Team, I just wanted to pass along a few reminders as there was still a bit of confusion between the Sprint Review and Sprint Regression. As shown in Schwaber & Sutherland (2020):

Brandon Hobbs

CS 250

12/10/2021

* Sprint Review - “… Team and stakeholders review what was accomplished in the Sprint and what has changed in their environment. Based on this information, attendees collaborate on what to do next”
  + Focus on what work we achieved in the Sprint and what we want to do in the next Sprint
* Sprint Retrospective - “The purpose of the Sprint Retrospective is to plan ways to increase quality and effectiveness.”
  + Focus on how the team performed and what we did well and poorly as a team – so we can improve within the next Sprint
  + Do not focus on individual work items or plan the next Sprint’s work

**Sprint Review:**

Team, thank you for the recent working session in performing yesterday’s Sprint #9 review. I wanted to pass on my notes generated during that session. Sprint #9 ran from November 15 to November 26, and we closed 15 User Stories for a total of 29 Story Points. Which is an improvement over Sprint #8 where we only closed 24 story points.

We also identified the following bugs:

|  |  |
| --- | --- |
| **Quantity** | **User Impact** |
| 1 | Critical |
| 3 | High |
| 5 | Medium |
| 4 | Light |

The team decided at the end of Sprint #8 to pivot from working on the User Interface (UI) to working on the back end. The thought was while engineering worked on the back end the Product Owner (PO) could review the UI work finished in Sprint #8 with the client and their user cohort. The PO has received quite a bit of feedback, see the User Review Report #1 (link), and the team decided to pivot back to UI work for Sprint #10 as the UI still has the most unknowns and poses the greatest risk – especially for scope creep.

The PO has re-organized the backlog but has not finished his grooming - especially updating the Acceptance Criteria based on the client’s feedback.

Lastly, we further discussed the implications of choosing Jira over Azure DevOps and the choice to use Amazon AWS instead of Google or Azure as we typically do. I have added a few tasks to the backlog, under the Engineering Requirements, to start capturing what we need to do for the backend (in future sprints) based on our understanding of AWS.

**Sprint Retrospective:**

Team, after the Sprint Review we did our monthly Retrospective for Sprints #8 and #9. I think there were a lot of good points brought up that I have captured here.

As the team starts to move away from our old waterfall methodologies towards agile-Scrum the team noted some good practice and some bad practices. The following points were discussed when asked how each discipline was embracing the transition:

* Testing moving “left” to embrace more continuous testing – looking at different tools to integrate with build pipelines and Jira
* The PO has started to review User Stories and groom them as he organizes the backlog. Also, starting to use Story Points to guide his grooming. Allowing dev team room/time to “experiment” and maybe fail before finding the “right way”
* Developers have started to be more transparent and collaborative. Also, starting to embrace core values of Scrum: Commitment, Focus, Openness, Respect, and Courage (Schwaber & Sutherland, 2020).
* Scrum Master: Acting as a good mentor and helping to prevent or solve blockers. Facilitating the scrum events:
* Sprint/Planning: setting up the goal and keeping team honest in the work selected
* Daily Scrum: Keeping members accountable to others; enforcing attendance and sharing
* Sprint Review: Guide discussion and make sure everyone is heard from; Make sure outcome aligns with PO and that PO updates backlog
* Sprint Retrospective: Generate discussion; capture good points; help to remove issues for next sprint

One of the reasons the team thinks the sprint velocity increased sprint on sprint was because the quality of the User Stories has increased. No longer is the team having to run around and ask for clarification; re-work has also slowed. Re-work has slowed because the team has started to embrace progressive elaboration – not believing that the first elaboration of a feature is the final. This is being enhanced by having the freedom to fail gracefully after a brief period of experimentation and refinement, particularly through the idea of spikes.

One point brought up by the Product Owner (PO) was we may want to read up on the technique of *Use of 5 Whys* (5 Whys, 2021). This may shorten the experiment/failure/refine loop.

A few other team members thought we are sometimes still over-designing and should take a look at embracing the idea of agreeing when done. The PO may be a key figure here or we may want to have more frequent end-user testing.

