JOIN OVER 100,000 PROS! Join Log In | Member Benefits

STICKYMINDS

Agile Testing DevOps Security AI

Are You Agile Enough? ARTICLE

By Steve Berczuk | March 25, 2010

SUMMARY

Being agile is a means to an end; your goal is to develop better software more effectively, not to be able to wear a "We are Agile" badge. If you're considering adopting agile, you are probably doing do because your current approach isn't getting you where you need to be so it's worth giving the 'by the book' technique a shot before you try to adapt an agile method to your circumstances.

I recently read an article in SD Times about how organizations tailor agile processes to fit into their environment, rather than feeling a need to be dogmatic. Adapting a process to work in your environment is very important to being successful. It's also important, however to understand how the variations you are making help you move toward your goals.

Being agile is a means to an end; your goal is to develop better software more effectively, not to be able to wear a "We are Agile" badge. If you're considering adopting agile, you are probably doing do because your current approach isn't getting you where you need to be so it's worth giving the 'by the book' technique a shot before you try to adapt an agile method to your circumstances.

This especially applies to when you consider omitting practices. Like many approaches, there is a synergy between the the core agile practices; any one can help you be better but the big wins come when you do them all.

The essential parts of an agile process in my experience are:

- Feedback, with a goal of continuous improvement.
- Honest evaluation of why you're doing and why.
- A (periodically) stable codebase so that you deliver functionality to customers quickly
- A set of goals that lead to customer value

We use cookies to ensure you get the best experience on our website. Accept

Read More

Agile methods define some techniques for achieving these goals. Scrum has you doing sprint planning, daily scrums and sprint reviews. You need technical practices like unit testing and continuous integration to verify that the code does what your project plan expects it to do. There are other ways to achieve these goals, but if you are trying to change how you work, a template is helpful.

When I read something like this finding reported in the article:

Of agile's core tenets—daily standup, iteration planning and unit testing—VerisonOne found in its fourth annual "State of Agile Development" survey that 69% of the 2,570 participants adhered to these three things.

or

Even a variation of Scrum exists. Known as "ScrumBut," it was dubbed for shops that don't fully comply with the methodology. It is for those who say, "We are doing Scrum, but..." While some people may view this as non-agile, others argue it's simply a customization of an agile process to work better for that particular company and its structures.

I wonder what these teams are they doing instead of the core "standard" practices to achieve the goals of agile. The argument about whether those practicing "Scrum-But" are "agile" misses the point of the label, which is to highlight the areas where teams can work to improve their productivity.

My former colleague Bruce Eckfeldt was quoted in the article as saying:

"I see the methodologies as a continuum, and at the end of the day it's all agile with the same principles and practices," ... "There's nothing set in stone on how to do something. You're always looking to improve."

If you're not improving as fast as you like, consider what steps in your process you are skipping. It's really hard to grasp the concepts behind agile until you are disciplined and try to apply the practices. If you have another approach that gets you to the end goal, by all means do it, but at the core, agile methods don't define a lot of detail. The basics are mechanically easy. The hard part is getting comfortable with the transparency agile methods require.

"Agile" methods define a set of goals and techniques to attain these goals. If you are trying to adopt agile, but don't want to adopt all of the practices of a method, it's fair to say that your organization isn't ready for a given practice, but be honest about it. Without the honesty and transparency, you're missing the key difference of agile methods.

As to the relationship between textbook and practical agile when adopting agile methods, Bruce is quoted as saying:

"things may be reasonably pure," but then people start to see what works or not and begin "to look beyond the textbook versions of agile."

The trick is not to customize too soon. Until you give the process a fair trial, there is a

We use cookies to ensure you get the best
experience on our website.

to deliver as quickly as they can. And in the end, organizations and people need to change, whether they are adopting agile, or any new process.

Before adapting you need to understand the core values or agile before you risk throwing the benefits away. You need to understand why you want to make the adaptation and honestly express the reason. And most important of all, you need to be clear and consistent about your goals. For example, if you want to encourage unit testing, you need to be clear that unit testing is encouraged, and accept that initially slightly slower progress is possible and OK if people are learning to write tests. Nothing can kill a change endeavor more quickly than inconsistent messages.

Being dogmatic for the sake of method purity isn't useful. Following the process until you understand the benefits of the practices can be very useful as a way to facilitate change. I'll end with a brief story.

When I first heard of XP, at an OOPSLA conference I came back to work and suggested that our definition of "done" include unit tests. The other person on my team grumbled each time I asked "did you test that?" The grumbling continued until a couple of days later when, being presented with a problem report from our QA team, he was able to diagnose the source of the problem in minutes with the help of unit tests. "Cool," I head him mumble as he found and fixed the problem.

Had we not given the discipline of testing a good shot because we believed that unit tests didn't help, we would not have understood why the practice mattered. If you think that a method has value, try it, learn, then adapt. The opposite, adapting before you try, is less useful.

Topics: agile transition,

ABOUT THE AUTHOR

Steve Berczuk

Steve Berczuk is a Principal Software Engineer with experience as a manager, Scrum Master and technical lead in Boston, MA. The author of *Software Configuration Management Patterns: Effective Teamwork, Practical Integration*, he is a recognized expert in software configuration management and agile software development. Steve is passionate about helping teams work effectively to produce quality software. He has an M.S. in operations research from Stanford University and an S.B. in Electrical Engineering from MIT, and is a Certified ScrumMaster. Contact Steve at steve@berczuk.com or visit berczuk.com and follow his blog at blog.berczuk.com.

UPCOMING WEBINAR

We use cookies to ensure you get the best experience on our website.

LETS HANG!

Log In or Register to post comments

USER COMMENTS

0 comments

Join the crew for FREE! Access exclusive content, contribute, and more.

See All Member Benefits

We use cookies to ensure you get the best experience on our website.

We use cookies to ensure you get the best experience on our website.