Agile in the DoD: Eighth Principle

featuring Mary Ann Lapham and Suzanne Miller

Suzanne Miller: **Suzanne Miller**: Welcome to the SEI Podcast Series, a production of the Carnegie Mellon University Software Engineering Institute. The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University in Pittsburgh, Pennsylvania. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts.

My name is Suzanne Miller. Today, I am pleased to introduce you to myself and Mary Ann Lapham. Welcome to our ongoing series exploring Agile principles and their application across the Department of Defense. In today's installment, we explore the eighth Agile principle: *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely*.

First, a little bit about myself and Mary Ann. Mary Ann's work focuses on supporting and improving the acquisition of software-intensive systems. For Department of Defense programs, this means assisting and advising on software issues at the system and/or segment level. In addition, she leads research into software topics that are germane to acquisition of software-intensive systems, including the SEI's research and adoption of Agile and Lean methods in regulated settings like the DoD.

My research focuses on synthesizing effective technology transition and management practices from research and industry into effective techniques for use of Agile and Lean methods in regulated settings. This is our eighth podcast exploring the real-world Application of Agile principles in Department of Defense settings. If you like what you hear today, there are already seven podcasts in this series, and we will eventually have 12, one for each Agile principle behind the Agile Manifesto. As a reminder, the 4 values and 12 principles of the Agile Manifesto can be found at www.agilemanifesto.org. Today, Mary Ann, let's talk about maintaining a sustainable development pace. What are challenges to that goal in DoD Agile settings that we have seen? She is smiling.

Mary Ann: I am smiling. I want to start with just software in general. Most of us who have been in the software field for even a little while are used to the, *We are going along. Things are not going well, but that is OK. We have time*. Then, panic sets in about six months--if you are lucky--before the product is due. Then, the overtime starts and the horrible hours.

Suzanne: They call it the death march.

Mary Ann: The death march, yes. The death march ensues, and people are burnt out and exhausted. If you are lucky, you do deliver on time, but it is this horriblehead-on-fire pace.

Suzanne: Firefighting is what we call it.

Mary Ann: Firefighting. Yes, and a hero comes in and saves the day. Well, that gets old.

Suzanne: The hero is really, really tired at the end of that.

Mary Ann: It gets old. Step back now. If you are doing Agile and Lean principles correctly, you avoid all of that.

Suzanne: What are some of the things in Agile practices that help you with this?

Mary Ann: OK, first off, you know how much your team can do every sprint.

Suzanne: We call that velocity.

Mary Ann: Velocity.

Suzanne: You know what the team's velocity is, and how much they are capable of.

Mary Ann: Right, now velocity is for each individual team.

Suzanne: Right, right.

Mary Ann: So you know how much they can get done. If you say one team can do X—(they do it in story points). That is, you know how they count the size and complexity of work. So, one team can do 30 story points every sprint, another can do 50 because they are more experienced, or whatever.

Suzanne: A lot of different skills.

Mary Ann: Lots of different skills, and then another can do 40. You figure out over your number of teams you have how many story points you can, in the first sprint, get done across your team. Then you figure out, *How much is the whole project going to be?* Then you can figure out mathematically, without even getting off your chair, how long it will take. If it is not going to get done when you need it to get done, then you can figure out, *Can we add more teams?*

There is one program we are aware of--they are into mostly what we would probably consider more sustainment--the software is already out in the field, but they are enhancing it. They are updating it and fixing bugs and so forth. Every cycle they do, they do it by release. They have 1,200 (what they call) "program points," and their users know they have 1,200 points, and so they know how much each of their wish-list things are worth. There is a lot of horse trading going on behind the scenes, so they get their 1,200 points, but they get what they need. But, they know we only can get 1,200 points—no more, no less—and it is constant.

Suzanne: Part of this is setting up reasonable expectations with the end users and the customer community, the sponsor community. If there is a good understanding between the developers and the sponsors of the project as to what actually can be accomplished, then we have got a chance at sustainable development.

I would also argue that the short iterations that are common in Agile settings is another piece. I can work for a couple of days to get something done and work really hard to get something done. Then, I know that is only going to last for a couple of days because I have only got two weeks in a sprint. Then, I am going to start over. I am going to start another cycle on another piece of the product. I can reset my clock both from a performance viewpoint and from the viewpoint of the team. So, we are not in this constant up-against-the-wall, but we are working at the right pace. If you are doing this well and you are working at a pace, then teams can keep going this way.

Mary Ann: Right. And, you want your team to be able to keep going in the long haul. These teams that go a hundred thousand miles an hour all the time because they are always in panic mode, they are less effective. If you are overtired--because you are going to get tired--if you are working 70 hours a week every week, week after week after week, the productivity is going to go down. It goes down either by the fact that people just don't get enough done because they just can't function anymore, or there are so many errors introduced because they are so tired and their attention...

Suzanne: They call that technical debt. You are introducing technical debt.

Mary Ann: Yes, you are introducing technical debt. The errors are there. Then, you eventually are going to have to fix them. One of the things I think will help avoid, too, this big panic at the end is because one of the things you are supposed to do every sprint is to have potentially shippable code. Now, what does that mean? I think we have already talked about this, but to remind folks, that means whatever you produce at the end of the sprint could be given to somebody to actually use. So, that means it has been tested. It has been approved. It is all those things.

Suzanne: Right.

Mary Ann: And integrated. So, as you are building things, each piece gets integrated with the other piece, so you are doing integration as you go along. Not the big bang we have seen so many times at the end, where the pieces don't fit together.

Suzanne: The integration point is where you start seeing the death march come in a lot of places, because everything seems to be going okay.

Mary Ann: *My piece works*.

