Why Agile Teams Put So Much Emphasis on Being Done Each Iteration

by Mike Cohn • 15 Comments

Why Agile Teams Put So Much Emphasis on Being Done Each Iteration

Agile processes put a lot of emphasis on being done with things. This isn't surprising since the Agile Manifesto says

- Deliver working software frequently
- Working software is the primary measure of progress

Further, Scrum teams try to achieve a potentially releasable state each sprint. And in Scrum, being done with work is so important that teams are encouraged to establish a *definition of done* to promote clarity around what it means to be done.

Many agile teams devote time each iteration to splitting user stories to be small enough so that being done with each is possible within a single iteration.

But why do agile teams put such an emphasis on being done?

It's Hard to Gauge Progress

Agile teams emphasize being done because software development teams have a notoriously hard time gauging progress.

For example, right now, I estimate I'm 2% done writing this blog post. But maybe I'm 3%. Or even 4%. Who knows?

The 90% Syndrome

And gauging progress on software development projects is even more difficult. This has led to software projects suffering from "The 90% Syndrome," which says that software projects are

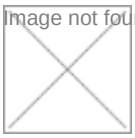
90% done for 90% of their schedules.

You ask a developer how done something is and hear, “90%.” A week later you ask again, expecting to hear that the work is complete. But the developer again says 90%.

This happens because the developer has realized the scope is larger, not always because stakeholders added work, but because the developer failed to anticipate all that would be needed.

Early in the work, the developer thought the job was a certain size. After getting into it a bit, the developer realized it was bigger. And then bigger again. And so on.

A programmer suffering from the 90% syndrome is actually making progress, but is progressing at exactly the same rate that his or her understanding of the problem’s scope is expanding.



This happens because they often fail to see the magnitude of an effort until they are well into that effort.

An Example: Developing Word for Windows

As an example, consider Microsoft’s development of Word for Windows. This project began in September 1984 and was estimated to take 12 months. Nine months later, the team estimated it would take 13 months. A year later, the team estimated 11 months.

For nearly three years, Word for Windows was consistently estimated to be about a year away. The product ultimately shipped after five years and three months.

What Agile Teams Do Instead

Good agile teams have learned that estimating a percentage complete is difficult. And so they simplify the question. Instead of asking, “What percentage done are we with this feature?” they ask merely, “Are we done with this feature?”

All work is put into one of two states: done or not started.

For work to be considered done, it must fulfill the team’s definition of done. Any work that is not done might as well not have been started as far as tracking progress is concerned.

Three Benefits

The simplification of trying to keep work in only two states (done and not started) helps agile teams in three ways:

1. Team members save time not having to answer questions about partially complete work
2. It provides a pessimistic estimate of progress. Since no credit is taken for partially finished work, velocity will be *slightly* understated. This counters the tendency many teams have to overvalue the progress they’ve made as was seen in the Word for Windows example.
3. It encourages teams to work with small, readily completable, product backlog items. Since small items are more likely to be finished within the iteration, and because they represent less uncounted work when they aren’t finished, teams intuitively prefer smaller items.

In accordance with the Agile Manifesto, agile teams emphasize delivering working software frequently and using working software as the primary measure of progress. Focusing on quickly moving things from not being started to done aids them in doing this.

How Do You Track Progress?

How does your team track progress? Does your team estimate percentage complete? Or do you try to move things as quickly as possibly from not started to done? Please share your thoughts and experience in the comments below.

Posted: January 29, 2019

Tagged: sprinting, definition of done, tracking progress