Vuk Vuković,
Jovica Đurković,
Jelica Trninić

Defining Performance Criteria and Planning Performance Tests for the Exam Registration Software

Article Info:

Received 12 June 2014
Accepted 24 June 2014

UDC 004.4

Recommended citation:

Vuković, V., Đurković, J., Trninić, J. (2014). Defining Performance Criteria and Planning Performance Tests for the Exam Registration Software. *International Scientific Journal of Management Information Systems, 9* (2), 15-19.

Summary

In addition to functionality, the quality of software products is also characterized by other software quality attributes, such as performance, availability, portability, reliability, usability, etc. These must inevitably be ascribed appropriate significance in the software product testing process. In the case of business software products, certain qualities stand out above others in terms of significance, so that they must receive adequate attention in the business software testing process. Software performance is definitely one of them. By means of a case study, this article presents a possible avenue of planning a business software performance testing process.

Keywords

Software, performance criteria, quality, testing process, planning

1. Introduction

Software performance is an indicator showing the degree of a software product's ability to meet the defined requirements over a given time period. (Smith & Williams, 2002).

"Performance testing is defined as technical examination with the basic purpose of determining, i.e. assessing the features in terms of speed, scalability and/or stability of the systems or applications that are tested" (Meier, et. al., 2007, p.15). Stress and load testing are regular sub-categories of software performance testing (Đurković, Trninić & Vuković, 2011).

Load testing involves exposing a system to statistically representative, normal or expected load. Normal load includes minimum and maximum load that a system can endure without the need for engaging additional resources, as well as functioning of an application without excessive delays. Stress testing focuses on determining or evaluating the characteristics or performances of the application tested in the conditions, i.e. under stress which is above the levels anticipated in practice. Moreover, this form of testing may involve system or application test in "harder" working conditions such as limited memory and disk space.

These tests are specifically designed to determine under which conditions an application may fail, how and which indicators can be monitored preventively so as to point to potential failures. Errors and shortcomings detected by this form of testing may be fully eliminated, but not necessarily, which is a question of assessment of the testing team. (Meier, et al, 2007, p.21).

Microsoft Corporation has established a total of seven core activities in their approach to software performance testing based on acquired experience, representing an excellent methodological framework for performing this form of testing. (Meier, et al, 2007, pp. 45-46).

According to this methodology, two mandatory activities in software performance testing process are: (1) identify performance acceptance criteria and (2) planning performance tests. The subject of this article is focussed on these software performance testing process activities, on the example of the Exam Registration business software, as a part of the information system of higher education institutions. The following section presents one of the modalities of defining acceptable performance acceptance criteria, and planning performance tests, which is also the aim of the paper.

2. The Eight-Second Rule

One of crucial factors to be taken into consideration when defining acceptable performance test criteria is the Eight-Second Rule, discovered and formulated in the 1970s by measuring the impact of system response time to the system user's productivity. Measurement was performed in such a way that some transactions were completed within a split second, whereas

others took up to 20 seconds. The conclusion made then, and confirmed in the 1990s with the emergence of the Internet, was that the user's productivity declines dramatically if the system response time is longer than eight seconds per transaction. The explanation for the above stated is that, if the response time per transaction amounts to eight seconds or less, the user will, in that case, keep the next activity that he plans to do on the system in memory. Response time longer than eight seconds will result in forgetting the system user's intended subsequent activity. For this reason, the Eight-Second Rule must be taken as a relevant factor when defining requirements in terms of response time, i.e. responsiveness of business software products. (Everett & McLeod, 2008)

3. Planning performance tests for load testing

3.1. Response time

For all key system transactions it is necessary to define requirements in terms of system response time. Transactions with common requirements in terms of response time can be grouped, and response time presented by transaction batches, whereas transactions with specific requirements in terms of response time are presented separately. The maximum values of response time are taken as referent for response time (Table 1).

Table 1 Response time requirements per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time
System login/logout	Maximum 2s
Student's data display	Maximum 2s
Display of possible examinations to take	Maximum 3s
Exam registration	Maximum 3s

3.2. Workload

In addition to response time, it is necessary to determine maximum workload for identified transactions or transaction batches, and the time periods in which the peak workload may appear. A usual measurement for workload is the number of active users (Faban, 2013). If 300 users log into the web application for exam registration during the morning hours, and 1000 users in the evening when there are no lectures, then the workload referent value is taken to be 1000.

