



**Michael James**  
Michael James is a software process mentor and Certified Scrum Trainer, focusing on the engineering practices that enable Agile project management. Having worked in the software industry for more than 20 years as a software developer (formerly “architect”), he has experience in automated testing that predates the Extreme Programming movement; formal, phased, high-ceremony processes based on DOD-STD-2167A; chaotic non-processes of the dot-com era; and Agile processes including Scrum and XP.

# Scrum and Quality Assurance

Posted by Michael James on September 24, 2007 at 5:31 p.m.

A recent email:

*[x] and I are in charge of quality and we put folks from our teams on to the Scrum development teams to help ensure quality. Some of the key questions we have are regarding how we ensure quality. How do we know that the quality is adequate with the Scrum process? What are key ways we can report and track the quality during a Scrum development cycle?*

Remember how [Tito Puente](#) moved the drummer from the back of the band to the front of the band? That’s what Scrum allows you to do with Quality Assurance.

The Scrum framework itself is silent on engineering practices. Scrum does require you to build a *potentially shippable product increment* every Sprint. “Potentially shippable” generally means you could confidently get it out the door within one stabilization Sprint or less. A stabilization Sprint is not a testing Sprint. Every Sprint is a testing Sprint. That means the team gets [zero credit](#) for work that isn’t tested.

[No one said Scrum was easy](#). If your testing is any good (at least [more than a bunch of unit tests](#)), you may find it difficult to get this done every Sprint. It’s a lot of extra work. This responsibility is owned by the whole team, because the whole team won’t get credit if it’s not done/done/done.

To make this explicit, [our courses](#) encourage you to negotiate a robust definition of “done” (or “acceptance criteria”) for every Product Backlog Item. Write the acceptance criteria right on the card. This means taking on fewer Product Backlog Items per Sprint. Welcome to real life. A smaller amount of thoroughly tested work is worth more to us than a larger amount of low quality work. Instead of reporting and tracking regression failures, we fix everything we broke in the same Sprint we broke it or the item is not demonstrated at the Sprint Review Meeting. Sometimes it’s slow-going, but, this way, we always know where we stand.

If your team gets sick of all the extra work (which increases every Sprint as your codebase grows) and is willing to learn new skills, it will automate as much as possible: end-to-end (system) testing, load testing, “negative testing,” security testing, and so on. When anyone can reach the “push to test” button and get rapid feedback as to whether it’s broken, they can make more radical design changes than they would otherwise because they’re not flying blindly. Another useful engineering practice we borrow from the eXtreme Programming folks is [continuous integration](#).

Remember the principle behind the practice of combining Quality Assurance skills with design/coding skills in a single team is to tighten the feedback loop. Don’t track bugs; detect them when they’re created and fix them! (Of course, if any slip through,



Danube Technologies, Inc.  
520 SW 6th Ave, Suite 940  
Portland, OR 97204 USA  
1.888.5.danube  
T: +1.503.248.0800  
F: +1.503.248.0801

you can create Product Backlog Items for them to be prioritized like all other work.) The traditional practice of waiting until the end to test wreaks havoc on our release planning. Maybe we can predict how long it takes to test, but how long will it take to fix the things we find during testing? And then how long will it take to fix the things we broke while fixing those things? With the long feedback loop of the [waterfall process](#), we can't predict how long it will take for the ball to stop bouncing. A flimsy definition of “done” (in Scrum, or any other approach) leads to an *unbounded* amount of work before we can ship.

The software industry has an imbalance of skills, personnel, and clout. There hasn't been much career incentive for our best and brightest to get good at Quality Assurance. I visited one company that gave their “developers” the desks near the window while the “testers” were clumped toward the center. (They nearly threw me out of that window when I suggested grouping by cross-functional teams instead.) Scrum can change that when combined with Agile engineering practices and a robust definition of “done” for Product Backlog Items.

--mj

Software Process Mentor

(former embedded systems design engineer and embedded systems verification engineer)

The logo for Danube Technologies, featuring the word "danube" in white lowercase letters on a blue rectangular background.

Danube Technologies, Inc.  
520 SW 6th Ave, Suite 940  
Portland, OR 97204 USA  
1.888.5.danube  
T: +1.503.248.0800  
F: +1.503.248.0801

Copyright © 2007-2008 Danube Technologies, Inc.

A small version of the Danube Technologies logo, consisting of a blue square followed by the text "danube.com" in white lowercase letters on a blue background with horizontal lines to the right.