There's Nothing Like the Real Thing

by Jonathan Kohl

Because I am a software tester, my family, friends, and acquaintances frequently complain to me about poor software quality. They tell me that the software they use is unreliable, difficult, and generally gets in their way as they try to complete their work. Once in a while, people tell me about software that saves them time and makes their lives easier. The positive stories give me hope for the future of software development. The other stories are not so hopeful. "Why don't they develop dependable software?" people sometimes ask.

That's a difficult question to answer; there are many reasons why software fails. One reason might be that some teams value ideals over reality. Convinced that the process they follow and the tools they use are the keys to their success, teams don't seek hard evidence from their projects to confirm their decisions and actions. They measure success by the degree to which they follow a process and forget to measure how that process helps them achieve the results they need.

Sometimes teams tell me that a process, practice, or tool is unquestionably good, as if the goal of the team is to implement the tool or practice itself. In these cases, there is little motivation to test whether the results they believe they are achieving match what is actually being achieved on the project.

In one instance, a programmer on an Extreme Programming team was astounded that my testing revealed bugs in the software the team was developing. The team had brought me in prior to a release to do some exploratory testing. As test-driven development (TDD) practitioners, they had an impressive array of automated unit and acceptance tests. They had gathered metrics to show their process adoption progress and were confident in their approach. They hoped I would be an insurance policy to confirm that they were ready to ship their software. We were collectively surprised when my testing in a production-like environment revealed showstopping errors. I began to investigate why this might be the case.

A look at the automated unit tests revealed that many were of poor quality. They were often very basic and relied too heavily on mock objects that were used to speed up builds. The automated acceptance tests were also quite simple, and the customer representative ran manual acceptance tests based on the automated tests. Furthermore, they had stopped deploying builds in a production-like environment, preferring to have the customer test in the development environment.

My programmer friend and I were discussing why my tests failed and his team's automated tests passed, and we were getting frustrated with each other. He kept saying that the code *had* to work. He had the tests to back it up, and their methodology was proven. "Your tests are running against an ideal, while my tests are using the real thing," I replied. We were at an impasse. With nothing left to say, I began singing Marvin Gaye's "Ain't Nothin' Like the Real Thing." As I sang the title line in the chorus, he looked at me like I was crazy. He then turned around, sat down, and began setting up connections to a production-like environment that could be used during development. The song had succeeded where my words had failed.

Once he and the other programmers supplemented their automated tests with real-world scenario tests in a production-like environment, I saw a marked improvement in the software they delivered. They also began using manual test results to qualitatively measure their TDD process so they could improve the tests they were creating.

Adopting a tool or a process is hard work, and it's a good idea to measure ourselves to see if we are on the right track. However, we need to remember to check whether there is a difference between what we believe the results might be and the actual results we achieve. There's nothing quite like testing our project ideals against the real thing. {end}

Jonathan Kohl is a software testing consultant with Kohl Concepts Inc., based in Calgary, Alberta, Canada. Jonathan writes about and speaks on software testing. Read more of his work at www.kohl.ca. Contact Jonathan at jonathan@kohl.ca.