As regards the number of users, it is important to differentiate between active users and users using the system simultaneously. The number of active users is relevant for defining the maximum workload, whereas the number of parallel users is relevant when determining specific tests for the database. In addition to the number of active users, identified transactions also require the definition of the time period in which the maximum workload is expected (Table 2).

Table 2 Response time, active users and time period per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time	Maximum number of active users	Time period
System login/logout	Maximum 2s	1000	Exam registration times before examination periods
Student's data display	Maximum 2s	1000	Exam registration times before examination periods
Display of possible examinations to take	Maximum 3s	1000	Exam registration times before examination periods
Exam Registration	Maximum 3s	1000	Exam registration times before examination periods

However, maximum workloads for particular transactions are not always the same at the same time. Given the university students' obligations and commitments, the example of exam application web application can be used for simple forecasting whether the application will be used when there are no lectures, that is, late in the afternoon, in the evenings, or at weekends. Several plans for testing maximum workload must therefore be defined (Tables 3, 4, and 5.)

Table 3 Response time, active users and 1st time period per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time	Maximum number of active users	Time period
System login/logout	Maximum 2s	500	Exam registration times during lectures
Student's data display	Maximum 2s	500	Exam registration times during lectures
Display of possible examinations to take	Maximum 3s	500	Exam registration times during lectures
Exam Registration	Maximum 3s	500	Exam registration times during lectures

Table 4 Response time, active users and 2nd time period per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time	Maximum number of active users	Time period
System login/logout	Maximum 2s	1500	Exam registration times before examination periods - evenings
Student's data display	Maximum 2s	1500	Exam registration times before examination periods - evenings
Display of possible examinations to take	Maximum 3s	1500	Exam registration times before examination periods - evenings
Exam Registration	Maximum 3s	1500	Exam registration times before examination periods - evenings

Table 5 Response time, active users and 3rd time period per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time	Maximum number of active users	Time period
System login/logout	Maximum 2s	1500	Exam registration times before examination periods – weekends
Student's data display	Maximum 2s	1500	Exam registration times before examination periods – weekends
Display of possible examinations to take	Maximum 3s	1500	Exam registration times before examination periods – weekends
Exam Registration	Maximum 3s	1500	Exam registration times before examination periods – weekends

3.3. Throughput

Throughput is another ineluctable parameter used when testing business software product performance. It is defined as the number of completed transactions in a unit of time, or the capacity an application can process. Throughput as a parameter is one of performance criteria, and its target values must be defined before beginning to test business software product performance.

Throughput can be figuratively explained on the example of a petrol station. Let us assume that a single pump on the petrol station has the capacity to fill up a tank load of fuel (the size of the tank is irrelevant in this case, for the sake of analogy) within a minute. If the petrol station has five pumps, then its maximum throughput is five cars per minute. If, however, a larger number of cars appear at the station, longer queues are expected to form (Colantonio, 2011).

The situation with software is identical. If an application receives 100 requests per second, and can process only 50, the remaining 50 will have to wait. The most frequent measure unit for expressing throughput in software performance test is number of transactions per second. In this particular example, throughput would be expressed in number of registered exams per second.

After defining the performance criteria, designing the performance tests, and configuring the testing environment, the next activities are implementation and then execution of performance tests. (Meier et al., 2007). Planning the execution of performance workload tests is recommended by means of the following three steps (Everett & McLeod, 2007):

1. Increase workload to maximum value;
2. Decrease workload from maximum value; and
3. Measure performance during maximum workload.

1. Increasing the workload to maximum value is the process of supplying a sufficient number of user sessions to achieve maximum workload. If the maximum workload is 1500 users, in that case 1500 users must log into the system in the testing environment (the users are virtual). However, the aim of this step is to verify user login, so that it is unnecessary to ask users to perform transactions. In other words, if the system does not pass this stage, it is redundant to perform transactions until the system is adjusted to set requirements. Increasing the workload will reveal all the possible problems in terms of resources such as lack of operative memory, insufficient processor capacities, insufficient memory space for files, etc. When the software has successfully passed the maximum value workload increase test without performing transactions, it is ready for the subsequent stage.

2. Decreasing the workload from maximum value is the process of ending all user sessions opened in the previous step. Users log out of the systems in the testing environment in particular time intervals. Although vacating the resources should result in faster work and reduce problems, certain errors may appear due to this process, so that this step in performance testing must be taken.

3. Performance measurement during maximum workload is performed after the previous two activities, and is focussed on measuring responsiveness, i.e. response time of the software.

