Analysis of Scrum Vs. Popular Development Processes

for
Jeff Sutherland
Ken Schwaber
Founders and Creators
of Scrum

by
Christian Doughty
Software Engineering Student
Milwaukee School of Engineering

February 11, 2020

324 E Juneau Ave
Milwaukee, WI 53202

February 11, 2020

Mark Zimmerman
GS1002 Professor
Milwaukee School of Engineering
Milwaukee, WI, 53202

Dear Professor Zimmerman:

This is my formal report, Analysis of Scrum vs. Popular Development Processes. Prior to the writing of this document, I had little to no background in different development processes. Through my research I have learned much about how several processes work, as well as the different benefits of each. I thank you for your teachings, as constructing a report in this style will be important for my field of work in the future. It is a wonderful skill to have.

Although development processes are often quite different from one another, their end goal is the same. The bulk of the process is going from an idea to a product or a release, something I have been working with already and will continue to see in my future work. Having an understanding in several processes will amplify my ability to work using these systems when applying for jobs during and post college.

This research project was very enjoyable, and I thank you again for the information taught during this quarter. If any questions arise feel free to contact me at cdoughty10@yahoo.com.

Sincerely,

Christian Doughty

Table of Contents

List of Tables and Figures

Abstract

Two popular development processes, Scrum and DevOps, are explained in detail, followed by a comparison analysis between the two. The analysis determines that both processes lack a proper long-term development cycle, focusing primarily on short term development and faster releases.

To rectify this lack of longer cycles, a proposal is made to improve Scrum with a new event called a Marathon. Similar in nature to a Sprint, Marathons would be primarily used for

- Larger development tasks
- High risk tasks with potential for issues and bugs
- Many smaller, but still related, development tasks

Marathons would close the gap between long and short term development, allowing for higher quality and integrity while sacrificing the ability for a fast release time.

Introduction

Development processes are an efficient and effective way to turn an idea into a product. Two popular processes are becoming more recognized by companies and integrated into the company workings. However, the practicality of these processes may not always be the most beneficial to the company employing them.

Current Scrum guidelines call for 1-4 week long Sprints, which is a short amount of time to complete larger or more risky development projects. Sprints are designed for quick release times with shorter development periods *(SCRUM Development Process, 1995, p. 13-14)*. Companies employing Scrum and having difficulties implementing the Sprint may have an easier time implementing a new proposed event. A Marathon would allow for more flexibility in scheduling and development completed. Marathons would also be used for larger development tasks likely to not fit within a Sprint.

A few key questions should be asked before implementing the Marathon into Scrum.

- Are companies struggling with shorter development windows?
- Would longer development windows remedy these problems?

These questions will determine the usefulness of a Marathon, and whether or not companies would benefit from Marathons. According to later analysis in this report, these questions are answered with a “yes” for certain circumstances.

This report describes Scrum and DevOps in detail, compares them, and provides an analysis on how Scrum could be improved. The improvement proposal is meant to allow more flexibility in Scrum events and timelines, allowing companies to choose which path is more appropriate towards their product and requirements.

Data Section

The Framework of Scrum

Developed in the 1990s by Ken Schwaber and Jeff Sutherland, Scrum is a branch of Agile development and has evolved into a framework for product creation or software development. It describes the roles of specific team members, events between teams, and artifacts that make the process work. *(The Scrum Guide, 2017, p. 3-4)*

For Scrum to be a successful development process, all members must become proficient in and exercise the values of:

- Courage
- Commitment
- Focus
- Openness
- Respect
- The Scrum Theory

(The Scrum Guide, 2017, p. 3-4)

Scrum Theory

The Scrum Theory is built upon the theory of empirical process control. Three values - transparency, inspection, and adaptation - are the three pillars of any empirical process control adaptation. *(The Scrum Guide, 2017, p. 4-5)*

Transparency - All partners in the project must be up to date and in complete understanding of what is going on. A common definition of "Done" must be shared between all partners.

