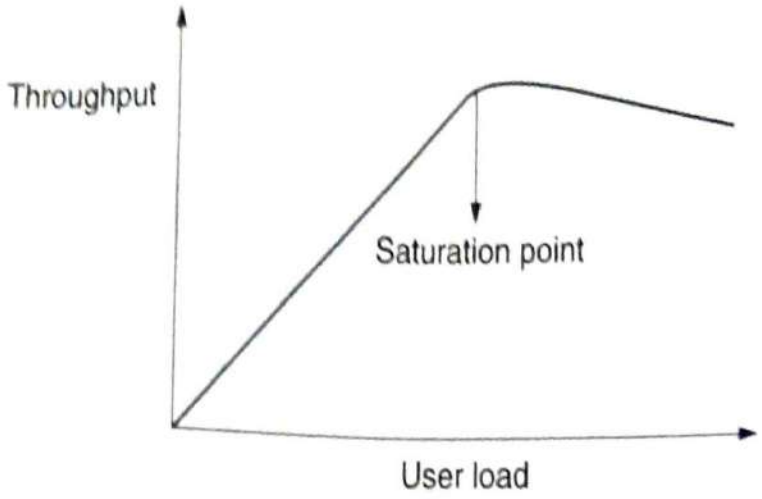
- Performance Testing is a type of software testing that ensures software applications to perform properly under their expected workload. It is a testing technique carried out to determine system performance in terms of sensitivity, reactivity and stability under a particular workload.

- The performance of an application is measured from different perspectives to improve scalability and performance of the application. Load testing, stress testing, and volume testing are some of the types of performance testing that are normally one during the application development stage to ensure that the application performs at the expected level in production.

- The testing performed to evaluate the response time, throughput, and utilization of the system, to execute its required functions in comparison with different with different versions of the same product or a different competitive product(s) is called performance testing.

Factors Governing Performance Testing:

- ✓ The capability of the system or the product in handling multiple transactions is determined by a factor called **throughput.**
- ✓ Throughput represents the number of requests/business transactions processed by the product in specified time duration.



- ➢ The load to the product can be increased by increasing the number of users or by increasing the number of concurrent operations of the product.
- ➢ Initially the throughput keeps increasing as the user **load** increases. This is the ideal situation for any product and indicates that the product is capable of delivering more when there are more users trying to use the product. Throughput represents how many business transactions can be serviced in a given duration for a given load.
- ➢ Hence measuring "**response time**" becomes an important activity of performance testing.
- ➢ Response time can be defined as the delay between the point of request and the first response from the product.
- ➢ Another factor is latency. Latency is a delay caused by the application, operating system, and by the environment that are calculated separately.

- The next factor that governs the performance testing is tuning.
- Tuning is a procedure by which the product performance is enhanced by setting different values to the parameters (variables) of the product, operating system, and other components. Tuning improves the product performance without having to touch the source code of the product.
- Yet another factor needs to be considered for performance testing is performance of competitive products.
- One of the most important factors that affect performance testing is the availability of resources.

Methodology for performance testing:

Performance testing is complex and expensive due to large resource requirements and the time it takes.

1. Collecting requirements.
2. Writing testcases.
3. Automating performance testcases.
4. Executing performance testcases.
5. Analyzing performance test results.
6. Performance tuning.
7. Performance benchmarking.
8. Recommending right configuration for the customers.

1. Collecting Requirements:

Collecting requirement is the first step in planning the performance testing; typically, functionality testing has a definite set of inputs and outputs, with a clear definition of expected results.

There are several sources for deriving performance requirements. Some of them are as follows.

- A performance compared to the previous release of the same product.
- Performance compared to the competitive product(s).
- Performance compared to absolute numbers derived from actual need.
- Performance numbers derived from architecture and design.

2. Writing Test cases:

- List of operations or business transactions to be tested.
- Steps for executing those operations/transactions.
- List of products, OS parameters that impact the performance testing and their values.
- Loading pattern.
- Resource and their configuration.
- The expected results.
- The product versions/competitive products to be compared with and related information such as their corresponding fields.

