

---

# I've Seen The Future and SCM is Not There!

Posted by Damon Poole - 2007/05/12 14:27

---

Hello died-in-the-wool SCM person with your branching strategies and your release schedules and your fancy SCM terms. I have seen the future, and we are not in it! The time to save ourselves is now!

Ok, so that's just one possible future. The need for good SCM and good SCM people is not going away by any means, but things are changing fast. If we embrace this change, we have the opportunity to have a positive influence on it.

I work for an SCM tool vendor, so I'm involved in a wide variety of companies that are in the midst of change every day and I'm seeing this future more and more frequently.

Let me give you an example. I was recently visiting a large mainstream company that is in the midst of an SCM tool evaluation. They have 200 developers split between two sites, one on the west coast and one on the east coast. They maintain a very broad product line. They are using one of the market leading tools. I say they are in the midst of an SCM tool evaluation, but really they are in the midst of a shift to Agile Development and the change of SCM tool is just part of it.

They have hired an Agile Coach to help them change their software development process. Here are some of the things the Agile Coach said. Not all Agilists feel this way, but this is just one example of what the future holds:

"When I come into a company and I see that BigSCMTool is in use, I tell them that job #1 is to get rid of it."

"They need to all be working on the trunk with no branches."

"The problem with BigSCMTool is it doesn't do X, Y, and Z."

I wasn't about to disabuse him of his notions on BigSCMTool because there were plenty of other issues with it, but the particular problems he mentioned had to do with the way the tool was configured and not the tool itself. It could just have easily been configured to handle X, Y, and Z.

Over the course of the past year, at least 3 of the BigSCMTool experts had been let go.

Do you see any problems here? :woohoo: It is clear that many Agilists are not SCM experts and thus not familiar with all of the benefits of good SCM, nor are they experts with the various SCM tools out there. As a result of the experts being let go, the duties have been distributed to various developers which has had some success, but it is clear (to me) that there are now more problems than before and if they do move to all-mainline development they have even more problems in store.

This is not an isolated example by any means. Another popular Agile technique is to use 3x5 cards or in the more modern shops, post-it notes to replace issue tracking systems. I kid you not. I've seen it even in large shops; giant swaths of multi-colored 3x5 cards waving gently in the air currents as one walks down the corridors.

And let's not forget the first principle of the Agile Manifesto: "Individuals and Interactions over Processes and Tools". I have no beef with individuals and interactions. Software development works best when people are interacting well. However, there is such a thing as going too far.

How did this happen? Well, I think we need look no farther than the mirror for part of the answer. I admit that showing the value of SCM is hard, especially in the face of the perception by many developers and managers that "it is just moving files around, how hard could that be?" But that's no excuse when we blame them that they "just don't get it." We need to try harder.

And then there is the other reason. There really are major benefits to Agile Development! The people that have realized this are doing everything they can to get to those benefits. Unfortunately, that sometimes means that they throw out the baby with the bathwater on their way to those benefits. It is much easier for many developers and managers to see the benefits of Agile Development than it is for them to see the benefits of merge tracking, branches, and SCM experts.

As SCM experts who know the benefits of SCM to the software development process, what can we do? We can remember that the purpose of SCM is to serve the development process and we can figure out how to apply our skills (which are rare and valuable) in ways that are recognized as rare and valuable. We can take a leadership position, advocate for Agile Development and be looked to as experts instead of getting involved after the shift has already begun.

Here are some ideas. Advocate for and implement continuous integration. This will have immediate and obvious benefits for the developers. Get certified as a Scrum Master. Bone up on Agile Development and start advocating for it in your organization, thus taking a leadership role. Or, align with the developers and managers that are already advocating for it

---

in your organization. In any case, there is a lot involved in moving to Agile Development and IMHO most of it will be best done by an SCM expert, especially if they are leveraging their SCM skills and translating them into an Agile context.

Here are some good places to start:

SCM Patters for Agility (<http://www.scmpatterns.com/index.html>)  
A great resource that covers the intersection of SCM and Agile.

Integrating Agile Development in the Real World, Peter Schuh  
An excellent survey of most of the Agile methods.

Extreme Programming Explained - 2nd Edition, Kent Beck  
Whether you agree with the idea of 3x5 cards or not, Kent's book is a short and easy read that is well worth the effort. A seminal book on Agile development.

Lean Six Sigma, Michael L. George  
I read this book because I was interested in learning more about Six Sigma. However, I selected this book because the idea of "Lean" was appealing. The way that the problem and solutions for finished product sitting on a factory floor is explained immediately suggested parallels to finished features sitting in the source repository waiting for release.

Lean Software Development, Mary Poppendieck and Tom Poppendieck  
These folks have done an excellent job of taking the lessons of Lean manufacturing, pioneered by the Toyota Production System, and translating them to software development. I read "Lean Six Sigma" first and on my next trip to the book store "Lean Software Development" became an obvious choice. Reading this book was the tipping point for me.

Cheers,

Damon Poole  
CTO, AccuRev  
blog at <http://damonpoole.blogspot.com>

=====