Analyzing how effective your testing efforts are, and working to improve processes

Tips for analyzing the overlooked test data

As testers, we think about delivered software quality and the quality of our work all the time. It's very important to have a set of quantifiable measurements in place so that you can gauge the effectiveness of your efforts to ensure that the software under test meets the intended business objectives.

by Mush Honda

While the traditional context of test data analysis mostly deals with the data used during testing, it is important for testers to also focus on the analysis of data generated as part of the testing performed by the team. That's the kind of 'test data' that's often overlooked.

If we want to improve the way we test, we need to start looking at this data. There are some important aspects that cannot be ignored when we start to ask questions, like how much effort did the test team put into completing specific tasks, how many hours were spent on each cycle or project, and how close to the estimated timeline did we come?

By analyzing this kind of data we can calculate the ROI (return on investment) of the tools we're using, figure out how to estimate more accurately in the future, and streamline our processes.

Focusing in the right place

A test plan should not be set in stone. If you want to make sure you are focusing your limited resources in the right place, you need to continually reassess your plan. Which functional modules have the highest number of defects? Is it possible that something you marked as low priority should actually come first? Can something else be moved onto the back burner? Flexibility is essential if you want to get maximum value from your testing efforts.

You also need to think carefully about where problems might hide. Should you really be focusing your regression on an area where you found a lot of issues, or will you get more value if you look at interdependent areas? If developers have pinpointed a specific issue in that area, then that function probably works well. It might make more sense to look at re-

lated features that haven't been tested as much. If you gather data on this, you can build a set of rules for your regression testing, so that you can focus on the places where you're likely to find more problems.

Report on usability

Testers should always give usability feedback, even if it's beyond the scope of the functional test they're conducting. This kind of feedback represents vital information from expert software testers emulating the end users and can give developers and product owners a new perspective that can help them improve the final product.

Gathering usability data from testers and collating it gives you an idea of where the software needs further attention. It often reveals low-hanging fruit in terms of easy improvements that can have a big impact on the final quality of the software.

Calculating return on investment (ROI)

How do you know that automation scripts are saving you time? Why is it better for testers to write manual tests in the ALM rather than Word? If you don't measure the effective-

There are times when sophisticated tools might be too complex.

ness of the techniques and tools that you use, you can't say for sure that they provide any advantages.

There are times when sophisticated tools might be too complex for the job at hand. You may also discover that testers are wasting a lot of time working on complex tasks with tools that aren't really fit for that particular purpose. You should examine how testers interact with their tools and what level of manual effort is involved in the process.

You might discover ways to improve interactions, identify alternative strategies, and score major efficiency gains. Something as trivial as analyzing your tools and processes would often generate solid ideas on how to save time and improve aspects.

Common sense

We accept that we can only really improve software quality by measuring the right things, but we fail to apply the same logic to the practices, processes and tools we employ to complete our testing efforts. By assessing our approach and taking action to focus in the right places, we can realize concrete improvements in terms of efficiency and widen our overall test coverage significantly.

To think of test data analysis too narrowly and just focus directly on software quality will inevitably lead to a missed opportunity to improve the way we test. It just requires a slight change of perspective. Analyze the way you test, make changes and measure their impact, then rinse and repeat. Ultimately, improvements in the way we test will also have a positive impact on final software quality.

Mush Honda is Vice President of Testing for KMS Technology, a provider of IT services across the software development lifecycle with offices in Atlanta, GA and Ho Chi Minh City, Vietnam. He was previously a tester at Ernst & Young, Nexidia, Colibrium Partners and Connecture. KMS services include application management, testing, support, professional services and staff augmentation.