Understand performance testing and why it is becoming important

by Lee Miguel Lopez - Thursday, April 27, 2017

http://blog.belatrixsf.com/understand-performance-testing-and-why-it-is-becoming-important/

Today we have greater expectations of the software we use than ever before. This is the number one reason why performance testing has become so important. Research suggests that just a 1 second delay in page load time results in 7% fewer conversions, 11% fewer page views, and a 16% decrease in customer satisfaction. And this translates to real dollars – if your site earns $100,000 per day, you're losing $2.5 million every year due to this 1 second.

In this blog I want to give a brief overview of performance testing, and some of the factors to consider when conducting it. To start with, it's worth noting that performance testing is a non-functional test, which examines the relationship between the application and its environment. It includes a range of different types of test, from load tests to stress tests, soak tests, each of which have different objectives. It is an essential part of the development lifecycle because it is required for each new release, in order to see if new changes or features impact (positively or negatively) the performance of the application.

What is "good performance"?

"Good" performance is a matter of perception and depends on the purpose of the software. It may be acceptable for your business for example, if an internal file transfer system takes a few seconds to work, but your customer facing website needs to offer customers a seamless and lightening-fast experience. Similarly, the speed and stability of a mobile application will be key factors as to whether customers will actually use the app, or if they will uninstall it at the first opportunity they have.

It's important to start thinking about what constitutes acceptable performance at the start of your project, rather than waiting until it is ready to release. By building in considerations of performance during the early stages of product development, you improve the likelihood of your application meeting or exceeding expectations when it is live.

Scripting is at the core of performance testing

Ultimately performance testing is about "humanizing the robot". We need to simulate different uses of the application. At the core of this, you'll find scripting. Scripting can emulate what humans are doing with the application and how they are interacting with it. However, not every test has to be made with scripting. For example, it is also important to monitor the environment, and export and correlate the results - that way we can understand what is happening on the server-side.

Choosing the right tool

For scripting there are many tools available in the market to design and run scripts. Many of these tools are under license, but there are also some open source options such as Apache JMeter which can be used for load testing both static and dynamic resources. I recommend evaluating different tools and choose the most appropriate for your needs - your choice will depend on different factors, from the communication protocol your application is using, to how many concurrent users you want to simulate.

There are also tools available to monitor and analyze the performance of your servers. These tools can monitor different aspects such as operating system consumption or memory usage. However, these tools can be expensive, and many will charge by the number of servers or throughput generated, or time.

Evaluate the cost-benefits of performance testing

Although performance testing has many benefits, I still recommend evaluating the cost-benefits of conducting it. There is a balance between the cost of conducting extensive performance testing and improving performance (as mentioned earlier, many tools are expensive), versus the expected business gains of that improved performance. I recommend considering the following:

- **The number of users.** If an application will be used by just a few users, then it becomes less important to conduct extensive performance testing, but with a larger number of concurrent users, then it becomes critical.
- **Response time.** If the response time is important, again make sure to conduct performance testing (such as for your customer-facing mobile app).
- **Architecture.** If you have a distributed architecture, made up for example of different kinds of database servers, application servers, and web servers, it’s important to know if there could be a bottleneck.
- **Infrastructure.** If you are using shared infrastructure or components, then it’s important to evaluate the performance of your servers.
- **And finally, make sure to evaluate the expected use of your application.** For example, think of an ecommerce sales campaign and the expected use before, during, and after the campaign. Prior to starting the campaign, it’s essential to test what happens if it’s successful and you have a sudden spike in the number of users.

To find out more on this topic, I recommend listening to a recording of my webinar which I gave last week, and where I go into more detail about the different types of tests that make up performance testing.