My Experiences with Agile Development

By Julie Odenbach

I have been writing software for a long time, but only started working in an agile environment over the last few years. In hopes of identifying what I found helpful with using an agile approach, I will discuss how working with an agile framework compares with working with a more traditional software development software framework.

Four Core Values of Agile Development

1. People and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Collaborating with customers over contract negotiation.
4. Responding to change over following the plan.

The core values of agile development focus on creating high-value working software that meets the customer's needs. More attention is paid to face-to-face communication with the customer and with each other. Producing demonstrably working software is valued more highly than producing reams of documentation. Since change is inevitable, the agile process allows the developers to respond to change quickly. User stories that are no longer relevant can removed from the product backlog, and new user stories added to address needed changes starting with the next sprint.

Twelve Agile Manifesto Principles

1. Customer satisfaction through working software delivery at regular intervals.
2. Accommodating change.
3. Deliver working software frequently.
4. Collaboration between key stakeholders and the development team throughout the project.
5. A motivated development team will deliver the best results.
6. Prioritize face-to-face communication.
7. The primary measure of progress is working software.
8. The development team should be working at a sustainable pace.
9. Attention to technical detail and design is essential.
10. Simplicity means doing only what needs to be done right now.
11. Self-organizing development team allows developers to have ownership of their work and promotes open communication with other team members.
12. Take the time to review and revise how the team can make changes throughout the development process.

The twelve agile manifesto principles bring up several points to discuss. Our primary agile goal is customer satisfaction by providing customers with working software at regular intervals. The customer will be able to verify the working software when demos are held at the spring review meeting with the stakeholders.

Face-to-face communications between the customer, product owner, and the development team does make a difference. Doing a demo for the customer allowed problems to be caught early on so that they could be resolved quickly. Of course, face-to-face may mean face-to-face over zoom or Teams these days, but that works too.

It’s so important that the development team is working at a **sustainable pace**. Yes, management will push for the development team to take on more and more points at each sprint. And the development team should be able to increase their productivity as they become more familiar with the problem domain, but pushing too hard to increase productivity beyond what is sustainable will only backfire.

I have worked on teams where management decreed how many points everyone needed to complete each sprint, and the demands were only sometimes reasonable. Agile principles mean that:

- Self-organizing development teams where the developers have ownership over their work.
- Self-motivated teams will produce the best results.
- The development team must agree on what work they can commit to completing in the upcoming sprint.
- Development teams must work at a sustainable pace.

Working with agile thus requires management buy-in. Trusting the self-organized development team is often difficult for managers who honestly believe that nothing would ever get done without their micromanaging every minute detail. But keep in mind that if management requires unreasonable workloads, the quality of the software produced will suffer, and developers will eventually get frustrated and leave for other jobs.

It's often up to the scrum master to educate not only the development team about agile principles and processes but also to educate company management. Some companies bring in agile coaches on a contract basis to initially train management and developers, in agile principles and processes.

Five key Agile Scrum Meetings

1. Product backlog refinement
2. Sprint planning
3. Daily standup
4. Sprint review
5. Retrospective

The regularly scheduled agile scrum meetings were very helpful in maintaining open lines of communication with members of the development team. Before working within the defined structure of agile scrum, contact with other developers, managers, and key stakeholders was often somewhat sporadic. There needed to be more regular communication to prevent problems from arising.

The **daily standup** is a great way to get in touch with those you are working with and make sure that we are all on the same page. The daily standup is time-boxed to only 15 minutes. It’s a quick standup meeting where everyone shares what they accomplished the day before, and what they plan to accomplish today, along with a short rundown of any problems or roadblocks encountered. If one or more individual identifies an issue that will take too long for the standup meeting, the team can set up a later discussion to address that issue later.

If you have one or more “chatty” people on your development team, you can pass around a ball or baton, and only the person holding the ball or baton can speak. That helps to ensure that everyone gets their chance to update the team on their status. The scrum master should ensure that the daily standup moves along quickly in an orderly fashion, and that more time-consuming problems are handled with a subsequent meeting for only the individuals involved, rather than taking up time of the entire team.

I did work on one project where the scrum master held two one-hour standup meetings every morning (one for the internal team, and the other with the client). That is not in keeping with agile scrum rules. It’s also a terrible waste of time for everyone on the development team. That should not be happening.

The product owner in conjunction with the development team holds the **product backlog refinement** meeting. The purpose of this meeting is to add, delete, and modify the user stories in the product backlog. Team members can suggest new stories to add to the product backlog, and the product owner will decide if that would be appropriate. The product owner may also check with the team if they agree before removing a user story from the backlog. The developers can offer new updated details to flesh out the user stories remaining in the product backlog.

Sprint planning is done just before the start of a new sprint. The development team, which includes the developers, scrum master, and product owner, will assign story points to each story from the product owner’s prioritized product backlog being considered for the upcoming sprint. Pointing the user stories may take place prior to the sprint planning meeting, or take place at the start of the sprint planning meeting. Taking the assigned story points in consideration, the development team determines what stories they will commit to complete in the upcoming sprint. Everyone should have a high degree of confidence in being able to complete the work, as well as assuring that all team members will have enough work to keep busy.

At the end of each sprint, the team will hold a **sprint review** meeting where working software produced during that sprint is demonstrated for the team and key stakeholders. The demo allows the customer to see how the software performs and offer their input.

Finally, the last meeting at the end of a sprint is the **retrospective** meeting. This meeting is where the development team, again including the scrum master and the product owner, can review what went well, what went poorly, and how the team can improve their development process. It’s also an excellent time for all team members to assess how their work is going, and how to avoid making the same mistakes in the future. It might be best not to have management sitting in on the retrospective so that people can speak honestly without fear of reprisal. The scrum master can address issues with management if necessary.

The Downside of Using Agile Scrum

Complex software projects are always challenging, and using agile scrum will not guarantee success for your project. It can take time for team members to become accustomed to working in an agile scrum framework. If the development team is not composed of self-motivated and technically competent individuals, you will have problems. I do not see agile as making software development any quicker, but I do see agile as accommodating change much easier to handle. And there are always many changes needed as developers gain more insight into the required technical challenges.

Where Agile Scrum Would Not be Useful

An agile scum framework may not benefit to your organization if your software development team is too small or if your software project is small, simple, and straightforward. The development team usually consists of the product owner, scrum master, and at least two to eight developers. Software development teams smaller than a product owner, the scrum master, and at least two developers are unlikely to gain much by following all agile scrum principles. But staying focused on producing working software frequently and maintaining good communications with customers is still recommended.

If the size and complexity of your software development project is too much to be handled by a single scrum team, then you can establish multiple scrum teams. A Scrum of Scrums meeting is used to coordinate numerous scrum teams working in parallel. On an even larger scale, you can implement agile at an enterprise level using Scaled Agile Framework or SAFe©.

Summary

In 2023, at least 71% of U.S. companies are now using agile[1] for their software development. Many companies say that the agile process helps them to better manage priorities, increase communication, and enhance their software quality.

In summary, using an agile framework for software development provides a proven methodology for creating working software. That working software will keep your clients happy. It's worth the time to do your research on implementing an agile process for your team, division, or company.

[1] Zippia. "16 Amazing Agile Statistics [2023]: What Companies Use Agile Methodology" Zippia.com. Nov. 27, 2022, https://www.zippia.com/advice/agile-statistics/