Inspection - Team members must inspect artifacts and progress towards a Sprint Goal, without the inspection impeding progress towards that goal.

Adaptation - If the results of the inspections are deemed unacceptable, the product must undergo adjustment to account for the lacking quality where applicable. This adjustment must be at the highest priority to avoid further deviation.

Team Member Roles

The Scrum team consists of three major roles, each being self-organizing and cross-functional with one another. Each role decides the most efficient way to complete their work and maximize opportunities for feedback. *(The Scrum Guide, 2017, p. 6-8)*

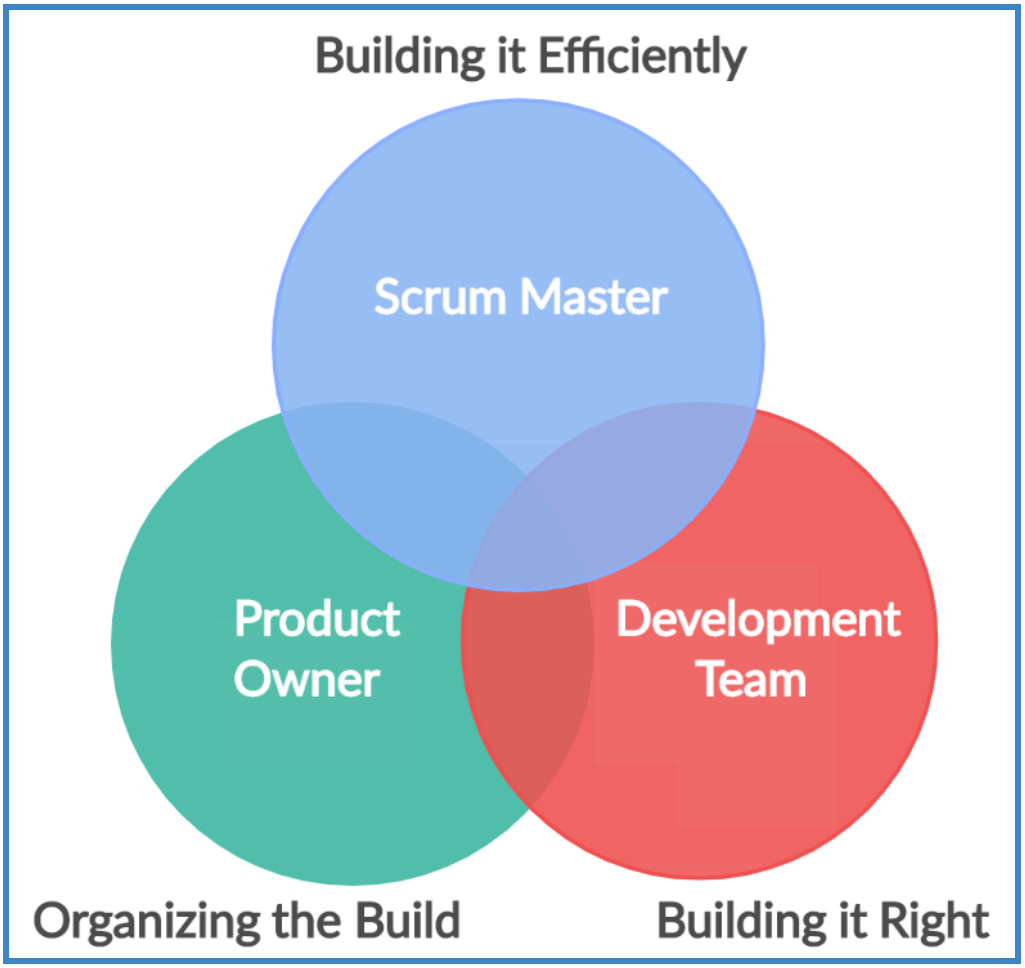

Figure 1 **Scrum Team Roles**
Source: Christian Doughty

The Product Owner is the person in charge of expressing what is on the Product Backlog, and ensuring that the Development Team understands what needs completing. Their purpose is to maximize the value of the work completed by the Development Team.

