Is Testing a Service?

Fiona Charles and Jerry Weinberg

Originally published in the April 2012 issue of Tea-time with Testers.

This article arose out of an email conversation between Jerry Weinberg and Fiona Charles. Fiona had just participated in POST, the Calgary "Perspectives on Software Testing" peer conference.

FC:
I just spent the weekend at POST, the Calgary peer conference Lynn McKee and Nancy Kelln founded after coming to TWST. It was the third POST, and the first one I've been able to make it to. It was just great, though I think I upset the applecart a little with my presentation. The topic was "testing as a service". This comes from *Lessons Learned in Software Testing*, where Cem Kaner, James Bach, and Brett Petticord wrote that testing is a service to the project. It's a fashionable stance for context-driven testers. I argued that it's a very dangerous thing to say from a practical standpoint, because it in fact puts testing outside the project. Doing that emphasizes our second-class status and can even lead to a belief that the testing service is optional. Testing may be appropriately optional in some contexts, but in most cases I think separating testing from "the project" does more harm than good.

I prefer to say rather that testing is an integral part of software development, and that software development is a service to its business stakeholders.

Of course, there are other ways in which you can see testing as a service, e.g., if you are selling testing services, as I did for many years and do now to some extent. I talked a little about that, too.

JW:
I would say testing is a service in the same sense that, say, my team of physicians did a service for me during my cancer period.

In that case, the service-providers are higher-status than the patient. Could be that way in testing business, if we wanted to make it that.

Or maybe we want it to be a partnership, like a couple happily married for 50 years.

FC:
I want the whole project to be a partnership.

I don't have a problem with "testing is a service". I do have a problem with "testing is a service to the project". To me, that separates testing from the project—unless project management, business analysis and programming are also a service to the project. They may well be, but the people who do those things don't describe their work that way.

JW:

It can be either way. To take a less controversial example, providing network access and capacity could be within a project or a service to a project. For testing, there are pluses and minuses for each way.

Some people believe testing (at least some) should not be in a position to be influenced by project management.

Other people see testing ideally as a cooperating function, fully integrated with the rest of the project all of the time.

In general, I prefer the second, but not if management is not capable of leaving them alone to do their job, not pressuring to give the answers that make the managers look good.

FC:
In my experience, management that pressures testers to give rosy answers (i.e., lies) will do that whether or not testing is integrated with the project team. It can be helpful for testers to have a separate reporting line, but when organizational cultures are rotten the rot usually starts above the level in the hierarchy where the development and testing lines diverge.

JW:
You're right about that, yet though the rot develops there, it's usually easiest to detect early at the tester/developer level.

Anyway, I'd rather not work with organizations where I have to dig behind screens to see the rot.

FC:
The idea of tester independence has come full circle in the course of my career. Early on, we fought hard for organizational independence—not primarily because of pressure to make people look good, but because managers had programming backgrounds and tended not to understand or care about testing, meaning they often didn't allow us to test enough or well. I've frequently had to deal with idiocies like "your testers are asking too many questions and slowing down development" or that old song "your testers are finding too many (or the wrong) bugs".

JW:
Yep. My career, OTOH, started earlier, when the idea of separating testing from development basically didn't occur to us. Testing was simply part of the software-building job, though we might have members of the team specializing in testing for a particularly difficult-to-test part of the project. But they were always under the single project manager, one way or another.

FC:
One problem (or at least a perceived problem) was that once testers achieved organizational independence, the separation often hardened into structural antagonism. This was reinforced by the counter-productive idea—promoted by both managers and testers—that testers were gatekeepers. Among other detrimental effects, many testers became judgemental, burning with a righteous belief that only they cared about quality. I've always challenged this in my teams and worked to promote cooperation, but it

seems to have pervaded many organizations. On the big project I worked on in the UK, our government customer was convinced that it would create a conflict of interest if my test team worked closely with programmers.

JW:
<sigh> We've known that scenario several times.

FC:
The push to integrate testers more fully into project teams has come partly from agile, with its emphasis on collaboration, and partly from people like me who have fought for decades to get testers involved earlier in projects when they could really contribute to getting better requirements and more testable code, as well as getting their hands on code to start finding bugs earlier.

JW:
We had already won that battle in 1960 or even earlier, without realizing we were in a battle.

FC:
I don't believe that managers are any smarter about testing than they ever were, but I also don't believe the organizational separation has worked very well to alter their behaviour. And it has certainly contributed to bad behaviour among testers and bad feeling about testers among programmers. I don't know any easy answers, but I do think testers have to be tough as well as skilled. We need to maintain independence of mind while collaborating fully on software projects.

JW:
I can attest to that. (BTW, if we do collaborate on an article, we're going to have to battle over the spelling of "behavio(u)r.")

And testers have to be skilled at reframing, but be careful you don't put all the burden of change on testers. That's been part of the battle all along.

Still, all we can do as testers is change what we can do, and hope that others will follow. That does happen (sometimes).