Scalability performance of Software Testing by Using Review Technique

Mr. Ashish Kumar Tripathi, Mr. Saurabh Upadhyay, Mr.Sachin Kumar Dhar Dwivedi

Abstract-Software testing is an investigation which aimed at evaluating capability of a program or system and determining that it meets its required results. Although crucial to software quality and widely deployed by programmers and testers, software testing still remains an art, due to limited understanding of the principles of software, we cannot completely test a program with moderate complexity. Testing is more than just debugging. The purpose of testing can be quality assurance, verification and validation, or reliability estimation. Testing can be used as a generic metric as well. Correctness testing and reliability testing are two major areas of testing. Software testing is a trade-off between budget, time and quality.

Index Terms--Software testing modules, measurement process, performance and review technique.

1-INTRODUCTION

Testing is not just finding out the defects. Testing is not just seeing the requirements are Satisfied which are necessary in software development. Testing is a process of verifying and validating all wanted requirements is there in products and also verifying and validating any unwanted requirements are there in the products. It is also seeing any latent effects are there in the product because of these requirements. In software testing Scalability testing is to determine system behavior by increasing the load with a particular scaling ratio. For every scaling point all the performance attributes have to be determined. Also the factors affecting the application scaling capacity have to be determined. In software measurement review technique can apply for Scalability testing is to determine system behavior by increasing the load with a particular scaling ratio. For every scaling point all the performance attributes have to be determined. Also the factors affecting the application scaling capacity have to be determined; Software test documentation, Software user documentation, Maintenance manuals, System build procedures, Installation procedures and Release notes are possible candidates for the review.

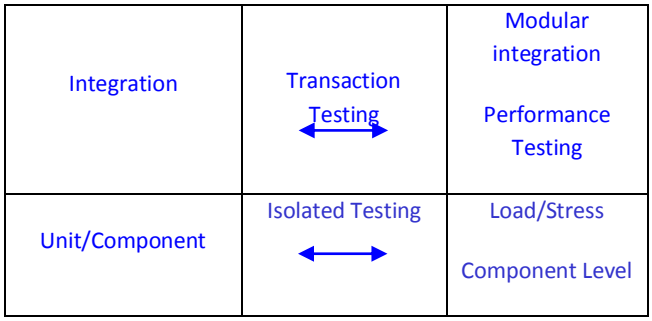
2-OBJECTIVE

Scalability performance in software testing is commonly used synonymously with **load testing**, although many testing professionals would argue there are subtle differences. We will include both types of testing in the resources available here. Both forms involve measuring throughput of a web application based on requests per second, concurrent users, or bytes of data transferred as well as measure the performance.

3-PROCESS FOR MEASUREMENT

In the measurement of configured software there are several technique can be applicable
In testing as a Software requirements specification , Software design description, Software test documentation, Software user documentation, Maintenance manuals, System build procedures, Installation procedures and Release notes are possible candidates for the review. The review meetings should be planned in the project plan or they can be held on request e.g. by the quality group.

Development Life Cycle	Performance life cycle	
Dev and Functional Team	Performance Team	
End-to-End	Transaction Testing ⟷	Full Transaction Performance Testing
Functional	Transaction Testing ⟷	Business Process Performance Testing



Fallowing life cycle diagrams the performance of functions which are reliable and uses in applicable methods for testing. There are some way which are necessary for uses as well as-

1-Load Testing: Load test is many concurrent users running the same program to determine whether a system can handle the load without compromising functionality or performance.

2-Volume Testing: Volume Testing determines the weaknesses in the system to handle large amounts of data during short time periods.

3-Stress Testing: Stress Testing determines if the system has the capacity to handle large numbers of processing transactions during peak periods. This data shall be then analyzed to determine the overall health of the system and to identify the bottlenecks problems.

4-Functional Testing- Functionality testing validates that an application conforms to its specifications and meets its expected functional requirements. During functionality testing, a range of inputs as test data is created and tests are performed to validate if whether each feature conforms to the requirements.

5-Spike testing-Spike testing is done by suddenly increasing the number of, or load generated by, users by a very large amount and observing the behavior of the system. The goal is to determine whether performance will suffer, the system will fail, or it will be able to handle dramatic changes in load.