The Development Team organizes and manages their own work, optimizing their efficiency and effectiveness. Their main goal is to deliver an Increment of "Done" after each Sprint.

The Scrum Master is in charge of keeping all Scrum Team members up to date with the Scrum practices and ensure they are being followed appropriately. All Scrum Events are facilitated by the Scrum Master. This position interacts with each other role directly.

Interactions with the Product Owner are for understanding the product, ensuring goals are set, and assisting with the prioritization of backlog items. This helps make the product as quickly as possible, while maintaining the Product Owner's vision of the product.

Interactions with the Development Team are for coaching in organization, cross-functionality, and general Scrum methodologies. The Development Team may also request Scrum Events when necessary, which the Scrum Master will facilitate.
(The Scrum Guide, 2017, p. 6-8)

Scrum Events

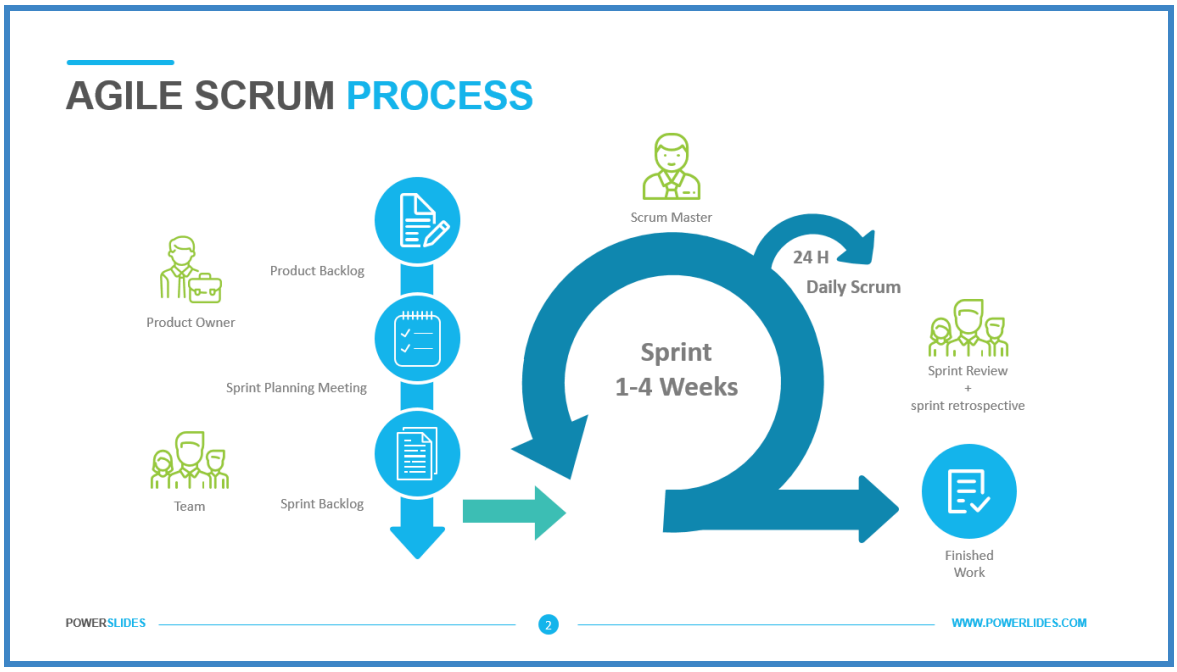

Figure 2 **The Agile Scrum Process**
Source: "Scrum Product Details" PowerSlides.com

Sprints are one month long sessions of dedicated development towards a final usable product. Sprints are created through the analysis of the Product Backlog, and the associated priorities with each item. *(SCRUM Development Process, 1995, p. 13-14)*

Two primary questions are asked during the Sprint Planning process.

What can be completed?