The team also felt that pivoting and adapting to change is getting easier. The team no longer must wait for the “right people” to be in the “right place” since everyone on the team should be aware and capable of performing the task. Before we were siloed and if someone went on vacation or transferred the whole project was sometimes at jeopardy.

The team is also enjoying using the Kanban board as a communicator of “status”. The past silo-ing of the team used to cause poor dissemination of information. Sometimes the testing team would communicate bugs to the developers through email or just in person or development would take notes about a User Story change after a discussion with the PO. This created division in the team – those who know about a bug or change and those who didn’t.

Using Jira to assign tasks whenever the team pivots or there are changes to User Stories will also help with transparency. The agile core value of openness and awareness allows the teams to pivot together as everyone moves together.

The Lead Tester brought up the change of the Top 5 from a list to a slide show an example where we handled change well. Others thought we could improve a bit more. I am looking at setting up a special area in the break room to act as an information radiator. Using Jira, as mentioned before, could also help here. There were also requests that that the PO needs to be more up to date in his grooming of the backlog as a few bugs were introduced because the User Stories had not been fully updated before the start of a Sprint.

The team also spent a bunch of time discussing new organizational tools and how the Scrum-agile principals are aiding development. The team, and recent feedback from the client, pointed out that the use of release and sprint planning phases has dramatically reduced our time to market and the “correctness” of the solutions presented. Most of the team still believes that ignoring iteration reviews is the correct choice currently. The idea to skip this phase was originally suggested based on 3 factors:

* User base is too small to draw conclusion over such a small time period, i.e., power-users were dominating the surveys and building their preferred software at the expense of the casual user
* Release pipelines are not set-up for A/B testing, i.e., splitting the user-base into A-users and B-users to test User Experience hypotheses is not possible
* Client has time constraints making it difficult to meet at that frequency and they prefer not meeting every 2-weeks

The lead tester also mentioned that he is evaluating tools that connect back to Jira to support the idea of continuous delivery and testing.

The last topic that was brought up was if changing to Scrum-agile was the proper fit for the team and the work we are doing. In the past, when we did a lot of financial tools, it wasn’t a fit but now that we are seeing more projects like the current project the timing seemed more appropriate. The team saw these good features of agile-Scrum:

* We can pivot easier
* Potential to garner feedback earlier since the software is working at the end of Sprint
* Earlier/often feedback may lead to pivots but software is more “fit for purpose and useful”

There were also a few negatives brought up:

* The team turnover has hurt adoption as we are just starting the agile-Scrum transition
* Need intimate connection to stakeholders/visionaries or else direction may be lost
  + Clients aren’t always available when they need to be
  + With that lack of “deep” understanding waterfall provides, team is sometimes lost on the next step
* The looser schedule – especially pivoting so often – can feel uncomfortable (change of culture)

However, in the end the team felt that Scrum-agile was a good fit. Our small team allows us to be nimble. The guiding documents suggest this about team size, “…small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people (Schwaber & Sutherland, 2020)”. The small team - even though there are some improvements (information radiator) - keeping everyone in the loop of changes was well handled. Our team size allows team to act as back up to each other as we all become responsible with no sub teams.

The last point the team felt that was allowing Scrum-agile to be successful is that upper management has also aligned. A mentioned in Chapter 19 of Cobb (2015) “taking the agile transformation seriously and supporting it at all levels of the business [is] essential to make the culture shift to the agile mind-set…”.

**References**:

1. Charles G. Cobb. (2015). *The Project Manager’s Guide to Mastering Agile: Principles and Practices for an Adaptive Approach*. Wiley.
2. Gilley, C. (n.d.). *The Pros and Cons of Agile Product Development*. User Voice. Retrieved Nov. 28, 2021, from https://www.uservoice.com/blog/the-pros- and cons-of-agile-product-development
3. Leather. E. (April 7, 2017). *Amazonian Agility*. Medium. https://medium.com/frontira collection/amazonian-agility- e3720ff004f7
4. Mind Tools Content Team. (n.d.). *5 Whys*. Retrieved Dec. 1, 2021, from https://www.mindtools.com/pages/article/newTMC\_5W.htm
5. Schwaber, K., & Sutherland, J., (2006). *The Scrum Guide*. [White Paper]. Scrum Guides. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum Guide US.pdf#zoom=100