3. Automating performance test cases:

- Performance testing is repetitive.
- Performance test cases cannot be effective without automation and in most cases, it is, in fact almost impossible to do performance testing without automation.
- The results of performance testing need to be accurate and manually calculating the response time, throughput and soon.
- Performance testing takes into account several factors. There are far too many permutations and combination of those factors and it will be difficult to remember all these and use them if the tests are done manually.

4. Executing Performance test cases:

- The most effort consuming aspect in execution is usually data collection. Data corresponding to the following points.
- Start and end time of test case execution.
- Log and trace/audit files of the product and operating system.

- Utilization of resources (CPU, memory, disk)
- The response time, throughput, latency and soon.

5. Analyzing the performance test results:

- Analyzing the performance test results require multi-dimensional thinking. This is the most complex part of performance testing where product knowledge, analytical thinking and statistical background are all absolutely essential.
- Before analyzing the data some calculations of data and organization of the data are required.
- Calculating the mean of the performance test result data.
- Calculate the standard deviation
- Removing the noise.

6. Tools for Performance Testing:

- There are two types of tools that can be used for performance testing that is functional performance tools and load tools.
- Functional performance tools help in recording and playing back the transactions and obtaining performance numbers. This test generally involves very few machines.
- Load testing tools simulate the load condition for performance testing without having to keep that many users or machines. The load testing tools simplify the complexities involved in creating the load and without such load tools it may be impossible to perform these kinds of tests.

Some popular performance tools:

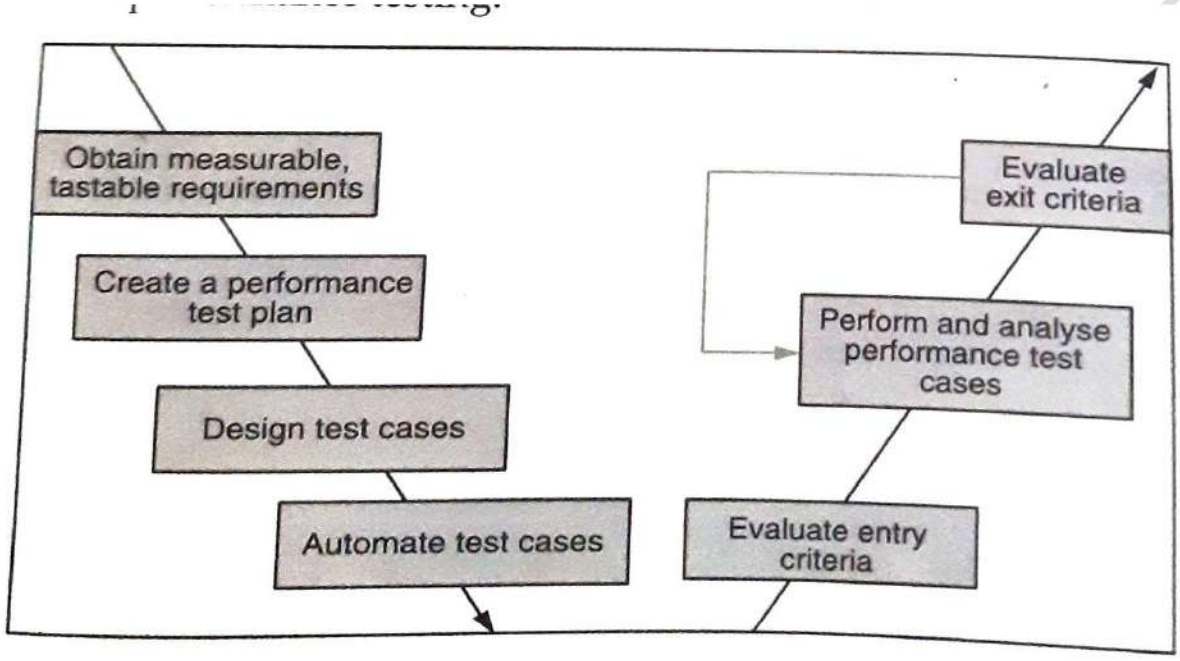
- Win Runner from Mercury
- QA Partner from Compuware.
- Silk test from Segue.