- Only the Development Team can assess and judge what can be completed during a Sprint. The Product Owner discusses what objectives and goals should be completed.

  The set of goals agreed upon are named the Sprint Goal. At the end of the Sprint, all items under the Sprint Goal should be completed.

How will it be completed?

- After the Sprint Goal is clearly defined, the Development Team decides on the best path to take to create a "Done" product Increment. The chosen tasks are logged into the Sprint Backlog.

  By the end of the Sprint Planning, the Development Team should have a plan in place and be able to explain to the Product Owner and Scrum Master how the Sprint Goal will be accomplished.

(SCRUM Development Process, 1995, p. 13-14)

Daily Scrum meetings occur each day during the Sprint within the Development Team. Several aspects of the ongoing sprint are analyzed.

- Completed work towards the Sprint Goal
- Upcoming work towards the Sprint Goal
- Potential roadblocks in upcoming work

Analysis of this information will result in an idea on how to best move forward. Meetings also promote better communication between team members, and improves general knowledge about the product being created. *(The Scrum Guide, 2017, p. 12)*

Post Sprint Reflections occur following the conclusion of a Sprint, where two specific meetings are planned to review and reflect upon the content of the last Sprint. Looking back assists team members in learning from potential mistakes, analysis of what new tasks have been created, and general improvement preparations for the next Sprint. *(The Scrum Guide, 2017, p. 13-14)*

Sprint Review happens directly after the end of a Sprint. The most recent Increment is inspected and the Product Backlog is modified where necessary. Various elements of the recent Sprint are analyzed.

- Attendees determine what Product Backlog items are "Done"
- What went well during the Sprint, and encountered problems
- Collaboration on content of the next Sprint
- General review of finances, timeline, and marketplace of the Product

Sprint Retrospective is a meeting within the Scrum Team to reflect upon recent progress, and improve for upcoming Sprints. The Definition of "Done" may be analyzed and modified during these meetings if appropriate.
(The Scrum Guide, 2017, p. 13-14)

Scrum Artifacts

Artifacts are common terms that all members of the Scrum Team know, allowing all members to have the same definition for a certain word. *(The Scrum Guide, 2017, p. 14-18) (Scrum: A Pattern Language (...), 1998, p. 4-6)*

Product Backlog - An organized list of prioritized tasks required to complete the Product. The Product Backlog is never complete. Changes to business requirements, market conditions, technology, or consumer usages can cause items in the Product Backlog to be created, removed, or revised.

Sprint Backlog - A set of items taken from the Product Backlog as the main focus of the next Sprint. The progress of a Sprint can be gathered by viewing the Sprint Backlog, through analysis of work completed and work to be done. Work is added to the Sprint Backlog when necessary by the Development Team.

Sprint Increment - The sum of all Product Backlog items completed during a Sprint. Each Increment must meet the team's definition of "Done" and be in a usable condition.

"Done" Definition - A shared idea of completeness between all team members. This definition ideally contains total functionality, usability, and satisfaction. Sharing a definition of "Done" results in all members being in agreement. This is defined on a per-team basis based on the views and ideas of the Scrum Team.
(The Scrum Guide, 2017, p. 14-18) (Scrum: A Pattern Language (...), 1998, p. 4-6)

The Framework of DevOps

Initially created in 2008 by Patrick Debois and Andrew Shafer, DevOps kicked off in 2009 as a combination of software development and operations methodology

Several main aspects are required to make DevOps a viable development process:

- Frequent Communication
- Real-Time Collaboration
- Focus on Service and Quality

(The Way of DevOps)

Values and Methodologies

The CAMS model is best used to describe the values of DevOps. CAMS is an abbreviation of the important values of DevOps. *(The Way of DevOps)*

Culture - Even with more innovative tools and technology, the elements of human culture are essential to software development. An environment of open communication should be created, with shared goals and understanding between all team members.

Automation - Acknowledging the idea that something will go wrong and the inherent instability in complex frameworks are the purposes for implementing automated tests. Automation is used to create test cases to stress and simulate partial failures of frameworks in order to improve them prior to release.

