SCRUM

THE COMPLETE GUIDE TO THE AGILE PROJECT MANAGEMENT FRAMEWORK THAT HELPS THE SOFTWARE DEVELOPMENT LEAN TEAM TO EFFICIENTLY STRUCTURE AND SIMPLIFY THE WORK & SOLVE PROBLEMS IN HALF THE TIME

JOSH WRIGHT

Scrum:

the complete guide to the agile project management framework that helps the software development lean team to efficiently structure and simplify the work & solve problems in half the time

JOSH WRIGHT

Table of Contents

Disclaimer Notice:

Please note the information contained within this document is for educational and entertainment purposes only. All effort has been executed to present accurate, up to date, and reliable, complete information. No warranties of any kind are declared or implied. Readers acknowledge that the author is not engaging in the rendering of legal, financial, medical or professional advice. The content within this book has been derived from various sources. Please consult a licensed professional before attempting any techniques outlined in this book.

By reading this document, the reader agrees that under no circumstances is the author responsible for any losses, direct or indirect, which are incurred as a result of the use of information contained within this document, including, but not limited to, — errors, omissions, or inaccuracies.

Introduction

Scrum is decisively a development of Agile Management. Scrum strategy depends on a lot of characterized practices and jobs that must be included during the product advancement process. It is an adaptable approach that rewards the use of the 12 spry standards in a setting concurred by all the colleagues of the item.

Scrum is executed in brief hinders that are short and intermittent, called Sprints, which generally run from 2 to about a month, which is the term for criticism and reflection. Each Sprint is an element in itself, that is, it gives a total result, a variety of the last item that must have the option to be conveyed to the customer with the least conceivable exertion when mentioned.

The procedure has as a beginning stage, a rundown of targets/necessities that make up the task plan. It is the customer of the task that organizes these goals considering an equalization of the worth and the expense thereof that is the manner by which the emphases and resulting conveyances are resolved.

From one perspective the market requests quality, quick conveyance at lower costs, for which an organization must be extremely deft and adaptable in the advancement of items, to accomplish short improvement cycles that can fulfill the need of clients without undermining the nature of the outcome. It is an extremely simple technique to actualize and well known for the snappy outcomes it gets.

Scrum technique is utilized for the most part for programming advancement, however different divisions are additionally exploiting its advantages by actualizing this procedure in their hierarchical models, for example, deals, showcasing, and HR groups and so on.

Scrum Development

In Scrum, the group centers around building quality programming. The proprietor of a Scrum venture center around characterizing what are the attributes that the item should need to manufacture (what to fabricate, what not, and in what request) and to defeat any deterrent that could ruin the undertaking of the improvement group.

The Scrum group comprises of the accompanying jobs:

Scrum ace: The individual who drives the group managing them to agree to the standards and procedures of the system. Scrum ace deals with the decrease of

obstructions of the undertaking and works with the Product Owner to expand the ROI. The Scrum Master is responsible for staying up with the latest, giving instructing, tutoring and preparing to the groups on the off chance that it needs it.

Item proprietor (PO): Is the delegate of the partners and clients who utilize the product. They center around the business part and is liable for the ROI of the task. They Translate the vision of the venture to the group, approve the advantages in stories to be fused into the Product Backlog and organize them all the time.

Group: A gathering of experts with the fundamental specialized information who build up the task together doing the narratives they focus on toward the beginning of each dash.

Applications of Scrum

Scrum has been used worldwide extensively and applied across various use cases including but not limited to: research and identify markets, technologies, and product capabilities; develop and release products and enhancements as frequently as many times per day; maintain and sustain products, systems, and other operational environments. Further, Scrum has been used to develop software (embedded and otherwise), hardware, networks of interacting functions, autonomous vehicles, schools, government, non-profit organizations, marketing, operations, and almost everything we use in our daily lives.

Fundamental Scrum Trade-offs

There are four fundamental trade-offs defined by the Agile Manifesto that Scrum Framework implements:

1. Individuals and interactions OVER processes and tools
2. Working software OVER comprehensive documentation
3. Customer collaboration OVER contract negotiation
4. Responding to change OVER following a plan

Furthermore, there are three main focus areas that the Scrum Framework implements:

1. <u>Focus on value:</u> Everything that is done with the *Agile Mindset* focuses on the value it creates. If there is value, then do it. If there is no value, then don’t do it.
2. <u>Focus on collaboration:</u> Scrum Framework focuses on teaming the people with the right skills and the right mindset for creative

collaboration by providing the right cultural environment to enable and amplify strong collaboration.