Load Testing tools:

- Load runner from mercury.

- QA Load from Compuware.
- Silk performer from Segue.

Process for Performance Testing:



- Performance testing follows the same process as any other testing type. The only difference is in getting more details and analysis.
- Ever changing requirements for performance is a serious threat to the product as performance can only be improved marginally by fixing it in the code. As mentioned earlier, a majority of the performance issues require work or changes in architecture and design.
- Hence, it is important to collect the requirements for performance earlier in the life cycle and address them, because changes to architecture and design late in the cycle are very expensive.
- While collecting requirements for performance testing, it is important to decide whether they are testable, that is, to ensure that the performance requirements are quantified and validated in an objective way.
- Performance testing is to create a performance test plan.

✓ Resource requirements:

All additional resources that are specifically needed for performance testing need to be planned and obtained. Normally these resources are obtained, used for performance test, and released after performance testing is over.

✓ Test bed (simulated and real-life)

The test tab with all required equipment and software configuration has to be set up prior to execution. Performance testing requires a large number of resources and requires special configurations.

✓ Responsibilities

Performance defects, as explained earlier may cause changes to architecture, design, and code. Additionally, the teams facing the customers normally communicate requirements for performance.

✓ Setting up product traces, audits, and traces, and traces (external and internal)

Performance test results need to be associated with traces and audit trails to analyses the results and defects.

✓ Entry and exit criteria:

Performance tests require a stable product due to its complexity and the accuracy that is needed. Changes to the product affect performance numbers and may mean that the tests have to be repeated. It will be counter-productive to execute performance test cases before the product is stable or when changes are being made.

Entry and exit criteria play a major role in the process of performance test execution. At regular intervals during product development, the entry criteria are evaluated and the test is started if those criteria are met. There can be separate set of criteria for each of the performance testcases.

Regression Testing

Definition

When any modification or changes are done to the application or even when any small change is done to the code then it can bring unexpected issues. Along with the new changes it becomes very important to test whether the existing functionality is intact or not. This can be achieved by doing the regression testing.

OR

Regression testing is done to ensure that enhancements or defect fixes made to the software works properly and does not affect the existing functionality.

Example:

Let's assume that there is an application which maintains the details of all the students in school. This application has four buttons Add, Save, Delete and Refresh. All the buttons' functionalities are working as expected.

Recently a new button „Update" is added in the application. This „Update" button functionality is tested and confirmed that its working as expected. Butat the same time it becomes very important to know that the introduction of this new button should not impact the other existing buttons functionality.

Along with the „Update" button all the other buttons functionality are tested in order to find any new issues in the existing code. This process is known as regression testing.

Types of Regression Testing

Majorly there are two types of regression testing in practice.

1. Regular regression testing
2. Final regression testing

1. Regular regression testing:

A regular regression testing is done between test cycles to ensure that the defect fixes that are done and the functionality that were working with the earlier test cycles continue to work. A regular regression testing can use more than one product build for the test cases to be executed.

2. Final regression testing:

A final regression testing is done to validate the final build before release. The final regression test cycle is conducted for a specific period of duration, which is mutually agreed upon between the development and testing teams.

Progressive Regression Testing:

Progressive regression testing is used when the modifications are done in the specifications and new test cases are designed.

Retest-All Strategy:

The retest-all strategy is very tedious and time consuming because here we reuse all tests which results in the execution of unnecessary test cases. When any small modification or change is done to the application then this strategy is not useful.

Selective Strategy:

In selective strategy we use a subset of the existing test cases to cut down the retesting effort and cost. If any changes are done to the program entities, e.g. functions, variables etc., and then a test unit must be rerun. Here the difficult part is to find out the dependencies between a test case and the program entities it covers.

When to do regression testing?

- ✓ A reasonable amount of initial testing is already carried out.
- ✓ A good number of defects have been fixed.
- ✓ Defect fixes that can produce side-effects are taken care of.
- ✓ Any new feature is added.
- ✓ Any enhancement is done.
- ✓ Any bug is fixed.
- ✓ Any performance related issue is fixed.

