What’s in a Name?

Originally published on stickyminds.com Oct. 22, 2008.

> What’s in a name? That which we call a rose
> By any other name would smell as sweet.
> (William Shakespeare. *Romeo and Juliet. ACT II Scene 2)*

What do you call your professional role? Are you a tester? A Quality Analyst? Quality Control Specialist? Quality Engineer? (These “Q” titles always seem to be capitalized.)

Does the name matter?

It’s true, as Shakespeare said, that a rose would still look and smell just the same, even if we suddenly started calling it “skunk cabbage.” But changing the name would influence how people perceived the plant. Those who’d never encountered a rose would be much less attracted to the idea of something called skunk cabbage (though skunk cabbage happens to be quite an interesting garden plant in its own right.) If they wanted fragrant flowers, they’d be less inclined to buy skunk cabbage than roses.

What if we turned this around? If we decided that skunk cabbage should be called roses, I think we could predict the result. Their essential nature wouldn’t change, and they’d still smell skunky—not at all like roses. But in this case the unwary public who wanted nice-smelling flowers could be enticed into buying a plant that doesn’t smell pretty.

So there is something in a name. In fact, naming is a very powerful act.

Marketing professionals know this, of course. So do managers and others who want to change perceptions in their organizations. When an organization sees testers as something like “the people who walk behind the programming elephants with shovels,” then it’s understandable if some people want to change that perception. So, they change the name, expecting perceptions to follow.

Jerry Weinberg calls this “name magic,” a term from cultural anthropology. Here’s a riddle he uses to illuminate name magic.

Q: If a dog has four legs and you call its tail a leg, how many legs does the dog have?

A: Four, of course. Calling a tail a leg doesn’t magically make it a leg.

The power of name magic is that names carry associations that resonate in our minds. When we give a thing a name, we imply all the attributes of that name. If it’s an inflated name, like “engineer,” we are attempting to inflate the role. But calling a tester a Quality Engineer doesn’t make the tester a person who engineers quality into software. No tester can do that.

In my plants example, knowledgeable plant-lovers would continue buying and planting roses for their sunny gardens and skunk cabbage for their bog gardens, whatever they were called. That’s because adopting marketing tactics and changing a role name will only influence the perceptions of those unfamiliar with the actuality of that role. If your organization's culture causes the programmers to see testers as skunk cabbage, then testers are still going to be skunk cabbage even when they're called roses—or Quality Engineers.

Name magic doesn't make the problem disappear. So why not call yourself a tester and address the problem directly? That's harder than attempting name magic, of course.

How can we do it? Maybe we'll never really get anywhere until we reverse an act of name magic that happened early in our industry's history.

We testers have an honorable calling. We have specialized skills and a mindset that is unique—and uniquely valuable—on software projects. But many programmers and managers don't believe we can do anything the programmers can't do just as well. For these people, programming is the highest calling, and the techies who write code are the only essential resources in software development.

I think one reason for this is a historical piece of name magic that we now take for granted. We started calling programmers "developers."

This occurred to me a few weeks ago at the Agile 2008 Conference. Listening to agilists question how—or in some cases even whether—to integrate testers on their teams, I wondered yet again why testers are still having to struggle for acceptance as essential players in software development. It's not a new struggle; this is just the latest incarnation. In session after session, I kept hearing about the "developers" on agile teams. At some point I started thinking, "But what's a 'developer'? Aren't we all involved in software development?"

Software development consists of analysis, design, programming, and testing. These are all difficult intellectual activities that cannot be done effectively without a high degree of skill. The skills needed to analyze requirements, design user interfaces, or test are not the same skills required for programming. It isn't reasonable to expect one person or one group of similarly skilled people to possess them all to the level required to develop complex software. That's why specialist trades have evolved in our industry.

The reality is that we are all "software developers," whether we are business analysts, architects, designers (including the people Alan Cooper calls "interaction designers"), programmers, or testers. But by calling programmers "developers," we imply that all of software development can be done by programmers. Too many managers (most of whom started their careers as programmers) believe this, and so do a lot of programmers—perhaps most.

Calling programmers "developers" is an instance of name magic that inflates the programmer's role and devalues everyone else's. It reinforces a class system where programmers are the aristocrats and all the other trades are peons, perceived to be lower skilled, interchangeable, and expendable.

I was encouraged at the Agile Conference to hear that testers are finding roles on some agile teams and that this is even institutionalized in some organizations. I heard some agilists say they're beginning to realize that business analysts have an important role, too. As experience with agile projects grows, we will probably see an increased return to specialized roles and the benefits that this will bring to the resulting software.

But unless we deal with the class system in software development, we testers will keep fighting the same battles over and over again. Reversing the "developer" name magic won't solve the whole problem, but I think it's an essential step.