3. Focus on adaptability: Scrum Framework deals with the fact that requirements do change quickly and frequently. Therefore, teams re/de-prioritize existing work when it is realized that it is not valuable. For that reason, the *Agile Mindset* gives special emphasis on adaptability.

The Background of Scrum

You can apply these principles to your life to create an immediate sense of accomplishment. Although you may accomplish some of the goals you set, these goals may not be aligned with your values. Plus, if you don't recognize accomplishments regularly, you'll likely lose enthusiasm or get distracted easily.

Using this method, your definition of success will shift from “what can I accomplish this year?” to “what can I do to move closer to my goals today?” Every day becomes a successful day, and at the end of four weeks, just 28 days, you have something tangible to call success. It’s inspiring and self-reinforcing. You become unstoppable.

You also become more flexible. If life throws you a curveball, it’s easy to adjust because you are nimble and focused on your values rather than some obscure, immovable object in the future.

Chapter 1 What is Scrum, A General Overview and A Little Bit of History about it

What is Scrum

Scrum is a deft improvement technique utilized in the advancement of software dependent on an iterative and gradual procedures. Scrum is versatile, quick, adaptable and powerful spry structure that is intended to convey an incentive to the client all through the improvement of the venture. The essential goal of Scrum is to fulfill the client's need through a situation of straightforwardness in correspondence, aggregate duty and ceaseless advancement. The improvement begins from a general thought of what should be fabricated, explaining a rundown of qualities requested by need (item build-up) that the proprietor of the item needs to get.

In 1993, Jeff Sutherland and his group at Easel Corporation made the Scrum procedure to be utilized in programming advancement forms by consolidating the ideas of the 1986 article with the ideas of item arranged improvement, exact procedure control, iterative advancement and gradual, programming procedures and profitability improvement, just as the advancement of mind-boggling and dynamic frameworks.

"Doing half of something is, basically, sitting idle." – Jeff SutherlandCLICK TO TWEET

Scrum Methodology and Process

What Does Scrum Mean and How Does It Fit In?

Now that we have spent some time in this guidebook talking about the Lean methodology and the Agile Framework, it is time to take a look at Scrum and how it can fit into all of this. There are going to be some different approaches to using the Agile Framework, different methods, and types that the user can work with based on what works the best for them. Each approach or each type of agile framework is going to be considered lightweight. What this means is that all the practices and the rules for each approach are going to be kept down to a bare minimum.

In addition, each approach will also need to ensure that the main focus here is on empowering your developers to make good decisions together and to collaborate with each other. You will notice that each developer who works on this framework is going to come into the project with a different background

which can be great news for allowing the group to work more effectively and quickly. You never want to end up with a group of people who all know how to do exactly the same thing.

Having a group of people with different backgrounds and knowledge can help you create a team that is more cohesive and more ready to get things done.

As we go through here, it is important to remember that the big idea of working with the Agile system is to create all your products, applications, and more in small increments. Each of these individual increments is tested before you can consider them complete. This assures that the product is built with quality at that time, instead of trying to waste time finding quality later on down the road.

Remember that the Agile Framework is going to be a process that your developmental team is able to follow to ensure that only positive things occur. It has been designed to make it so all parties, both your business and the customer, can provide feedback while the project is being developed, rather than after the product is done and has gone to the market. Doing this can cut out a lot of problems that may creep up and actually makes your business and your process more efficient in the long run.

And while there are quite a few different choices that you can make when it comes to software development approaches in this framework, the most popular ones that many people choose is Scrum. Scrum is a framework that has a broad application that allows users to manage and control incremental and iterative projects of all different types.

Scrum can be really useful if there are many different types of projects that you need to complete at the same time and it will ensure that your team is able to complete each of these projects in a timely manner. At the same time, Scrum will work to make sure that the value of your product doesn’t end up changing at the same time. The philosophy behind Scrum depends on connections and collaboration which can ensure that your team completes each of these projects in the best way possible.

To make sure that all of this can be done, there are several different roles that come with the Scrum process and each person on the team needs to be willing to take their role seriously. If even one person on the team doesn’t do the job the right way, then the whole process is going to fail. Remember that Scrum is all about staying connected and about teamwork.

To break it down - even more, there are three basic principles that come with the Scrum ideology including:

- Adaptation
- Inspection
- Transparency

The Scrum ideology is all about seeing how things are and making sure that everyone on the team knows what is going on as well. It's important that you be concise and clear as much as possible and that everyone is always informed. It may be tempting at times to hide mistakes, but this just leads to a ton of issues down the road when other team members don't know what is going on. Transparency is so important during every step of this process to ensure that everyone is held accountable.

Scrum is also going to require inspections to help everyone and everything stays on track. Without this, it is hard for the team to know if they are even working on a project that will hold value for the customer and they may get to the end and find that the product doesn't even work.