Advantages of regression testing:

- It helps us to make sure that any changes like bug fixes or any enhancements to the module or application have not impacted the existing tested code.
- It ensures that the bugs found earlier are NOT creatable.
- Regression testing can be done by using the automation tools
- It helps in improving the quality of the product.

Disadvantages of regression testing:

- If regression testing is done without using automated tools then it can be very tedious and time consuming because here, we execute the same set of test cases again and again.
- Regression test is required even when a very small change is done in the code because this small modification can bring unexpected issues in the existing functionality.

How to Do regression testing:

The failure of regression can only be found very late in the cycle or found by the customers. Having a well-defined methodology for regression can prevent such costly misses.

1. **Performing an initial "Smoke" or "Sanity" test.**
2. **Understanding the criteria for selecting the testcases.**
3. **Classifying the test cases into different priorities**
4. **A methodology for selecting testcases.**
5. **Resetting the test cases for test execution.**
6. **Concluding the results of a regression cycle.**

1. Performing an initial "Smoke" or "Sanity" test:

- Identifying the basic functionality that a product must satisfy.
- Designing the cases to ensure that these basic functionality work and packaging them into a smoke test suite.
- Ensuring that every time a product is built, this suite is run successfully before anything else is run;

2. Understanding the criteria for selecting the test cases:

- The defect fixes and changes made in the current build.
- The ways to test the current changes.
- The ways of testing the other impacted parts.
- Regression testing should focus more on the impact of defect fixes than on the criticality of the defect itself.

3. Classifying test cases

- It is important to know the relative priority of test cases for a successful test execution.

4. Methodologies for selecting test cases

- Once test cases are classified into different priorities, the test cases can be selected. There could be several right approaches to regression testing which need to be decided on "case to case" basis.

5. Resetting the test cases for regression testing

- When there is a major change in the product.
- When there is a change in the build procedure which affects the product.
- Large release cycle where some test cases were not executed for a long time.

6. Concluding the results of regression testing

- Everyone monitors regression test results as this testing not only indicates about defects and but also their fixes.

Usability and accessibility Testing:

Usability testing attempts to characterize the "look and feel" and usage aspects of a product, from the point of view of users.

- Usability testing tests the product from the user's point of view. It encompasses a range of techniques for identifying how users actually interact with and use the product.
- Usability testing is for checking the product to see if it is easy to use for the various categories of users.
- Usability testing is a process to identify discrepancies between the user interface of the product and the human user requirements, in terms of the aesthetics aspects.
- The testing that validates the ease of use, speed and aesthetics of the product from the user's point of view is called as usability testing.

Approach to Usability Testing:

The number of mouse clicks, number of sub-menus to navigate, number of keystrokes, number of commands to perform a task can all be measured and

checked as a part of usability testing. Obviously if there are too many mouse clicks or too many commands to execute a user task, then the product would not be considered easy to use.

Usability testing is not only for product binaries or executable. It also applies to documentation and other deliverables that are shipped along with a product. The release media should also be verified for usability. A right approach for usability is to test every artefact that impacts users-such as product binaries documentation, messages, media covering usage patterns through both graphical and command user interfaces, as applicable.

When to do usability testing:

The most appropriate way of ensuring usability is by performing the usability testing in two phases.

- First is design validation.
- Second is usability testing done as a part of component and integration testing phases of a test cycle.
- When planning for testing, the usability requirements should be planned in parallel, upfront in the development cycle, similar to any other type of testing.

Phases and activities of usability testing

- Planning
- Design
- Coding
- Usability Testing
- Usability design validation
- Include usability in planning.

Usability is a habit and behavior. Just like humans, the products are expected to behave differently and correctly with different users and to their expectations.

