Audience: Developers, maybe project managers

Big Brother Testing

Author: Bill Tuccio

I felt I had reached an epiphany in my career as a developer recently when a customer accused me of being a stalker. Ok, you're thinking, "A developer who's a stalker – not a surprise!" In this instance I knew from the history of the project the customer was very busy and might not recognize the pivotal role they played in requirements validation. My approach was to design into the solution a simple, yet robust logging solution which allowed me to verify the customer was fulfilling their testing responsibilities.

Design Approach and Environment

The application was a web application to track travel itineraries. Since the web application was driven by a SQL database, it was a simple chore to make a "history" table which recorded the data entry and edit activity for any itinerary. The history table would be useful to the users in production, since they might want to see a history of edits, and it would also be useful to support personnel to identify problems. During testing, it also allowed me to watch activity of the users.

A word of caution, sometimes I hear developers of web applications want to use IIS logs for traffic tracking. I have almost always found this to be a wasted effort. With storage space and SQL performance, writing to dedicated logs owned by the application eliminates semantic ambiguity.

Testing Stalking

When the application was finished and unit tested, I contacted the user to let them know they could begin testing. Like many small shops, the user role was not only acceptance testing but also iterative fixes to usability and business logic. I attempted to guide the users to record their results and test cases, but didn't have a lot of confidence they would take detailed notes.

After I notified the users to begin testing, I kept a SQL query to the history table open on one the Windows on my desktop. Every now and then I would re-query the history table to see if there was activity. After a day, when there had been no activity, I contacted the user and asked them when they would get to testing. This initial prodding helped the users try to arrange their schedule to fulfill their testing obligations.

Eventually I saw activity in my SQL Query! They were testing! The logs allowed me to open up the transactions they were working on in the web interface and see if they understood the interface paradigms. As I compared their activity to the results being produced, I could gauge without their direct feedback where usability flaws might exist.

Early on, the interface flaws were easy to identify and the changes would be dramatic enough that a new release would be needed before fruitful results would be produced. When I saw this point being reached and the changes to be made were obvious to me, I would either call or email the user to discuss what I had observed. I made the contact for two reasons, one was the direct reason of not having them do work once a discovery had been made, but also to try to create a sense of urgency in their testing effort and avoid the project stalling. By integrating prior experience on the project (the two month project had been going on for nearly seven months with little progress) with a technical solution to logging, I was able to add in a project management philosophy to guide untrained "testers" through some semblance of a test plan. Even though the users never provided a nicely filled out test matrix of testing results, I was able to gauge the coverage of their testing through logging and get the feedback I needed through verbal communication while the thoughts and experiences were still ripe in their brain matter.

Conclusion

The measure of a good subject matter expert (SME) is likely how busy they are doing their job – which almost by definition is not engaging in software development. So, if you do find the right SME, do anything you can to value their time and read between the lines. Stalking by logging is one way to do it. At least they will know you are watching. Creepy, huh?

About the author:

Bill Tuccio has been a software developer and hands on project manager for over 15 years. He was the CTO of movietickets.com and was the lead programmer on stickyminds.com. Today he programs Microsoft SharePoint solutions for a defense contractor.