And of course, we can't forget all about adaptation. Things are going to change —that is just a part of life. It's important for your Scrum team to be able to adapt to all the different changes that this process is going to go through. The customer may change their mind, the team may decide that they want to change up the way they will complete a task, and the Product Owner can make some changes to the Product Backlog.

Each product will go through a ton of changes throughout its process and being adaptable can make these changes easier to handle. If there is no adaptability when it comes to this process, the product is not going to be created and the entire process must be reworked.

- **More about Scrum**

Scrum is basically a framework where people are able to address many complex adaptive problems, while still being able to creatively and productively deliver products that have a high amount of value for their customers. Scrum by itself is just a simple framework that can be used to ensure that team collaboration occurs effectively, even on a complex product. Scrum is a great framework to use because it is simple to understand, can take some time to master but is worth the time, and lightweight.

It is not a methodology like some of the other Lean ideologies that we talked about before. Instead, this framework is going to implement the scientific

method of empiricism. It is going to replace a programmed approach with a heuristic one, with respect to self-organization and people to help deal with any unpredictability and when it comes to solving complex problems.

If you are looking to implement Agile into your business and your programs, then Scrum is probably the framework that you are going to work with. This is such a popular framework that many people think that Agile and Scrum are the same things. You can easily use other frameworks to help implement Agile, including Kanban, but many times Scrum is preferred because it is easy to work with and it has a nice commitment to doing smaller iterations of work.

- **What is So Special about Scrum**

With Scrum, there are a lot of things that you are going to enjoy about the framework and how it can make your product work better. The product is going to be built in a series of fixed-length iterations which are known as sprints that can give teams a framework for dealing with software on a regular cadence. Milestones, which are also the end of a sprint, are going to come often. This helps you to feel like there is a tangible amount of progress with each cycle. This quick movement can make team members feel like they are meeting goals and can provide a form of energy to the work.

Short iterations in the Scrum framework are going to help reinforce the importance of good estimation and some fast feedback from tests will help you know if you are taking the right direction on the product or if you need to try something new.

When you are working with Scrum, you are going to need to call up four ceremonies. These four ceremonies are going to include:

- Sprint planning: This is going to be a planning meeting that the whole team needs to attend. During this time, you need to determine what you and the team want to be able to complete in the coming sprint.

- Daily stand-up: This is known as the daily Scrum. This is going to be a short meeting, about 15 minutes, where the software team can get together and sync up to know what to accomplish that day.

- Sprint demo: This is a sharing meeting where the team can get together and show what they were able to ship that sprint.

- Sprint retrospective: This is a review of what went well with the process and what didn't go so well. You can use this information to ensure that the next sprint goes a bit better.

During your sprint, some visual artifacts such as burndown charts and task boards will be visible to everyone on the team and can be some powerful motivators. They can show each person of the team how far everyone has come and can even be used to show off new work during the sprint demo to motivate people even more.

- **Three Important Roles for Scrum Success**

A Scrum team can often have a different composition than a traditional project and you will see that there are three main roles that need to be covered. These three roles include product owner, development team, and scrum master. And because these Scrum teams are cross-functional, the development team is going to include engineers, designers, and testers in addition to all of your developers on that project.

First are the product owners. These are the champions of the product. These individuals need to focus on understanding all the market and business requirements and then will take this information and prioritize the work that needs to be done by the other groups. Products owners who are the most effective are able to:

- Manage and build up the backlog of the product.
- Work closely with the team and the business to make sure that everyone understands the items that are in the backlog and what needs to be done.
- Provides the team with clear guidance on which features need to be delivered next.
- Decide when to ship the product with a slant towards delivering more quickly.

Keep in mind that the project manager and the product owner are not the same people. Product owners are not going to be in charge of managing how the program proceeds. Instead, the product owner is going to focus on making sure that the team is able to deliver the most value to the business. Also, the product owner needs to be just one person. Even if there are technically several people

who can hold this role, it is best to just pick out one individual who can do the work. You do not want your development to get guidance from several product owners at once. This slows down productivity and can just make things more difficult to handle.

Next on the list is the Scrum master. These are the champion for Scrum within the team. They are going to be in charge of coaching not only the team but also the business and the product owner on how the Scrum process works. They should also spend some time learning how to fine-tune their own practice of it. An effective Scrum master really understands what work the team is doing and can ensure that the team optimizes its deliver flow. As a facilitator in chief, they are going to be there to schedule the logistical and human resources, planning out sprints, stand-up, sprint review, and even the spring retrospective.