Tools for Usability:

Name of the tool	Purpose
JAWS	For testing accessibility of the product with some assistive technologies.
HTML Validator	To validate the HTML source file for usability and accessibility standards.
Style sheet validator	To validate the style sheets for usability standards set by W3C.
Narrator	Narrator is a tool that reads the information displayed on the screen and creates audio descriptions for vision challenged users.
Soft Keyboard	Soft keyboard enables the use of pointing devices to use the keyboard by displaying the keyboard template on the screen.
Magnifier	Accessibility tool for vision challenged.

When to do usability testing:

- Comprehensibility.
- Consistency.
- Navigation.
- Responsiveness.

Comprehensibility:

The product should have simple and logical structure of features and documentation. They should be grouped on the basis of user scenarios and usage. The most frequent operations that are performed early in a scenario should be presented first, using the user interface.

Consistency:

A product needs to be consistent with any applicable standards, platform look- and-feel base infrastructure, and earlier version of the same product.

Example: If there are multiple products from the same company it would be worthwhile to have some consistency in the look-and-feel of these multiple products. Some services it irritates the users since they need to become comfortable with different templates and procedures for using each of them.

Navigation:

This helps in determining how easy it is to select the different operations of the product. An option that is buried very deep requires the users to travel to multiple screens or menu options to perform the operation.

Responsiveness:

How fast the product responds to the user request is another important aspect of usability. This should not be confused with performance testing. Screen navigation and visual displays should be almost immediate after the user selects an option or else it could give an impression to the user that there is no progress and cause him or her to keep trying the operation again.

Aesthetics Testing

It is also one of the important aspects in usability is making the product "beautiful" Performing aesthetics testing helps in improving usability further. This testing is important as many of the aesthetics related problems in the product from many organizations are ignored on the ground that they are not functional defects.

Example:
It's not possible for all products to measure up with the Taj Mahal for its beauty. Testing for aesthetics can at least ensure the product is pleasing to the eye.

Accessibility Testing:

Verifying the product usability for physically challenged users is called accessibility testing. There are large numbers of people who are challenged with

vision, hearing, and mobility related problems-partial or complete.

Accessibility to the product can be provided by two means.

- Making use of accessibility features provided by the underlying infrastructure.
- Providing accessibility in the product through standards and guidelines called product accessibility.

Basic Accessibility:

Basic accessibility is provided by the hardware and operating system. Keyboard accessibility:

A keyboard is the most complex device for vision- and mobility impaired users. Hence it received plenty of attention for accessibility.

Similarly, the operating system vendors came up with the some more improvements in the keyboard. Some of those improvements are usage of sticky keys, toggle keys, and arrow keys for mouse.

Sticky Keys:

To explain the sticky keys concept, let us take an example of <CTRL><ALT> <DEL>. One of the most complex sequences for vision impaired and mobility impaired users is <CTRL><ALT><DEL>. This keyboard sequence is used for various purposes such as log in, log out.

Filter Keys:

When keys are pressed for more than a particular duration, they are assumed to be repeated. Sometimes this troubles physically challenged users. Filter keys help in either stopping the repetition completely or slowing down the repetition.

Toggle key sound:

When toggle keys are enabled, the information typed may be different from what the user desires. For example, :<INS> key is a toggle.

Sound keys:

To help vision impaired users; there is one more mechanism that pronounces each character as and when they are hit on the keyboard. In some operating systems, this feature is available as part of a Narrator utility.

Arrow keys to control mouse:

Mobility impaired users have problems moving the mouse. By enabling this feature, such users will be able to use the keyboard arrow keys for mouse movements. The two buttons of the mouse and their operations too can be directed from the keyboard.

Screen accessibility:

Many of the keyboard accessibility features assist the vision impaired and mobility impaired users.

Visual sound:

Visual sound is the "wave form" or "graph form" of the sound. These visual effects inform the user of the events that happen on the system using the screen.

Enabling captions for multimedia:

All multimedia speech and sound can be enabled with next equivalents, and they are displayed on the screen when speech and sound are played.

Soft keyboard:

Some of the mobility impaired and vision impaired users find it easier to use pointing devices instead of the keyboard. A soft keyboard helps such users by displaying the keyboard on the screen.

Easy reading with high contrast:

Vision impaired users have problems in recognizing some colors and size of font in menu items.