Measurement - Collecting data and information is essential to determine whether progress is moving forward. Many questions are asked about the team during this process, including both technical and cultural questions. Valuing measurements and consistently improving from them are essential aspects to DevOps.

Sharing - Teams embrace that they are all involved in producing an application, as well as ensuring it meets customer expectations. Being transparent and interpreting data from other teams is important, as all teams are then aware of all product progress across the company.

(The Way of DevOps)

Toolchain

The DevOps toolchain is a set of categories that determine which software development technologies are necessary to be used at each level of the software development process. *(The Way of DevOps)*
Four core categories define most of the tools in the toolchain. However, other categories can be added where necessary in the development process.

- Code developing
- Automatic testing
- Produce release management
- Performance monitoring

(The Way of DevOps)

Architecture

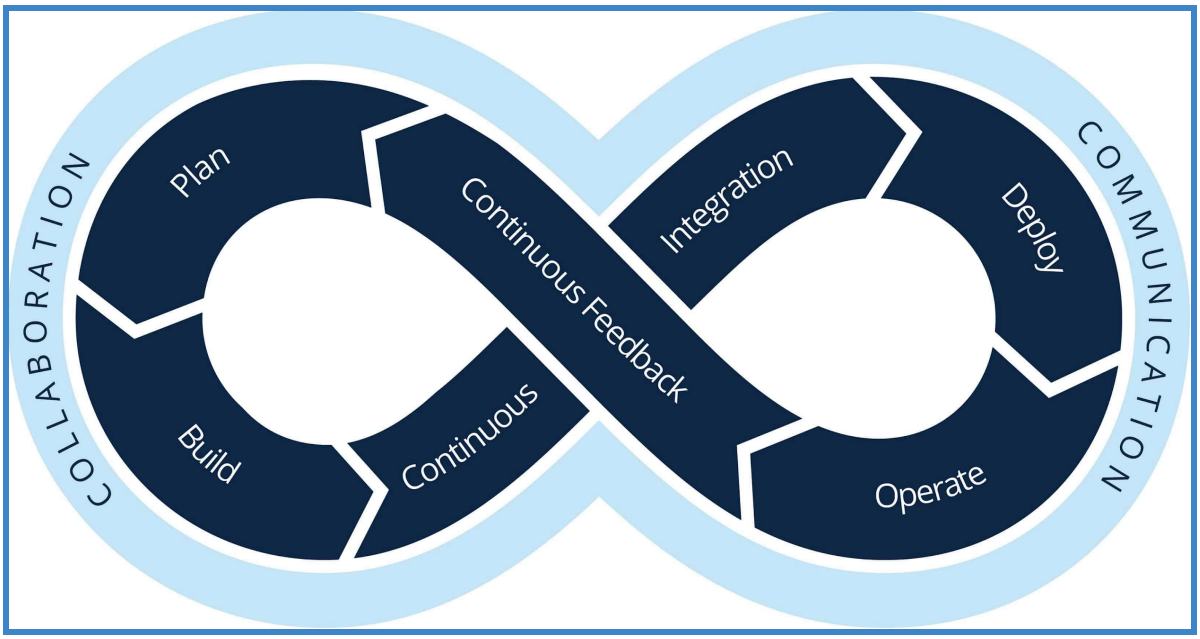

Figure 3 The Agile DevOps Process
Source: "DevOps IT: Toolchain and Architecture" smartsheet.com

The DevOps architecture is based on a continuous loop of consumer feedback and development integration. Several main stages during this process are shown in Figure 3.

Planning is used to analyze the feedback from consumers. The current product is compared to what people are looking for and a plan is put into place to refine all feedback into the final product.

Building is the process where planning is put into reality. The feedback has been changed into something tangible and working.

Continuous Integration is the process of adding a new build into the product. This may entail a rework of existing processes and code, as new systems may affect or be affected by older systems.