Part of the Scrum master's job is to make sure that there aren't any anti-patterns that start to show up. One that is pretty common with teams that are new to Scrum is when the team tries to change the scope of the sprint after the sprint has begun.

Keeping your scope airtight will help you to keep on track and can be great for product planning. Plus, it is going to fend off some of the disruption sources that occur with the development team.

Scrum masters are sometimes thought of as the project managers, but these types of managers don't have much of a place in the Scrum methodology. The Scrum team can often handle itself and will organize around their own work. Agile teams are going to use a pull model where the team is going to pull a certain amount of work from their backlog and then will commit to completing that work during the sprint. This can be a great way to maintain quality and can ensure that the best performance is going to come out of its team over the long-term.

In this process, neither the product owners, project managers, nor scrum masters are there to push work on the team. They will list out the work that needs to be done, help to direct some of the things that should happen during the sprint, and more. The team gets to do a little bit of self-regulating in this process that can increase morale and provides higher quality work for the business.

And to finish the team out, there is also the Scrum team. These are the ones who will work on sustainable development practices. The Scrum team that will be the most effective is co-located, tight-knit, and between five to seven individuals. The team members need to have a different set of skills so that they can cross-train each other. This ensures that there are more effectiveness and productivity in the team and ensures that one person doesn't become bottlenecked with the process. They will often learn how to use the Scrum

methodology in order to get things done for the business.

All three of these parts need to work together in order to make a product as successful as possible. You need someone who can list out the work that needs to be done by the team, you need someone who can avoid the distractions and keeps the team on task, and then you need the actual team who is able to do the work. When all of these different parts are able to work through this well, you will see that a business can become more effective and productive than ever before.

- **A Brief History of Scrum**

Project management is hardly new.

As a matter of fact, traditional project management was born out of the experiences that project professionals had as a result of their work in various fields. Since project management was largely experienced-based, it wasn't always easy for project managers to deal with the myriad of aspects that go into completing a project successfully.

Over time, project management became more systematic in such a way that experiences were collected and built into a framework that could be used as a reference for future work done in similar fields.

This collection of knowledge, experience, and lessons learned led to the creation of the traditional project management approaches known today. In particular, the methodology espoused by the Project Management Institute.

While there is nothing wrong with this methodology, the software industry realized that traditional project management approaches just weren't cutting it for them. There had to be something more suited to the needs and characteristics of the software industry that could provide a more dynamic framework.

And so, the nineties brought about innovations in the field of programming and software development. One of the new methods that emerged is known as “extreme programming” or XP. This led to one of the most popular versions of the Windows operating system known as “Windows XP.”

Extreme programming became one of the precursors to what would eventually become known as Agile. Considering that Agile was born out of the software industry, Agile project management methodologies became synonymous with the software industry.

In the late nineties, some of the most successful software developers and programmers began putting their heads together in order to come up with a brand-new approach that could encompass the needs and characteristics of the software industry.

This led to the emergence of the Agile Manifesto in 2001.

Chapter 2 Scrum Basics

Basics of Scrum

We have established that Scrum is a project management methodology that embraces change rather than avoiding it. In addition, we have established that Scrum is ideal for those projects with a high degree of uncertainty and volatility.

So, what exactly is Scrum?

The name Scrum comes from Rugby - the sport.

Rugby is a team sport that promotes a high degree of teamwork and collaboration. In case you are not familiar with this sport, I would encourage you to check out some highlights. You will see that all players need to support each other in order for the team to be successful.

In Rugby, all players must contribute. It is not like other sports where one player can dominate while the others play a supporting cast around them. In Rugby, one player that tries to take over will end up being pounded by the opposition. That is why Rugby is a highly collaborative sport.

Consequently, the name “Scrum” is derived from the interaction and gameplay that takes place in a Rugby match.

As far as Scrum in terms of project management, it is an Agile-based methodology. This means that Scrum is intended to serve as a means of organizing and coordinating the actions among all participants in a project.

We will look at the three basic personas in a Scrum-based project and the various stages of a Scrum project. Moreover, we expect that you will get a firm grasp on the underpinnings that support Scrum as a methodology.

Scrum can be defined as follows: Scrum is a project management methodology that values simplicity and is based on empiricism. It follows a set of guidelines and promotes the self-organization of teams.

This last point is important as self-organizing teams are a key factor in the production of outputs. When a team is able to organize itself efficiently, then the outputs that result from this approach are the result of the collective effort as opposed to having a long and drawn-out process as seen in traditional approaches.

Scrum has three important characteristics which differentiate it from its core, Agile:

- It is lightweight
- It is also simple to understand

- It is tough to master

These three points place emphasis on the fact that running a project based on Scrum requires minimal amounts of paperwork and stresses developing a collaborative nature among team members and the customer.