6-Security testing- Our Security Code reviews can be either automated or manual, but mostly a combination of both. Some of the components are:

- Application Component Study
- Questionnaire-supported Discovery
- Risk and Attack Profiling
- Business Impact and Risk Analysis
- Technical Flaw Analysis

4-RELIABILITY OF TESTING

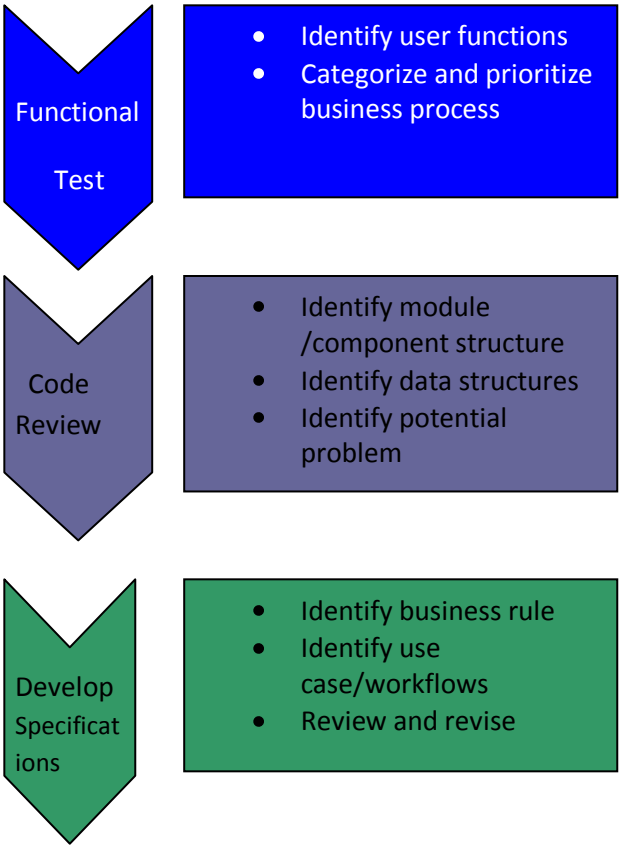
The reliability of a system is determines if the system will operate during a specified period of time. A system may be considered highly reliable it means it may fail very infrequently, but, if it is out of service for a significant period of time as a result of a failure, it will not be considered highly available, Reliability testing will tend to uncover earlier those failures that are most likely in actual operation, thus directing efforts at fixing the most important faults and also Reliability testing may be performed at several levels. Complex systems may be tested at component, circuit board, unit, assembly, subsystem and system levels. A key aspect of reliability testing is to define "failure.

Software Reliability Techniques
1-Trending reliability tracks the failure data produced by the software system to develop a reliability operational profile of the system over a specified time.
2- Predictive reliability assigns probabilities to the operational profile of a software system

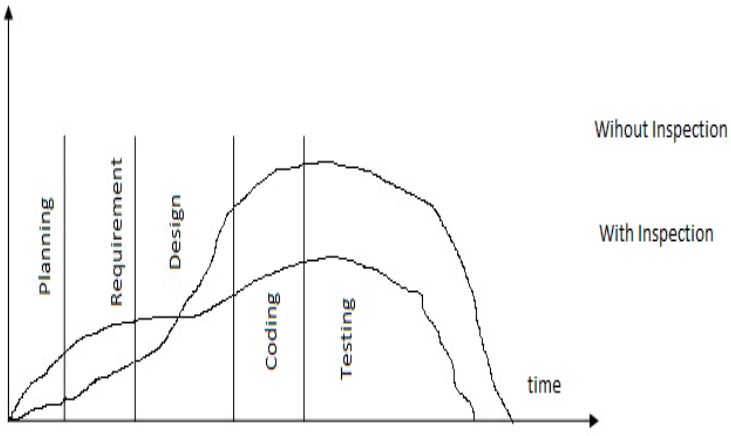
5-CHECK PERFORMANSE AND MEASUREMENT

In software engineering when we performance testing to check scalability of software product, in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate measure, validate or verify other quality attributes system, such as reliability and measurement of software it's resource usage.

Performance testing is a subset of performance engineering in software, an emerging computer science practice which strives to build performance into the design and architecture of a system, prior to the onset of actual coding effort.



Use performance testing to establish a baseline against which you can compare future performance tests. As an application is scaled up or out, a comparison of performance test results will indicate the success of scaling the application. When scaling results in degraded performance, it is typically the result of a bottleneck in one or more resources.



