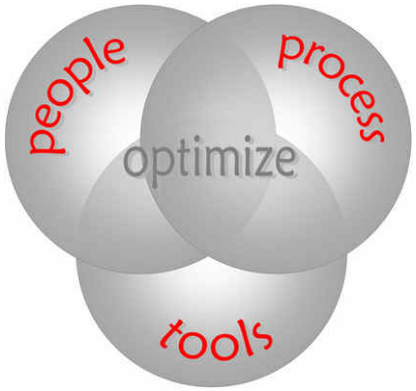
quasus

quality and agile solutions by united services

The senseless investment in performance testing



quasus

quality and agile solutions by unified services

Agenda

- The classical view
- Value of obtained results
- The complexities of performance testing
- How to set up a proper performance test
- A cheaper performance test
- Real-life case

The classical view

- Why do a performance test?
  - To determine whether an application supports a certain number of simultaneous users or tasks
  - To determine whether an SLA can be respected
  - To see if an application meets certain (legal) requirements
  - To meet a formal requirement, for example for a change implementation board
  - ….

In short, performance tests provide a certain peace of mind.

The classical view

- Many people in the project see performance tests almost as a necessary evil
  - Thus, they don't really care much about the validity of the results as long as the tests get done ASAP
  - In fact, many IT people, especially the more technically oriented ones, don't even believe in performance tests,
- At the same time, the stakeholders of the results will want those results to be reliable

The classical view

- Performance tests are expensive and require a large commitment of resources
  - Tools are expensive
  - Preparation of data is time-consuming
  - Set-up of infrastructure is time-consuming
  - During the tests, an exclusive use of the infrastructure is required
  - Support of technical people such as DBA, network admin,… is required
  - In administrative organizations, getting the required permissions can be even more time-consuming, since performance tests typically require many administrative rights

quasus

quality and agile solutions by unified services

Performance test results

- A performance test with a specialized tool can generate a vast amount of data
- Interpretation of these results is probably more difficult than setting up the performance test
  - It requires a lot of technical knowledge
  - You need to know how these results have been influenced
  - You need to know which results are relevant
  - You need to 'convert' the data gathered by the tool into *information*

Interpretation of results is the most difficult part of performance testing.

Performance test results

- You can repeat a test, but you cannot be sure whether the results will be really comparable
- Likewise, you can perform the same performance test on different pieces of software or in different environments, but will the results be comparable?
  - Cfr. Networkworld.com on hardware intrusion protection systems. “why we didn’t test performance” by J. Snyder, R. Thayer and D. Newman, 16/02/2004.

There are so many factors influencing the results of a performance test that it is impossible to account for them all.

Performance test results

- A small change in environment, configuration or application can have a large impact on performance
- Performance test cannot be executed in a production environment, since the risk of disrupting production is too large
- Thus, performance test should be executed in a “beta” environment, an approximation of the production environment.

In real life, the test environment and the production environment are never exactly the same. Thus, the value of the obtained results can be disputed.

quasus

quality and agile solutions by unified services

The senseless investment in performance testing

Performance tests are expensive. A lot of budget and resources, spent on performance tests, might as well be spent on other kinds of tests.

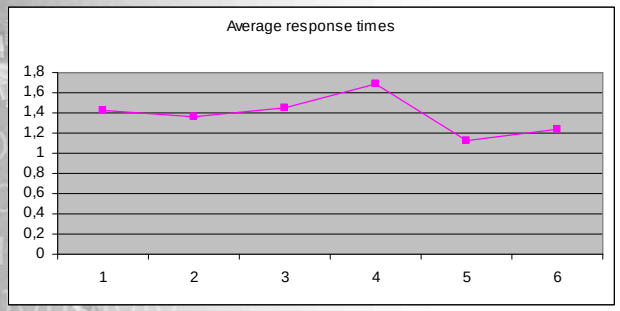
If you do not understand nor take into account the limitations of performance testing, it is senseless to invest in a performance test.

