All About Load Test Results and the Results DB (Part 3 – What Results Should I Use When Reporting Back To Management?)

06/10/2014 • 8 minutes to read

This question is extremely important and is necessary to answer BEFORE analyzing the results of a test run. In fact, it should already be answered *before you write the test harness*. This is because you need to know why you are executing a test in order to know how to write the test. In my role as a Performance Test Consultant, I work with customers to define, build, execute and report on various types of tests. The first thing I do with any customer is drill into the end goal of the engagement I am assigned to ("*Why are you paying Microsoft to have me help you and what do you expect to get from the work we do?* "). It is not uncommon to spend 50% of my allotted time for pre-testing work on an engagement just defining and writing the test plan, which includes details like:

- Goals and Objectives
- Success Criteria

These items give me the basis for how to

- write the test harness
- generate the data to feed into the test harness
- generate the data that is in the System Under Test (SUT)
- Setup the load test(s) to collect the proper performance metrics and data
- Etc.

They also give me an understanding of what to look at when I am analyzing the results. (NOTE: My next post will cover some details about what results are available to use and explain a few things to consider when reading the results). Here is an example of a conversation I might have to define the goals, objectives and success criteria for an engagement. Keep in mind as you read this that it is much simpler than the typical conversations I have and that I am leaving out a lot of details:

Conversation

Me: What do you want to accomplish in this engagement?

Customer: We want to know the breaking point of our application.

Me: What do you mean by breaking point?

Customer: When the application becomes too slow to be usable.

Me: What is too slow?

Customer: Um, well. Let's say that pages take longer than 2 seconds to return.

Me: All pages?

Customer: sure.

Me: One of the things your application does is compare prices from 10 different retailers to allow your end users to see the best price they can get. I would think that building that result will often take longer than 2 seconds. Can we define five or ten key pages and set different response goals for each?

Customer: Yeah, that seems reasonable. What pages should we use and what time for each?

Me: Can you reach out to the business group that owns the product and see what they say? That group should be able to know what their expectations for the application will be. Let's let them give us the first round of numbers and then work with them to adjust the numbers as needed. Also, we should have then define the number of failures that are acceptable, etc. As an example, we can say something like "90% of all requests to the summary results page should respond within 5 seconds."

Customer: Makes sense. We also have requirements from the IT department that we need to be aware of.

Me: Like what?

Customer: The CPU on the web servers should not be above 70%

Me: OK, is that 70% an absolute maximum, or is that an average over the entire run, or is that a value where it should not stay above 70% for more than XX seconds in a row?

Customer: I don't know. What do you think?

Me: It depends on the application and your IT department, but I would personally start out with it can spike above 70%, but it should not stay sustained above that value for more than 30 seconds. However, this really needs to be decided by your IT department or the business partner.

Customer: OK

Me: Next, what is the expected load on the system going to be so we know how hard to drive the test, or are you simply looking to determine how much work the system can handle? (NOTE: Notice I did NOT ask about the number of users. I care about throughput, not just user count).

Customer: 1,000 concurrent users would be the maximum we would expect to see.

Me: doing how much work each?

Customer: What do you mean?

Me: The number of concurrent users is not enough to define the load. You need to consider how much work each user will be doing over a given time frame. We call this the "Load Profile." If each user is hitting 10 pages and logging off in a given hour, then our profile is 1,000 users X 10 pages per user per hour, or 10,000 pages per hour.

Customer: Oh, got it. That's easy enough. We have around 32,000 pages per hour (according to our web logs) so we'll just make that our target load.

Me: We also need to ensure that those 32,000 pages are representative of the type of work being done. Do you know the breakdown of the work?

Customer: All 10,000 log in once and perform at least one search, but only 2,000 of them complete an entire purchase, about 6,000 of them perform two searches, and 2,000 of them perform three searches.

Me: So do the 2,000 users who purchase something also perform a search? Or multiple searches? Do we need to worry about different search types?

Customer: Every user logs in, 2000 users will search once, 6,000 will search twice, and 2,000 will search three times.

Me: So that means we have: 10,000 logins, 20,000 searches and 2,000 purchases. Correct?

Customer: Yes, but I will confirm those numbers. What else do we need?

Me: What tolerance do we have for failures? For instance, if 90% of the page times are below the values, is that acceptable? How many iterations can fail completely before the test is a failure? Etc.

Customer: I will get that information too.

Me: Good. Now we can configure the test to execute each of these steps the proper amount of times and measure the results independently. Then we can see if the system can handle the load. If not, we will tune it to see if we can make things faster.

Customer: What if we cannot tune it enough to make it meet goals?

Me: Then we need to decide if we want to try scaling out (or up) the system, or if we end the testing and say "this is what the system can handle. Anything above this will likely cause issues." Also, this is something that the business partner needs to decide. We can offer some options, but at the end of the day, it is the business partner that will be impacted by the problem so they need to decide how to handle it.

Customer: OK, I will work with the business team to get the expected load profiles and proper response times, etc. as well as the next steps if we do not meet goal.

Implementation

Now that we have defined some things, I have a set of Success Criteria that I can use to look at results. I can build a test harness to drive load and then execute tests to measure the application. Even though this list is simplified, here is what I would measure for this "engagement":

1. Web CPU should have CPU that is less than 70% on average with no sustained spikes above 70% that last for more than 30 seconds.
2. 10,000 logins with 95% responding in less than 3 seconds and less than 1% failing.
3. 20,000 searches with 90% responding in less than 5 seconds, 95% responding in less than 8 seconds and less than 1% failing.

I can (and often do) include a step load test that will drive the load at varying levels up to (and past) the planned peak load to help determine the breaking point by seeing when the first of the above criteria start to fail. That way I can determine the throughput that the system can handle. When I build the load test, I will set the test up with 4 different types of web tests (each web test represents the steps a single vUser will perform):

1. TEST1: This user logs in and performs one search, then one purchase

2. TEST2: This user logs in and performs two searches
3. TEST3: This user logs in and performs three searches

The load test will use “User Pacing” set up so that a single vUser will execute:

1. TEST1: 2 iterations per user per hour
2. TEST2: 6 iterations per user per hour
3. TEST3: 2 iterations per user per hour

The math then calculates out to a single vUser over the course of one hour will:

1. Login 10 times
2. Search 20 times
3. Purchase 2 times

If I setup the load test for a constant user load of 1,000 vUsers, you can see that I will drive the expected load over the course of an hour. I can also do step loads where I start at 200 vUsers, adding 200 vUsers every ten minutes and see where the system stops meeting the expected success criteria.

The results

Now I can run some tests and I know exactly what items to report back to management. If I run the step load test and I see that all criteria is met with 800 vUsers, but the criteria fails at 1,000 vUsers, I can tell them the capacity is currently between 8,000 and 10,000 users doing the work as defined above. I can also work with Subject Matter Experts to try to tune the system to make it handle more load.