Considering that we have defined what Scrum is, it is worth discussing what Scrum is not.

Scrum is not a magic formula that will solve all of the problems associated with running a project or getting stuff done.

Also, Scrum is not a type of predictive methodology that is based on an algorithm that can determine what you need to do at any step of the process.

The Process of Scrum

Since Scrum acts like a methodology, there is a series of steps which must be followed in order to maximize its effectiveness. As we have stressed earlier, Scrum is not rigid and it is not predictive, that is, Scrum is open and it feeds off the creativity of each individual practitioner as opposed to expecting a cookie-cutter approach.

The core of the Scrum process is called a “sprint”. A sprint is the individual iteration of the Scrum process. Thus, the entire project development is broken up into successive parts in which each of these parts must produce some type of output that can then be used as a measure of “working software”.

This last point is important as the concept of “working software” implies that at the end of each sprint, the project team must be able to deliver value to the customer. Now, whether that means an actual functioning piece of equipment depends on the nature of the project. Nevertheless, at the end of each sprint, the customer will be able to see the advancement in the project itself.

The number of sprints and the duration of each sprint depend on the project itself. So, it could be that one project may last one sprint, while there may be projects that last multiple sprints. It should also be noted that testing can be conducted as part of a single sprint, or multiple sprints, depending on the nature of the work being done.

Typically, each sprint lasts for four weeks. Although, this is only a rule of thumb. The length of an individual sprint depends on a number of factors. But, in general, four weeks is a good number. It should be noted that one-week sprints may not provide you with enough time to complete tasks while sprints longer than four weeks may drag the project on needlessly.

Each sprint begins with what is known as a “Sprint Planning Meeting”. The Sprint Planning Meeting consists of determining the objectives for the sprint that

is about to start. Given the collaborative nature of Scrum, the objectives, or user stories to be covered will be decided by the team itself. Since the team is self-organizing, the decision of what work to be done is not dictated by the project leadership, rather, the team will decide what they want to do based on any number of factors such as time constraints, the logical progression of the work begin carried out, or the team's understanding of the project's requirements.

Once the work to be done has been decided at the Sprint Planning Meeting, the team will begin working. Every day, the team will engage in what is known as the Daily Standup Meeting. This meeting consists of having a short meeting, about 15 minutes in length, in which the issues affecting the Development Team are brought up. This could be anything related to the work being done such as issues dealing with equipment, perhaps a lack of clarity on the tasks being done, or a team member asking for help.

The Daily Standup Meeting is intended to foster communication and ensure that the Development Team can communicate with the Scrum Master in order to keep the project moving along. This meeting is very much informal in nature. However, this does not mean that it is not serious.

Upon the completion of the sprint, the Sprint Review Meeting takes place. At the Sprint Review Meeting, the Development Team presents the outputs of that sprint to the customer. The customer is then able to see where the project is headed and how the team has been able to meet the acceptance criteria established as the measure of success for the project.

Once the customer has given their blessing on the outputs, and potential feedback, then the Development Team moves into what is called the Sprint Retrospect Meeting. The Sprint Retrospect Meeting consists of a debrief that the Development Team engages in, in order to discuss what went right, what needs to be improved, and what issues they encountered throughout the sprint.

The Sprint Retrospect Meeting can also segue into the Sprint Planning Meeting. In this transition, the Development Team can discuss any change requests from the customer. As we will discuss later on, it's important to note that change requests from customers should be addressed at the beginning of a new sprint so as to avoid altering the planning for the sprint currently in progress.

The process of Sprint Planning Meeting, sprint, Sprint Review Meeting, and Sprint Retrospect Meeting is called "iteration". This cycle is repeated as many times as needed in order to complete user stories. Once all user stories are completed and the product is fully functional, then the project can be considered to be over.

User stories

In short, user stories are a characterization of the end-users of the product being developed. Now, user stories do not refer to the customer per se. The customer is the person who is commissioning the project though they may not be the one who is paying for the project. The customer is the person who is motivated to have the project happen.

In the case of user stories, these are the folks who will be using the actual product. For instance, the customer is an electronics company that has asked the project team to develop a new set of earphones. The user stories will be the customer's customers, the folks who will visit a shop and purchase the earphones.

As such, the user stories must be in line with who the end-users are and why they would be motivated to purchase the product.

Since a project may have multiple user stories, the project cannot be deemed as completed until all user stories have been covered. It should be said that there is no need to develop individual products to address each user story, but rather, certain features of a single product may address individual user stories.

The Roles in Scrum

Also, there are roles to be played in Scrum.