The complexities of performance testing

- Infrastructure
  - Select the correct tool
  - Select and implement a testing environment
  - Realistic infrastructure usage
- Usability of tests
  - Tests are resource-intensive
  - Software needs to be in final version
- Creation of tests
  - How to determine what to simulate?
  - How to determine which performance factors are critical?
  - Know what the desired results are

The complexities of performance testing

- Real-life scenarios
  - Simulate human behavior → is this possible?
  - When to run these scenarios?
  - External influences can disturb the results
  - And it can be very difficult to determine their cause



- Results
  - How to read the results
  - How to interpret the results
  - How to communicate the results

A proper performance test

- Know what the limitations of the test are going to be
- Invest more time in preparation
- Use the right tool for the job
- Start tests only when software is in its final version
- Know your users and how they use the software
- Know what you need to measure to determine performance
- Take into account environment limitations
- Do baselining and multiple runs
- Have a ‘professional’ interpret the results
- Take the results with a grain of salt

A proper performance test

- Limitations of a performance test
  - Even a perfectly prepared and executed performance test is still a simulation of reality. Factors like human behavior, environmental influences,… will influence the results in real life.
  - Make sure all stakeholders are aware of this, and of what they can and cannot expect from the results. This will avoid disappointment.
- Invest more time in preparation
  - Only a properly prepared performance test can be a success. A good preparation will save you time in the long run.
  - A thorough preparation includes activities such as tool selection, business process analysis, determining the required results, baseline testing,…

A proper performance test

- Use the right tool for the job
  - Tool selection is an essential process. Even if your organization has purchased a tool, you need to determine whether it is the right tool for the job
  - Build your own tool or purchase a tool
- Test the final version
  - Only the final version of software can be used for performance tests. Even small software changes can have a significant impact on performance
  - This seems logical, but it is the most sinned to rule
  - A small exception can be profiling of an unfinished application or part of an application

A proper performance test

- User behavior and business process analysis
  - Users often use an application in a different way than foreseen by developers; this can have an impact on application performance
  - Use log files, surveys, expert opinions,… to determine application usage.
  - This step is essential to guarantee a somewhat realistic simulation
- Know what you need to measure
  - Know what figures you need to record and know what the measurements mean.
  - This is important to help developers identify problem areas; They have no use for a message like “the application is really slow”.

A proper performance test

- Set up a realistic environment
  - A performance test in a environment that is not production-like is almost useless
  - Many organizations have a standard test environment that is a sized-down version of the production environment. Performance test results from such an environment can NOT simply be extrapolated
  - Modeling can help with this extrapolation but does not always give reliable results
  - This is one of the most time-consuming steps, and also expensive. All stakeholders need to be aware that this step is required to obtain useful results.

A proper performance test

- Extrapolation through modeling
  - Modeling tries to quantify the dynamics of the relationships between different parts of the application and their environment
  - The goal is to reduce guesswork in extrapolation
  - The components of the architecture are represented in the model, that tries to mimic the behavior of the business flow through the system
  - The more complex the model, the more accurate, but also the more difficult to control
  - Extrapolating test environment results to production environment requires model accuracy of at least 90%
  - Reference: “Moving beyond Test and Guess”, Richard L. Gimarc, Amy Spellmann and Jim Reynolds, 2005

A proper performance test

- Set up a realistic environment
  - In case of multiple applications, test environment use is often not realistic
  - Exclusive use of the test environment seems tempting but is also not the solution
  - You need to do business transaction analysis not only for your own application, but in a limited way also for the other applications running in the environment
  - The easiest way to achieve a realistic environment, is to connect the test environment to the production environment, but there are often security issues or practical issues
  - You can also try simulator tools to mimic typical environment usage

A proper performance test

- Realistic environment: the poor man's approach
  - In case you don't have budget or time for complex measures
  - Find as many similar applications in the production and the test environment as possible
  - Calculate an average conversion factor for the difference in their performance, based on figures such as transactions per second, hits per second, database operations per second,…
  - Apply this conversion factor to your application
  - This will probably generate less accurate results than modeling, but in practice it works.

