Two Kinds Of Performance Requirements

Scott Barber

In my November column I explained the distinction between performance investigation and performance validation. For those of you who haven't had a chance to read it, I stipulated the following definitions:

Performance validation is an activity that compares the software being tested to the expectations that have been set or presumed for that product.

Performance investigation is an activity based on collecting information about the software being tested that may have value in determining or improving the quality of the software product.

Using those definitions, I generalized that there seems to be a common perception that "just like functional testing, performance testing is mostly validation"; that most performance testing projects necessitate both investigation and validation; and that with rare exceptions, performance testing is naturally investigative due to its common lack of predefined requirements or quantifiable expectations. I also observed that it is typical for a single performance issue detected during performance validation to lead to a pause, or even a halt, in continued validation. We closed with this simple rule: "Investigate performance early; validate performance last."

I'm guessing that you are curious about why I felt the need to recap last month before even introducing this month's topic—I know I would be! The answer is simple. For this issue, I knew I wanted to write about collecting and quantifying performance requirements. I've written on the topic several times before and have presented my theories and approaches at both practitioner and academic conferences on many occasions. Even though these papers and presentations have been well received, I've always thought there was something missing. I had thought what I was lacking might be supporting research data, and I kept hoping that one of the professors at the academic conferences would guide a young Ph.D. candidate toward conducting research to support my theories, but that never happened.

Then, as I was reading the final draft of my November column, it came to me. It was like the proverbial light bulb over the head of the cartoon character—I even noticed that I had thrust my index finger into the air as if to put an exclamation point on the end of my "Eureka!" moment. The point I'd been missing all along was that I was trying to treat all performance-related requirements as technical requirements. But they aren't; some performance requirements are usability requirements! Now, what kept me from figuring this out—while writing and presenting for the past four years about how most speed requirements need to be determined using a human factors approach considering user psychology, expectations and previous experience—is still a mystery, but it's one that I'm willing to leave unresolved. If you've ever been part of a usability study or even read about one, you'll immediately recognize user psychology, expectations and experience to be common and important factors to consider during such a study.

Before going into detail about how to handle these usability requirements that I and many others have been treating as technical performance requirements, let's consider how to determine which performance requiremens are technical and which are usability. Let's also explore how this relates to the distinction between investigation and validation.

During training, I break system performance into three categories—scalability, stability and speed—to simplify discussions, so let's use those categories here as we look at how usability fits in.

Scalability, also called capacity planning, is by far the most mature of the three categories in terms of requirements definitions, testing and predictions; there is probably an entire bookcase worth of relevant books detailing this activity. Regardless of where you go to research scalability and capacity planning, the material you'll find will be extremely technical. It will invariably presume either technical and well-defined system requirements (e.g., "The system shall be able to successfully process 2,000 stock trades per hour") or technical and well-framed test requirements ("Determine the number of trades the system will support during an hour before other requirements are violated"). It's pretty clear that the system requirements map to what we call performance validation, and that the test requirements map to performance investigation, and that both are technical.

Stability, which covers such specific areas as reliability, availability, robustness, recoverability and even some aspects of security, is also a technical category covering such requirements as "The system shall maintain five nines of availability," and "The system shall failover to a redundant backup system within two minutes of a detected server failure without corrupting any transaction data." Again, validation would be appropriate against these requirements, while investigation would be used to determine why validation of one of these requirements failed. While some

Scott Barber is the CTO at PerfTestPlus Inc. His specialty is context-driven performance testing and analysis for distributed multiuser systems. Contact him at sbarber @perftestplus.com.

aspects of stability deserve to be measured from a user's perspective, the requirements themselves are still fundamentally technical.

Speed, on the other hand, is where things get fuzzy. Some speed requirements are quite definable, quantifiable and technical. An example of a technical speed requirement could be as simple as "A stock transaction must be completed within 30 seconds of receiving a quote." In this case, "30 seconds" is a technical requirement, because the trade price is only valid for that period of time; on the 31st second, the trade request is canceled. Once again, this requirement fits with our previous discussions about investigation and validation.

Other speed requirements, however, are not so technical. For instance, I'd bet that the traders who use the application above with a 30-second transaction deadline would not be pleased with trades taking 20 to 30 seconds each; in fact, I'd hazard a guess that traders would be frustrated with 8-second trades. So while the technical requirement is an important factor to consider, when viewed from an "are we going to get/keep users on the system?" perspective, this scenario would likely yield a different requirement.

I think most of us are used to non-technical speed requirements that are defined from a user's perspective, such as in the following example: "Each stock trade shall complete in not more than 8 seconds, 95 percent of the time, when accessed by a client with a 128-Mbit/sec connection or greater." While this requirement is well defined and testable, I think it is where we may be off track. If we were to ask the person who wrote or mandated this requirement what purpose it serves, we'd surely get an answer like "This requirement ensures that our users don't get frustrated when using our site." On the surface, this makes perfect sense, but who decided that 8 seconds would ensure that the users of the system won't be frustrated? I know that I, for one, find an 8-second response time quite frustrating in many cases, but on the other end of the user spectrum, my mother wouldn't know how to act if she were to suddenly experience that kind of "blazing speed"—in fact, it might frustrate her that she doesn't have time to balance her checkbook while she's waiting for pages to load!

In the scenario above, by using validation, we can determine whether or not the pages meet the requirement, and we can use investigation to help track down why a validation test fails. But how exactly do we determine whether or not the requirement meets the unwritten intent of keeping users from becoming frustrated by slow response times? No amount of validation or technical investigation of the system will answer that question. To do so, we need to set aside the idea of creating technical requirements independent of the users for a moment and turn instead to usability studies to determine how fast these response times need to be.

Speed is where things get fuzzy. Some speed requirements are quite definable, quantifiable and technical; others are not.

A cursory Web search shows some fascinating usability research related to how users perceive download times, some dating back to 1998 and before. The most interesting to me is Usability News, a newsletter of the Software Usability Research Laboratory at Witchita State University (psychology.wichita.edu/surl/usability_news.html). On this site, I found research resulting in such findings as how the age of Web users affects their tolerance for delay times, and whether or not dial-up users have more tolerance for slow response times than users with faster connections. While I didn't find any real surprises in those results, I did encounter two other things that were far more interesting. First, this site includes links to a fantastic amount of reference material that I have barely started to sift through, and second, the actual procedures for how the research was conducted are included with the write-ups. The best part is that the only real challenge to duplicating the research is determining—and gaining access to—a reasonable sample of actual users.

The reason this is exciting to me is this: I have participated in usability studies before. I have researched how to design and conduct them, each time deciding that this was way too difficult for determining speed requirements. There were too many combinations and permutations of variables, too many ways to define user psychology, experience and expectations, and I had no actual training in these areas. After spending some time on this site, however, I realized that most of the research had already been done in general enough terms to be directly applicable to a variety of potential applications, and that the majority of the studies needed to collect data specific to an application or an application's users have already been designed, so all I'd need to do is conduct the study! All things considered, this is no more difficult than many of the other tasks we already do during a performance test.

So, if users' experiences are the reason behind a particular speed requirement, then why aren't we conducting a few usability studies to determine what that requirement really is? It looks like it's really not that difficult to conduct the research needed to determine speed requirements based on usability—research that will give the stakeholders enough information to make educated decisions about application performance. While this sounds simple, my guess is that this shift in thinking about requirements will take some time to catch on. Then again, maybe these usability studies are really nothing more than another performance investigation—only in this case, we are determining requirements for validation by investigating *users*.

It looks like now it's time for me to walk the walk and test this theory. I'll let you know how it turns out.