Why you should consider stopping sprint reviews

by Mike Cohn • 26 Comments

Why you should consider stopping sprint reviews

For many teams, the sprint review has run its course and it's time to stop doing them.

Blasphemy? Maybe, but hear me out.

The Purpose of the Sprint Review

The purpose of the sprint review is for the team to get feedback on what they've developed. That feedback can lead to new product backlog items that represent either newly thought of features or adjustments to what was recently developed. The product owner considers this feedback in finalizing priorities for the coming sprint.

Because its purpose is collecting feedback on what was developed, the review is traditionally done at the end of the sprint. For most teams, it is done immediately before the retrospective.

But, holding a sprint review at the end of the sprint is too late for many teams. Any feedback received during the review arrives too late for the team to use it in the current sprint.

Teams have known this for years. In nearly every Certified Scrum Master course I teach I'm asked questions like:

- Can the team take credit for a product backlog item being done if there's feedback that necessitates a change?
- What if it's a small change?
- What if the change is really an enhancement?

For years, the standard answer to this has been that teams should show their work to the product owner earlier and on an ongoing basis. There's never been a rule that feedback could occur *only* at the end of the sprint in a formal review.

But times have changed. A decade ago, great teams were continuously integrating their code--every time code was checked in, a build was triggered and automated tests confirmed that everything still worked.

Now, great teams continuously *deliver* or continuously *deploy* code. They don't just run automated tests after a check-in, many great teams actually deploy after each check-in.

This completely changes the nature of the sprint review.

Consider the situation of a team that delivers new functionality to production an average of seven times per day. For a great web development team, this isn't unusual at all these days.

What is the discussion going to be like in their sprint review at the end of a ten-day sprint? In that period the team delivered 70 new things (seven per day for ten days).

I imagine the team asking stakeholders in a review, "So, what do you think of 70 new things we developed?"

And the stakeholders reply, "Who cares what we think? What do our users think? They've been using these features already."

In the world of continuous delivery / continuous deployment, seeking feedback from stakeholders in an end-of-sprint review is too late--users are already using the functionality and the best feedback will come from them.

What Should Replace the Review for These Teams?

If a single, formal, end-of-sprint review isn't appropriate for teams that release frequently, what should they do instead?

They should get feedback often and in small doses. As soon a product backlog item is done, team members should show it to perhaps one to three stakeholders who need to see it before it's released. Often this could be only the requestor of the new functionality and the product owner, if they're not the same person.

One or two of the team members who worked on the product backlog item arrange a quick demo to the requestor. They show the item, and if it meets the requestor's expectations, it's delivered--right then.

Perhaps two hours later other team members have finished another product backlog item. They meet with the stakeholder or two who requested that item and possibly the product owner, give a quick demo, and deliver that feature.

This pattern can repeat as often as needed. Sometimes, of course, these mini, one-at-time feature reviews can be batched: the team may show 2-3 things to the product owner at once and then deliver all if approved. And in many cases, a level of trust will arise between the development team, product owner, and stakeholders that allows the team itself to make the call to deliver the feature without the mini-review. This can become a necessity if the team is releasing extremely quickly.

The risk of allowing a team to make the decision isn't huge. Many of these changes are isolated and only tens of lines of code. If a team mistakenly releases something the product owner doesn't approve of, the new functionality can usually be changed quickly or backed out.

Is There Still a Role for the Sprint Review When Delivering Continuously?

There's a reason it's called the *sprint review* rather than something like the *sprint demo*. This subtle difference in name reminds teams that a sprint review is more than a demo. The demo is the central activity and focal point of most reviews, but more happens in a good sprint review

than just a demo.

A sprint review is a chance for the team and stakeholders to discuss priorities and plans for the product. Topics such as these are common:

- Having seen what was finished in the current sprint, what do we think should be done in the next sprint?
- Are there items on the product backlog that should be moved higher or lower in priority?
- Are there backlog items that are no longer needed that can be dropped? Is there an opportunity to release the work of the current sprint?

If your team delivers continuously and addresses these sorts of topics in its sprint reviews, you may find holding a traditional, end-of-sprint review worthwhile even if individual product backlog items have been pre-approved in the one-at-a-time manner I've described.

The Educational Aspect of Sprint Reviews

There is also an educational aspect to sprint reviews. If a team delivers continuously and uses one-at-a-time mini reviews, it could be the case that every item is seen by at least one stakeholder before it's delivered. However, with this approach, many stakeholders will be unaware of much of the work of the sprint.

For example, a team might deploy 50 times during a sprint. You were shown five of them before they happened. I was shown another five. But neither of us was shown everything.

This provides the sprint review with an educational aspect of being the only time everyone sees everything. Seeing delivered functionality within the context of a broader set of features can generate new ideas (new product backlog items) but can also change whether stakeholders approve or reject implementations.

What to Do

I no longer consider the sprint review a mandatory element of Scrum. For many teams, it remains appropriate and as useful as ever. A team that deploys quarterly, for example, is likely

to benefit from the traditional, formal, end-of-sprint review as it's been described and run for years.

Other teams, however, particularly those deploying or delivering continuously, find the traditional sprint review too late for garnering useful feedback.

For these teams, the requirement to conduct a sprint review is replaced with a rule of "get feedback."

When and how they get feedback is up to them, but many choose to do it as one-at-a-time, mini-feature reviews. As an item is finished, it is shown to the small set of people who requested it or are affected by it. In some cases, these mini-reviews can happen multiple times a day.

But fully doing away with the sprint review can be too much for some teams. This is because a good review was always more than just a demo of the product. Those teams conduct the mini-reviews I've described but will augment those with an end of sprint review--just like normal--but with greater emphasis during the review on the educational, knowledge-sharing aspect of the review.

What Do You Think?

Do you sometimes feel that reviewing work only at the end of the sprint is too late? How has your team adapted by getting user and stakeholder feedback earlier? If your team delivers or deploys continuously, have you moved to one-at-time reviews? Please share your thoughts in the comments below.

Posted: August 27, 2019

Tagged: meetings, sprint review
