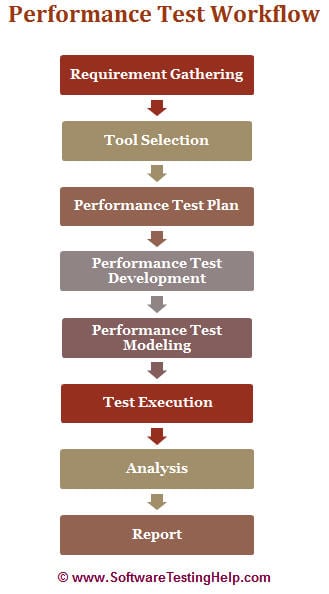
**Performance testing workflow**



**Requirement Analysis/Gathering**

The performance team interacts with the client for identification and gathering of requirements – technical and business. This includes getting information on the application’s architecture, technologies, and database used, intended users, functionality, application usage, [test requirement](https://www.softwaretestinghelp.com/how-to-test-software-requirements-specification-srs/), hardware & software requirements

.

**POC/Tool selection**

Once the key functionality is identified, POC (Proof Of Concept – which is a sort of demonstration of the real-time activity but in a limited sense) is done with the available tools.

**Performance Test Plan & Design**

Depending on the information collected in the preceding stages, test planning and designing are conducted.

Test Planning involves information on how the performance test is going to take place – test environment, workload, hardware, etc.

**Performance Test Development**

* Use cases are created for the functionality identified in the test plan as the scope of PT.
* These use cases are shared with the client for their approval. This is to make sure the script will be recorded with the correct steps.
* Once approved, script development starts with a recording of the steps in use cases with the performance test tool selected during the POC (Proof of Concepts) and enhanced by performing Correlation (for handling dynamic value), Parameterization (value substitution) and custom functions as per the situation or need. More on these techniques in our video tutorials.
* The Scripts are then validated against different users.
* Parallel to script creation, the performance team also keeps working on setting up the test environment (Software and hardware).
* The performance team will also take care of Metadata (back-end) through scripts if this activity is not taken up by the client.

**Performance Test Modeling**

Performance Load Model is created for the test execution. The main aim of this step is to validate whether the given Performance metrics (provided by clients) are achieved during the test or not. There are different approaches to create a Load model. “[Little’s Law](http://www.factoryphysics.com/principle/littleslaw.htm)” is used in most cases.

**Test Execution**

The scenario is designed according to the Load Model in Controller or Performance Center but the initial tests are not executed with maximum users that are in the Load model.

Test Execution is done incrementally. **For Example,** If the maximum number of users is 100, the scenarios are first run with 10, 25, 50 users and so on, eventually moving on to 100 users.

**Test Results Analysis**

Test results are the most important deliverable for the performance tester. This is where we can prove the ROI (Return on Investment) and productivity that a performance testing effort can provide.

**Report**

Test results should be simplified so the conclusion is clearer and should not need any derivation. Development Team needs more information on analysis, comparison of results, and details of how the results were obtained.

|  |
| --- |
|  |
|  |

**Performance testing tools**

1. WebLOAD
2. LoadNinja
3. HeadSpin
4. Ready API Performance
5. Load View
6. Key sight’s Eggplant
7. Apache JMeter
8. LoadRunner
9. Rational Performance Tester
10. NeoLoad
11. LoadComplete
12. WAPT
13. Loadster
14. k6
15. Testing Anywhere
16. Appvance
17. StormForge

Comparision between jmeter and loadrunner

JMeter is free while LoadRunner is expensive.

JMeter licenses on installation while the LoadRunner license is based on the number of virtual users.

JMeter has an unlimited load generation capacity while

LoadRunner has a limited load generation capacity.

JMeter is technically less proficient while LoadRunner is highly developed and complex.

JMeter lacks in the user interface while that of LoadRunner is impressive.