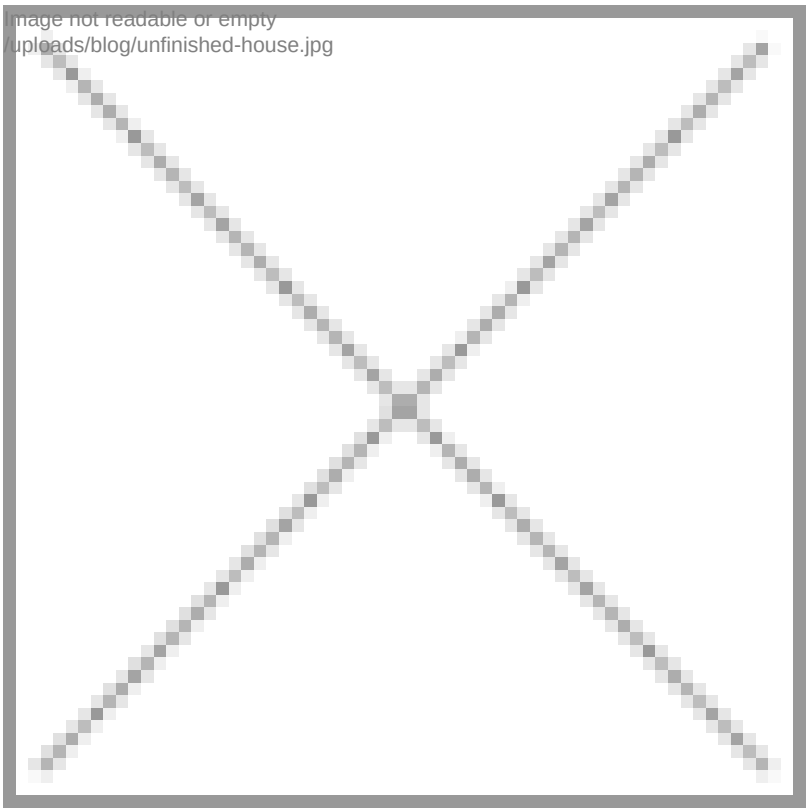
Unfinished Work at the End of a Sprint is Not Evil

by Mike Cohn •

56 Comments

I guess I angered a few Scrum purists recently when I said that it's OK for an experienced team to occasionally show unfinished work in a sprint review. I'm probably going to anger them further this week, as I want to again raise the issue of unfinished work.

I occasionally hear a bold statement like, "Unfinished work at the end of a sprint is evil and should never be allowed."



While I understand the dangers of unfinished work, and while I would never contradict the Agile Manifesto's valuing of working software, this statement is wrong.

It is not unfinished work that is evil, but rather, the accumulation of unfinished work.

By unfinished work, I am referring to product backlog items that are started in a sprint but not finished.

To prove my claim, consider a programmer on a team who finishes his work with a few hours left on the last day of the sprint. As a good team member, he asks his teammates if any could use any help in finishing what they are working on. All are in good shape and will finish without his help.

Perhaps he next considers the short list of desired refactorings the team maintains – but finds it

empty. Perhaps even the defect database is empty. (These last two assumptions may be too extreme, of course, but they'll serve to make a point.)

The programmer then looks at the product backlog and picks a user story to begin. He knows he won't be able to finish it during the sprint, but he's got a few hours and this product backlog item is the most valuable thing to be worked on next.

Preferring to always pair program, this developer spends the remaining hours of the sprint in deep thought about the feature – perhaps sketching some ideas on a nearby whiteboard.

And with that, the sprint ends. And unfinished work is left at the end of the sprint. Where, you may ask, was the unfinished work? In the programmer's head and on the whiteboard.

Thinking is work just as much as typing is. I can't imagine any plausible argument against this programmer's behavior, and so it proves that unfinished work is OK.

What remains though is the need to answer the question: How much unfinished work should be allowed? And that's simple: As little as possible.

So, while I disagree with the oft-made claim that unfinished work is evil, I want to stress that the accumulation of unfinished work is, indeed, evil. When the situation above is repeated, a team could soon find itself where each team member is perhaps two hours into separate product backlog items.

That team would obviously be better off being done with one 1o-hour product backlog item than two hours into five of them.

If your team is leaving unfinished work at the end of sprints, consider whether it's the natural result of sprints punctuating the flow of work, or whether team members should be working more collaboratively to finish one product backlog item before moving onto the next.

Graphing the number of hours of unfinished work in each sprint can help reveal trends. And, the sprint retrospective is a wonderful time to discuss whether unfinished work is building up and what to do about it.

Posted: February 12, 2014

Tagged: sprinting, unfinished work, sprint review

About the Author

Mike Cohn specializes in helping companies adopt and improve their use of agile processes and techniques to build extremely high-performance teams. He is the author of User Stories Applied for Agile Software Development, Agile Estimating and Planning, and Succeeding with Agile as well as the Better User Stories video course. Mike is a founding member of the Agile Alliance and Scrum Alliance and can be reached at hello@mountaingoatsoftware.com. If you want to succeed with agile, you can also have Mike email you a short tip each week.