According to the defined performance test plans, it is necessary to start increasing the number of transactions in the testing environment – in this specific case, the number of exam registrations. For illustration, Figure 1 shows the correlation between the number of registered exams and the web application's response time to exam registration, where the abscissa presents the number of transactions completed in the same time interval, and the ordinate records the lowest response time measured for any of the simultaneously completed transactions.

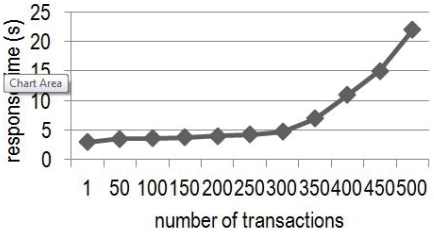

Figure 1 Correlation between the number of transactions and response time.

Analysing the diagram in Figure 1, we can conclude that after increasing the number of transaction to more than 300, the time response curve transforms from a linear to an exponential curve. The point where the curve changes its property is the bottleneck, which does not reveal the cause of the problem in the application, but serves as an important indicator for development team members.

4. Planning performance tests for stress testing

As said earlier, the aim of stress testing is to identify the software problems occurring only in extreme conditions of application functioning. Extreme conditions include greater workload than maximum, limited hardware resources, high level of simultaneous use of the software, etc. In load testing, the number of virtual users is increased until a point is reached where the software's response time grows rapidly (Figure 1) due to overloaded server where the application is located. Stress testing, on the other hand, is aimed at keeping the software around this bottleneck point for a longer time, so that the problems caused by simultaneous application use and those related to hardware resources can be identified. In other words, stress testing enables the identification of the software's endurance limit. If the identified limits are assessed to be satisfactory, the software can be handed over to the users. In the opposite case, the application's performance must be enhanced by removing particular software errors, adding new functionalities to reduce the load on particular segments of the application, or improve the hardware resources on which the application will function (Everett & McLeod, 2007).

Stress testing does not require new performance test plans. Plans developed for load testing can be used, with the difference that the number of virtual users must be adjusted (Table 6).

Table 6 Response time, active users and time period per selected transaction in the Exam Registration web application

Transaction / Transaction batch	Response time	Maximum number of active users	Time period
System login/logout	Maximum 2s	3000	Exam registration times before examination periods – weekends
Student's data display	Maximum 2s	3000	Exam registration times before examination periods – weekends
Display of possible examinations to take	Maximum 3s	3000	Exam registration times before examination periods – weekends
Exam Registration	Maximum 3s	3000	Exam registration times before examination periods – weekends

5. Conclusion

The business software product testing process cannot be regarded as completed unless performance is tested. However, appropriate planning is the prerequisite for successful performance testing. This article has presented a developed performance testing plan with defined performance criteria for the Exam Registration web application. To sum up, all the relevant aspects of business software product performance have been highlighted, and guidelines for planning performance testing of any other business software product have been proposed.

References

Colantonio, J. (2011). Performance Testing Basics – What is Throughput?. Retrieved November 11, 2013 from: http://www.joecolantonio.com/2011/07/05/performance-testing-what-is-throughput/

Đurković, J., Trninić, J., Vuković, V., (2011). Test Software Functionality, but Test its Performance as Well. The International Scientific Journal of Management Information Systems, 6(2) 2-7.

Everett, G. D., McLeod, R. (2007). Software Testing: Testing Across the Entire Software Development Life Cycle. New Jersey: John Wiley & Sons.

Faban (2013). Performance Workload Design. Retrieved November 11, 2013 from: http://faban.org/docs/WorkloadDesign.pdf

Meier, J. D., Farre, C., Bansode, P., Barber, S., & Rea, D. (2007). Performance Testing Guidance for Web Applications. Microsoft Corporation.

Smith, C., Williams, L. (2002). Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software. Boston: Addison-Wesley Professional.

Vuk Vuković
University of Novi Sad
Faculty of Economics Subotica
Segedinski put 9-11
24000 Subotica
Serbia
Email: vuk@ef.uns.ac.rs

Jovica Đurković
University of Novi Sad
Faculty of Economics Subotica
Segedinski put 9-11
24000 Subotica
Serbia
Email: djovica@ef.uns.ac.rs

Jelica Trninić
University of Novi Sad
Faculty of Economics Subotica
Segedinski put 9-11
24000 Subotica
Serbia
Email: trninicj@ef.uns.ac.rs