6-STRAITEGY APPLY FOR TESTING PERFORMANCE

1-A test plan should be created as part of the project planning phase.
2-Test strategy should refer to the requirements acceptance criteria that were developed as part of Requirements should be rejected if they fail the "Smart" test i.e. each requirement must be:

- Specific i.e. Objectives should state exactly what will be achieved (e.g.; unambiguous);
- Measurable i.e. Objectives must be quantifiable so that you will know if you have met the requirements

3-A test data acquisition plan should be completed at the end of the architecture and design phase.
4-Large volumes of data for should not be used for unit, system, integration, regression and quality assurance testing
5-A requirements traceability matrix should be used to outline all testing required for each requirement.
6-The author of each architecture and design document should identify how the design satisfies requirements
7-All code should be the subject of a peer review.
8-Each module design specification should identify the specific requirements satisfied by the
should focus on reviewing code and unit test results to provide additional verification that the code conforms to data movement best practices and security requirement.
9-Should include a code review to address potential coding vulnerabilities such as:

- Cross-side scripting.
- Injection flaws, particularly SQL injection;
- Malicious file execution;
- Insecure direct object references;
- Information leakage and improper error handling;
- Broken authentication and session management;
- Insecure cryptographic storage;
- Insecure communications;

7-LIMITATIONS

- We cannot test a program completely
- We can only test against system requirements

- Exhaustive (total) testing is impossible in present scenario.
- Time and budget constraints normally require very careful planning of the testing effort.
- Compromise between thoroughness and budget.
- Test results are used to make business decisions for release dates.
- Even if you do find the last bug, we'll never know it
- We will run out of time before you run out of test cases
- We cannot test every path
- We cannot test every valid input
- We cannot test every invalid input

8-CONCLUSIONS

Scalability Testing: Easy to increase the performance of the software if the application demands it. For example, a database application that gives good response time for 10 users should be scalable for 100 users if required.
The execution of our application build under customer expected configuration and customer expected load to estimate performance.

REFERENCES

[1] I-[BOE88] Boehm, B., "A Spiral Model for Software Development and Enhancement,"

[2] *Computer,* vol. 21, no. 5, May 1988, pp. 61–72.
[3] II-[BOE96] Boehm, B., "Anchoring the Software Process," *IEEE Software,* vol. 13, no.
[4] 4, July 1996, pp. 73–82.
[5] III-[BOE98] Boehm, B., "Using the WINWIN Spiral Model: A Case Study," *Computer,*
[6] vol. 31, no. 7, July 1998, pp. 33–44.

[7] IV-[DAV94] Davis, A. and P. Sitaram, "A Concurrent Process Model for Software
[8] Development," *Software Engineering Notes,* ACM Press, vol. 19, no. 2, April 1994,
[9] pp. 38–51.
[10] V-[DAV95] Davis, M.J., "Process and Product: Dichotomy or Duality," *Software Engineering*
[11] *Notes,* ACM Press, vol. 20, no. 2, April 1995, pp. 17–18.
[12] VI-[DON96] Donaldson, M.C. and M. Donaldson, *Negotiating for Dummies,* IDG Books
[13] Worldwide, 1996.

[14] VII-[ALV64] Alvin, W.H. von (ed.), *Reliability Engineering,* Prentice-Hall, 1964.
[15] VIII-[ANS87] ANSI/ASQC A3-1987, Quality Systems Terminology, 1987.
[16] IX-[ART92] Arthur, L.J., *Improving Software Quality: An Insider's Guide to TQM,* Wiley,
[17] 1992.
[18] X-[ART97] Arthur, L.J., "Quantum Improvements in Software System Quality, *CACM,*
[19] vol. 40, no. 6, June 1997, pp. 47–52.
[20] XI-[BOE81] Boehm, B., *Software Engineering Economics,* Prentice-Hall, 1981.
[21] [CRO79] Crosby, P., *Quality Is Free,* McGraw-Hill, 1979.
[22] XII-[DEM86] Deming, W.E., *Out of the Crisis,* MIT Press, 1986.
[23] XIII-[DEM99] DeMarco, T., "Management Can Make Quality (Im) possible," *Cutter IT Summit,*
[24] Boston, April 1999.

.

Author-1-


Mr. Ashish Kumar Tripathi

Assistant Professor Dept. of CS, Centre for Management Technology Greater Noida, India E-mail:- eraktphd@gmail.com

2-


Mr.Saurabh Upadhyay
Lecturer in Dept. of CS, S.M.S Varanasi affiliated to U.P.Technical University Lucknow, INDIA
E-mail:- Saurabh.bac@gmail.com

3


Mr.Sachin Kumar Dhar Dwivedi,
Assistant Engineer, NIELIT Aizawl, Ministry of CO & IT Govt of India.
E-mail:- Sachin.nielit@gmail.com