There are three major roles in Scrum: the Product Owner, the Scrum, and the Development Team. Each one plays a specific role within the team and none has authority over the other(s). Scrum requires a flat structure. Therefore, one team member cannot expect to have control over the others.

In the case of the Product Owner, this is the "voice of the customer," meaning that the Product Owner will be in constant communication with the customer and to relay this information to the Scrum Master and Development Team. Generally, the Product Owner will communicate with the Scrum Master and then the Scrum Master will incorporate any additional information to the Development Team. The Product Owner is generally the first player on the team and will most likely be in charge of selecting the Scrum Master and the Development Team.

The Scrum Master is known as a "subservient leader". The Scrum Master is intended to act as coordinator in order to ensure that all activities are running smoothly and that the Development Team has all the tools they need in order to do their job. This implies that the Scrum Master will be working very closely with the Development Team in order to address issues as they come up. In addition, the Scrum Master must develop a good sense for dealing with issues proactively before they become any type of serious hindrance to the project's

development.

As for the Development Team, these are the individuals who are actually in charge of creating the product that the project aims to develop. They are skilled in their craft and are more than likely specialists in their field. They may have some limited knowledge and experience in Scrum, whereas, the Scrum Master and the Product Owner must be very well-versed in Scrum methodology.

The Development Team is in charge of deciding what work will be done and when it will be done. This is what happens at the Sprint Planning Meeting. The ultimate decision of what to include and what not to include is the result of the discussion among the Scrum Master and the Development Team.

Finally, the Development Team generally consists of 4 to 8 members. Any team larger than 10 members should be broken up into two smaller groups, or perhaps subgroups, in order to facilitate communication and coordination among them. This is why transparent communication and collaboration are two of the foundational pillars of a successful Scrum team.

Scrum Flow

Scrum Projects can be a lot to deal with. There are many different things going on and a lot to keep track of. Even the terms themselves can be a little overwhelming! The important thing to remember is that Scrum Projects have 5 essential activities to ensure optimal product development. Using each process will help performance and make things more efficient from the beginning to the very end of the project. This is also referred to as the Scrum Process Flow.

1. **Sprint** - Used by the Scrum Team, a Sprint is a short development period of time where the team creates product functionality. Sprints typically last between 1 and 4 weeks and can even take as little as one day. They're considered to have a short development cycle, so shouldn't take longer than 4 weeks. The planned economic value is determined by how long the Sprint is, so if it takes longer than originally thought, then that means more money spent.
2. **Sprint Planning** - The Scrum Team meets at the beginning of each Sprint, and it is here where they decide and commit to a Sprint Goal. The Product Owner presents the Product Backlog, explains the tasks, and asks the team to choose the tasks they want to work on. The Scrum Team also figures out the requirements that will be used to support said goal and will be used within the Sprint. Plus, the Scrum Team will identify the individual tasks it will take for each specific requirement.

3. **Daily Scrum** - A short 15 minute meeting that is held every day in a Sprint and includes the Scrum Master. During this meeting, the Scrum Team members coordinate on their priorities. They talk about what is most important to get done during the day, what they completed the day before, and if there are any roadblocks they might run into when doing the current day's work. Doing this helps streamline everything, and prevents any issues popping up unexpectedly.
4. **Sprint Review** - A meeting that is introduced by the Product Owner and takes place at the end of each Sprint. During this meeting, the Scrum Team shows off the working product's functionality that they completed during the previous Sprint to the Product Owner. The Product Owner then determines whether the whole Sprint Backlog is covered or not. It's possible that something might be put back into the backlog if not done correctly.
5. **Sprint Retrospective** - Similar to a Sprint Review, this is a meeting that takes place after each Sprint or project. However, it is not led by the Product Owner, but rather by the Scrum Team themselves, along with the Product Master. They discuss what went well, what possible changes they could make, and how to make those changes. They also discuss how to make the team work more efficient if there were any issues going on. It's important to speak up about issues, otherwise it could cause problems later on down the road that would prevent the project from continuing.

Basics of Scrum Theory

The basics of Scrum can be found in the empirical process control theory which itself is part of the philosophy of empiricism. The basic idea behind empiricism is that knowledge is gained most effectively via experience and making the best possible decision at the moment with the information that is available. To take advantage of this idea, Scrum uses an incremental and iterative approach as a means of control risk and increasing the predictability of the desired outcome. There are three pillars at play when the empirical process is used, adaptation, inspection, and transparency.

Transparency: Transparency is vital as it is important that those who are responsible for the outcome of a given process have a clear understanding of how it is proceeding at every step along the way. Additionally, transparency is also important to ensure that anyone else who needs to see what is going on can follow along as well. The end goal is that any observers will all have the same general understanding of whatever it is they are seeing.