Deployment is when the new system is finalized and shipped to consumers. It is considered done and ready for consumer usage.

Operation is done by the consumers. They make an opinion on the product and any new systems that come through during Deployment.

Continuous Feedback completes the infinite loop, where consumers send feedback and opinions to the product owners. The cycle begins again at Planning after this feedback is received.
(DevOps Explained)

DevOps Artifacts
Artifacts are terms that the entire DevOps team should understand to efficiently follow DevOps.

Cycle Time - It is the time required to design, build, deploy, and monitor software delivered to consumers. Observation, experimentation, and analysis of data is essential for determining this.

Continuous Integration - Abbreviated as CI, it is the practice of constant code testing. The main goal of CI is to reduce product problems, improve product quality, reduce time to release, and create feedback loops to reinforce the code integrity.

Continuous Delivery - Abbreviated as CD, it is the practice of frequently building, testing, and releasing code changes to a production or testing environment. CD is achieved when synchronized with the steps for CI.

(DevOps Explained)

Problem Analysis

Through analysis of the development processes covered, it can be determined that both support short term projects more than long term projects. Understandably, shorter development windows allow for:

- General sense of urgency
- Quicker development
- Faster release times

On the other hand, shorter development windows have downfalls as well:

- Less testing time
- More room for error and bugs

Long term development windows (1-3 months) would open up several improvement opportunities, while sacrificing the ability for a quicker release date. A process like this may be used to repair a company reputation after a particularly poor release, with the extra time being used to ensure everything is perfect for a new release. Long term windows allow for:

- More testing to ensure stability
- Larger or more content developed
- Room for consumer feedback during development

Long term downfalls entail:

- Longer waits between releases
- Lesser sense of urgency, potential for slacking off

The benefits of a longer development window outweigh the downfalls, as the downfalls can be rectified through proper encouragement and explanation to the Development Team.

Improvement Proposal

Scrum is the most likely candidate to implement a longer term development cycle, as the Sprint is already clearly defined. Adding a second branch, a Marathon, allows companies to choose whether a Sprint or Marathon is best for that development cycle.

The Scrum Marathon

Continuing the play-on-words tradition of Scrum, with a Sprint being a shorter event, a Marathon would be a longer Scrum Event. Marathons can be used in place of Sprints where necessary.

Marathon Planning

Planning would be done identical to Sprint Planning, however larger tasks would be chosen for the Marathon Backlog.

Larger tasks from the Product Backlog would be prime candidates for Marathons, as the length of the marathon allows for the selected tasks to be completed while allowing room for possible delays. General development length during a Marathon would be 1-3 months.

Weekly Scrum

Similar in fashion to the Daily Scrum during Sprints, a Weekly Scrum would be held once a week during each Marathon. The content would be identical, however Weekly Scrum meetings would last longer than Daily Scrum meetings, due to more content being covered in a Marathon.

Marathon Retrospective and Review

Generally identical to these events as they are for Sprints, these would be used to review the last Marathon and decide what went well and what did not. Modifications for the next Marathon or Sprint are noted down for the future.

Work Cited

Beedle, Mike, et al. “Scrum: A Pattern Language for Hyperproductive Software Development.” *ScrumInc*, ScrumInc, Dec. 1998, www.scruminc.com/wp-content/uploads/2014/05/Scrum-A-Pattern-Language-for-Software-Development.pdf.

“DevOps Explained.” *GitLab*, about.gitlab.com/devops/.

Schwaber, Ken. “SCRUM Development Process.” *JeffSutherland*, JeffSutherland, 1995, www.jeffsutherland.org/oopsla/schwapub.pdf.

Schwaber, Ken, and Jeff Sutherland. “The Scrum Guide™.” *The Scrum Guide*. N.p., Nov. 2017. Web. 20 Jan. 2020.

“The Way of DevOps: A Primer on DevOps Principles and Practices.” *Smartsheet*, www.smartsheet.com/devops.