A proper performance test

- Baseline runs and multiple runs
  - Baseline runs determine performance with multiple users but without load. They are required to determine the impact of adding load and they also avoid doing a full performance test and then finding out there are multiple-user issues
  - Multiple runs are required to determine how high environmental influences on your results are
  - If results from different runs differ much, this can indicate unacceptable external influences
- Have a ‘pro’ handy to interpret results
  - Nobody knows the system better than the relevant admins. Show them the relevant results and analyze with them. This can not only save enormous amounts of time, but also help to identify external influences

A proper performance test

- Take the results with a grain of salt
  - Everybody should be aware that the results will not be very accurate
  - This is essential when using them in SLA's and other critical applications

A cheaper performance test

- Considering previous rules, the test will be as ‘cheap’ as possible
- Act as if an expensive external will come in to execute the tests
  - Make sure everything is ready to go
- Consider tool rental schemes or offshore performance testing
  - This can save a lot of money without impacting test results
- If you do not have the relevant knowledge, get it somewhere else
  - Performance testing is the most complex form of test automation

Case

- Large international bank, international payments application
- Java technology, websphere and oracle
- Reporting application with upload part (Swift MT940 and MT942) and data distribution part.
- Previous performance test: 36 man days booked
- In reality, preparation of data, development of the script and execution of the tests took less than 1 workweek.
- Causes;
  - Environment configuration problems
  - Environment stability problems
  - Disturbances because of other applications
- Beta environment: scaled-down and unrealistic version of production

Case

- A new performance test was required, with two goals
  - Get numbers for a Service Level Agreement
  - Determine operational capacity of request-reply distribution part for client
- A much more comprehensive preparation was done this time
  - Profiling of the application was done before starting the performance tests; this already revealed some multi-user functional problems
  - A conversion factor between the test environment and the production environment was calculated. There were many applications available to calculate this factor, and it seemed to be comparable for almost all applications. This gave us confidence in this approach.
  - The scope of the performance test was reduced, to compensate for the environment instability problems.

Case

- A much more comprehensive preparation was done this time
  - Agreements were made with all other applications running on the server about the test period to simulate realistic use of the environment
  - The data structures to be used for the performance test were reserved in order to prevent data contamination
  - 1 month in advance, we began channeling production data to the test environment in order to have realistic data for the performance test
  - Messages used during the tests were also much more adapted to the real production data

Case

- Test Execution
  - Old scripts were not re-used
  - Many baseline test runs were done to determine base application performance.
  - These baseline runs were also very useful to detect unwanted external influences and to eliminate or compensate for them
  - Finally, these baseline runs allowed us to detect more than 10 configuration problems
  - Multiple runs confirmed the effort of preparation was not wasted: As opposed to all previous performance test, results were very stable

- SLA figures
  - To be on the safe side, after conversion of the results to ‘production results’, we deducted 30% to allow for conversion errors

Case

- Distribution performance
  - Request-reply distribution is message-based
  - Measured performance in beta environment was 12.92 messages per second, optimal number of threads was 20
  - The calculated conversion factor for messages per second was 2.34, this amounts to 30.23 messages per second in the production environment
  - The measured performance in the production environment is between 25 and 35 messages per second.

Booked time on the performance test is 17 man days. There were also 13 man days booked from other departments for the setup of the realistic use of the test environment, but this was mostly a one-off effort.

Case

- End results
  - The SLA has never been broken because of performance issues
  - The client is very happy with distribution performance
  - Icing on the cake: The baseline runs and multiple test runs allowed us to discover a number of issues with environment configuration and database configuration that were also present in the production environment, even for other applications!

quasus

quality and agile solutions by unified services

Contact

n.v. **Quasus** s.a.

Excelsiorlaan 89 box 1

B-1930 Zaventem

☎ +32 2 712 96 50

+32 2 712 96 59

@ info@quasus.be

URL:www.quasus.be