One such area in which this is the case is when it comes to having a common language throughout the process that can be shared by everyone who has a hand in it. For example, those creating the product and those looking at the results will both need to have the same understanding of when the project is actually completed.

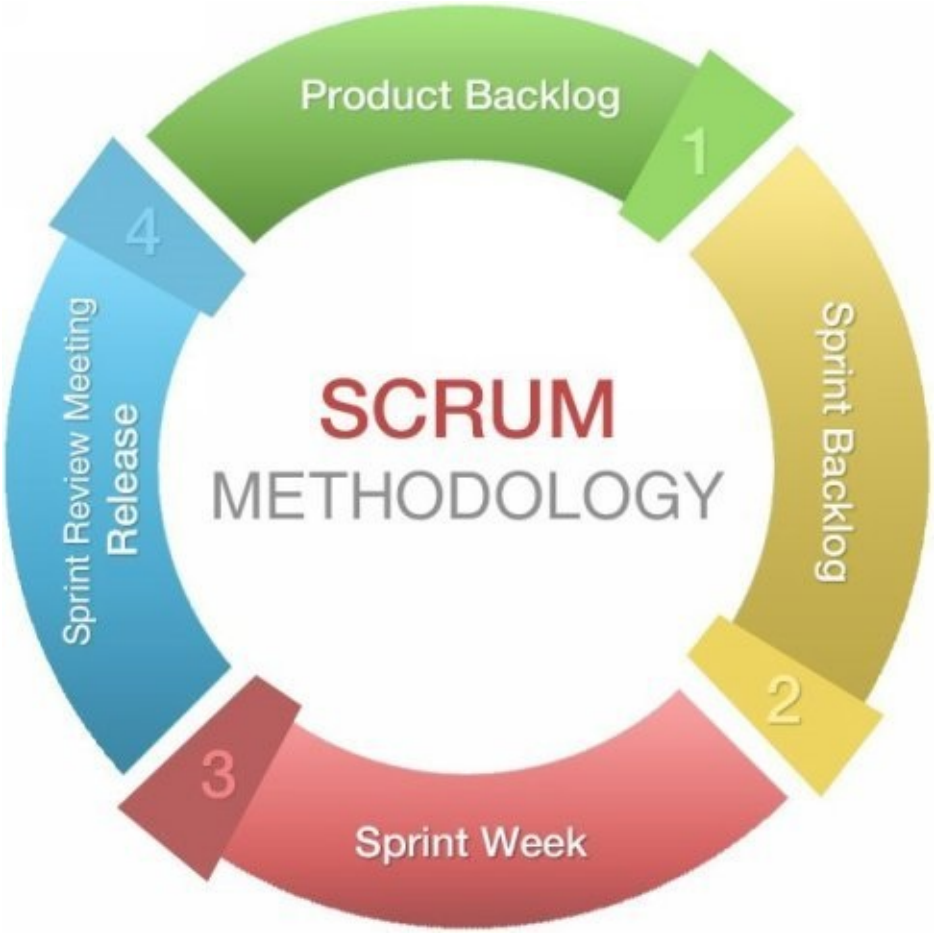
Inspection: Scrum users are frequently required to use Scrum artifacts as they progress towards a goal in order to determine potential variances that may be undesirable to that goal. These inspections should not be so frequent that the get in the way of the work that is being completed and are instead most effective when they are performed diligently by those who are skilled at inspecting this point of work.

Adaptation: When an inspector finds that some aspect or aspects of the process are deviating more than is acceptable, or that the resulting product will ultimately be unacceptable then the process must be changed as quickly as can be managed to avoid additional deviation as much as possible. When adaptation is required, there are several specific events that take place as part of the Scrum process, and they include the Sprint Retrospective, Daily Scrum, Sprint Review and Sprint Planning.

To ensure the pillars of Scrum all work at maximum efficiency while at the same time building trust among the group as a whole, the entire Scrum team needs to live by the values of respect, openness, focus, courage, and commitment. Scrum team members explore and learn to embody these values as they work with various Scrum artifacts, roles, and events. Using Scrum successfully ultimately requires team members to become more adapt at living these specific values over time. Likewise, it is important that the team feel the need to personally commit to achieving the goals of the Scrum team as well.

It is important that the Scrum team feels supported enough to have the courage to always do the right thing on a project, regardless of how difficult it might seem at the time. If the Scrum Team and its various types of stakeholders can all ultimately agree to be open about the work that is being done and the challenges being faced, then mutual respect will flourish and everyone can focus on the work of each Sprint and the ultimate goals of the team.