Suzanne: *My piece works and your piece works*, but when we put them together...

Mary Ann: They don't work.

Suzanne: The continuous integration practice that is very common in Agile helps you find those points, that are difficulties between different components, much earlier. So, there are a lot of things that are sort of in the background of Agile practices that you don't think about, like continuous integration, being a contribution to sustainable development. You don't think about understanding team velocity being one of these contributors. But, there are a lot of the practices that make this feasible, although I have also seen programs that abuse this. Where they keep wanting more and more and more story points. You start out with sustainable development, and then you end up with a situation that is just as bad as in a big-bang approach because the number of points that you are trying to produce is really more than what the team can [produce].

Mary Ann: But, if they get to that point, then they are violating the basic tenets of the Agile concepts. They do not have a strong...

Suzanne: This one in particular.

Mary Ann: Yes, this one in particular. They don't have a strong management cadre of support that says, *Time out. You can't do that. You are ruining what we have. This is going to break it. You are falling back into your old habits. You have not transitioned to the new way of thinking.* That is when they need a strong advocate.

Suzanne: You need a Scrum Master who understands how to deal with the management. You need product owners that understand.

Mary Ann: You need a senior manager who is going to run cover for you and who is going to keep all of these nice errors away.

Suzanne: And, we have seen some of that. We have seen both sides of that. We have seen teams that have devolved into just racing as fast as they can because they don't have the management top cover. We have also seen teams where the management understands that keeping that

sustainable pace is part of what keeps the software coming. It is part of what makes their users happy. So, they work really hard to protect the teams in maintaining that pace.

Mary Ann: For those who are going, *I don't get what they are talking about*— think about the old fable, *The Tortoise and the Hare*. Who finished? The tortoise went along at a very sustainable pace, kept trudging along. He finished before the hare, who was racing like mad. It is the same concept, but in software.

Suzanne: I think a big part of this is when we pull things into these smaller chunks and do this continuous integration all along, the developers can see what is happening. The management can see what is happening. A lot of what creates chaos in a lot of these big-bang approach programs is that you don't see anything real until you get to integration. You see documents. You see some simulations, but you don't see it really coming together until integration. Then you are up against the wall. That is way late in the program. If I can see things all along, I am much more likely to allow some leeway. And, I am much more likely as a manager to allow that pace to continue because as long as I keep seeing stuff, I know that we are making progress. I know that I am not going to have that integration wall that I am accustomed to seeing.

Mary Ann: Not only are you seeing stuff, it is stuff that you can actually go and poke at and use.

Suzanne: Sure. Use the sandbox or whatever your environment is.

Mary Ann: There is going to be a sandbox set up and set aside so people can actually go play with it [software] and see if it really does what they think it is supposed to do. Then they can get corrections early on, so even if you got through the whole big-bang integration, and you had something that went out in the field, if it is not what the people needed, then you still failed. As you go along at a continuous pace, at a very nice general pace, and you have something, checkpoints if you will, where I can try that piece.

Suzanne: Right, I've got demonstrations for user integrations.

Mary Ann: Demonstrations. I can play with it.

Suzanne: You have got them to the release end.

Mary Ann: I can find out it works, or it does not work, or if we've taken a slight left turn, and we don't want to go that way. Pull it back in, and if you do that early on, it takes less to fix it.

Suzanne: We have also seen some other things that help with a sustainable pace. We know of one program that has a semi-, or every two years, bi-annual user forum. Everybody in the program stops work. Whatever they are doing, everybody supports that user forum. Now, that does two things. One, it gives people a break away from their normal work. But, the other thing

it does is it really energizes the whole team because they get to meet their users. They get to see the people that are actually using the software. They get to talk to them. They get to understand what their needs are. So, there are different ways that you can introduce mechanisms for helping with that sustainable pace as well as other aspects of the development.

Mary Ann: You just made me think of another practice I have seen on some of the methodologies out there. They will have what they call a "hardening sprint." Now, in many cases that is to make sure all the pieces work together and, maybe, take care of some of the certification accreditation, if you have to do that kind of stuff, or whatever other regulations you have to fill in the paperwork for. But, they also use it to give their team a time to sit back, reflect, and actually do a *what if* kind of thing. As long as they are working on an issue that is going to help the program in the long run, they are allowed to go and use...

Suzanne: ...and do some innovation research.

Mary Ann: They can do some innovation. They call it a "hackathon" in some places, where they go out and they get to do whatever they want to do. But the caveat is, it has to be done within the two-week sprint (if it is a two-week sprint). It has to be something that can help the program one way or another. And, it has to be demonstrated at the end. This gives people renewed energy.

Suzanne: It is a different kind of break.

Mary Ann: Yes, it is a different kind of break, and it gives them renewed energy so they can continue the continuous sustainable pace, and in many cases, it helps the program in the long run solve technical issues that they wouldn't otherwise address.

Mary Ann: Yes.

Suzanne: Mary Ann, thanks for joining us today.

Mary Ann: You are welcome.

Suzanne: In the next episode in the series, we will explore the ninth Agile principle: *Continuous attention to technical excellence and good design enhances Agile*. Listings for papers, blog posts, and podcasts related to the SEI research on Agile adoption in DoD can be found at sei.cmu.edu/acquisition/research.

If you would like more information about the SEI's recent publications in all areas of our work, you can download all of our technical reports and notes at resources.sei.cmu.edu.

This podcast is available on the SEI website at sei.cmu.edu/podcasts and on Carnegie Mellon University's iTunes U site.

As always, if you have any questions, please don't hesitate to e-mail us at info@sei.cmu.edu. Thank you for listening.

Editor's Note: This transcript has been lightly edited to update links and increase readability.