**Performance Engineering**

Software Performance Engineering is the systematic approach of constructing the software that meets the performance objectives. It is more related to build a tuned software application at the early stage of the software development life cycle whose performance satisfies all the non-functional requirement during performance testing phase. On the other hand, [Software Performance Testing](http://perfmatrix.blogspot.in/2016/09/basic-of-performance-testing.html) is the act of evaluating the software system for its performance and finding the bottlenecks in the system. *It is more related to quality assurance*.

**Software Performance Testing:** To test and certify the quality of the application with respect to the performance.

**Software Performance Engineering:** To plan and build a quality software without or minimum performance bottlenecks.

Performance engineering is driven by business requirements. It is a methodology to optimize the performance of the application from the earliest design stages. The involvement of a Performance Engineer at the initial phase reduces the last-minute application tuning activities. In short, the main aim of Performance Engineering is to provide better business value for the organization by discovering potential issues early in the development cycle to reduce the cost of fixing the bottlenecks in the later stage.

**Performance Testing vs Performance Engineering**

1. Performance Testing is a quality check of the application in terms of the application’s responsiveness and load handling capacity. Performance Testing verifies how a system will perform under production load and to anticipate issues that might arise during heavy load conditions. On the other hand, Performance engineering aims to design the application by keeping the performance metrics in mind and also to discover potential issues early in the development cycle.
2. Performance Testing is a distinctive QA process that occurs once a round of development is completed whereas performance engineering is an ongoing process that occurs through all phases of the development cycle i.e. from the design phase to development, to QA.
3. A dedicated performance tester or team conduct the Performance Testing who has sound knowledge of performance testing concept, tool operation, result analysis etc. A Performance Engineer is a person who has enough knowledge of application design, architecture, development, tuning, performance optimization and bottleneck root cause investigation and fixing.
4. When a bottleneck is identified during performance testing then the role of the performance tester is to analyse the test result and raise a defect. On the other hand, the job of a performance engineer is to investigate the root cause and propose the solution to resolve the bottleneck.
5. Performance Tester does not much care about the design and architecture of the application. He just focused on the application behaviour under load whereas Performance Engineer cares about how efficient each component of the application is performing under load.

**Performance Engineering tools:**

**First of all, there is no tool for “Performance Engineering”**.

In this process or step we analyse the results given by performance tester. In this step we try to find out the reason why the exceptions did not match during performance testing. This process also called “Route Cause Analysis(RCA) “

In the above example , If the average response time of the application become 10 seconds then its performance engineering process in which we try to find out why the **response time became 10 seconds ? Why the application is not behaving as expected ?**

**So, we have tools for performance testing and not for performance engineering but we have few tools to monitor all the accepts.**

1. **Dynatrace**
2. **Appdynamics**
3. **Pinpoint**
4. **SiteScope**
5. Performon …etc

**Skills Required To Be A Perfect Performance Engineer:**

1. **Learn Development Skills First**
2. **Know The Latest Performance Testing Tools And Trends**
3. **Understanding Complex Systems and Architecture**
4. **Designing Effective Tests**
5. **Scripting**
6. **Interpreting Performance Test Results**
7. **Understanding User Behavior**
8. **Learn Databases If Want To Tune And Fix Them**
9. **Understand the WorkLoad Model**
10. **Application Performance Management**
11. **Learn Operating Systems**
12. **Know The Importance Of Load Balancers**