Chapter 3 Scrum Method



SCRUM SOFTWARE DEVELOPMENT METHODOLOGY

Implementation of Agile (Scrum) Software Development Methodology

The implementation process of Scrum's methodology can easily be explained with the help of the Scrum Famework. The framework is divided into three parts, i.e. Roles, Ceremonies, and Artifacts.

ROLES

Three defined roles are a part of the Scrum methodology. These are Product Owner, The Scrum Master, and The Team Ceremonies, which are the processes involved in the implementation of the Agile (Scrum) software development and include the following:

SPRINT PLANNING

The sprint planning meeting consists of the team, the Scrum Master, and the product owner. In the meeting, the product backlog items are discussed so that they can be prioritized and then the team selects which ones to do. The sprint planning meeting determines what will be worked on and it also helps to develop a considerable understanding of what needs to do in order to carry it out. One notable thing done in sprint planning is that tasks are measured in time (whereas before it was done in story points).

A rule of thumb, a sprint planning takes approximately the number of weeks in sprint * 2 hours (4 hours in our case).

DAILY SCRUM

The daily Scrum meeting is held daily for about 15 minutes. This is not a problem-solving meeting. The daily Scrum helps avoid unnecessary meetings. In the daily Scrum everyone answers three questions, which are:

- What did you do yesterday?
- What will you do today?
- Is anything in your way?

THE SPRINT REVIEW

In the Sprint Review (can also be referred to a Review & Demo) the team presents what has been accomplished during the sprint. It is a demonstration of new features or the existing architecture. It is an informal presentation and the entire team participates in it.

SPRINT RETROSPECTIVE

It involves looking at what is working and what is not. The time period for the sprint retrospective is around thirty minutes and is done after every sprint. It involves the participation of the product owner, Scrum master, team and even customers. In the retrospective, the whole team gathers to discuss what they want to start, continue, or stop doing.

ARTIFACTS

The artifacts can be called the tools of the Scrum methodology and include the following:

PRODUCT BACKLOG

The product backlog captures the requirements listed as items or works on the project. Each item is expressed in a way which provides value to the customer, prioritized by the product owner and reprioritized at the start of each sprint.

SPRINT BACKLOG

The sprint goal is a short statement about the focus of the work during the sprint. In the sprint, backlog work is never assigned and individuals choose their own work; the remaining work is estimated daily, and any member can add, changes, or deletes the sprint backlog. Sprint backlog determines the work for the sprint, is updated every day and each item has its own status.

SPRINT BURNDOWN CHART

The Sprint burndown chart shows the total sprint Backlog hours remaining each day and also the estimated amount of time to release. The sprint burndown chart should ideally come down to zero at the end of the sprint. The X-axis of the chart shows the time left in this sprint and the Y-axis show the hour's estimated remaining.

BENEFITS OF SCRUM

- Scrum methodology eliminates the need for comprehensive documentation
- Mistakes can be easily rectified
- Clear visibility of the project development
- Iterative in nature and requires customer feedback
- Short sprints and constant feedback make coping with changes easier
- Individual productivity improves as a result of daily meetings
- Issues are identified in advance and hence can be resolved rapidly
- A quality product can be easily delivered in the scheduled time
- Minimal overhead cost in terms of process and management
- It helps with the delivery of top-value features first
- Shorter time to market, which increases market feedback and ROI
- System is better prepared for adaption to business and external changes

PITFALLS

- Tasks can be spread over several sprints if it is not well defined
- Success and failure of the projects depends on the commitment of the team members
- Heavily relies on a dedicated Product Owner. The lack of it cascades down and hinders the quality of the backlog, which has an impact on essentially the entire process

- Works well only with a small team
- Needs relatively experienced members
- Works well for project management only when the Scrum Master trust the team

IMPLEMENTATION EXAMPLE

- A fixed time meeting is held at a fixed place each day
- The Team Lead (Scrum Master) asks the team members about what they did the previous day, what they plan to do and if any issues were observed by them
- Every day the team lead sends the report showing the daily progress and issues called a burndown chart
- A meeting is held at the beginning of the sprint by the team lead to discuss the product backlog in order to prioritize the work, resource allocation and the issues known as the sprint backlog; they meet once every week for 2 to 4 weeks
- The Product Owner defines the scope of the sprint based on the time estimates set at the sprint planning and the team's capacity for the next sprint. This scope needs to be clearly communicated to the team since completing these tickets will be a commitment for the sprint
- A daily Scrum meeting is held in order to synchronize the activities while the teams work through the spring backlog tasks
- One more time during the sprint, backlog grooming sessions are held to present and discuss upcoming user stories for next sprints. The output may be an estimation of a story in story points, or if the team needs more clarifications, questions that the product owner needs to research on a sprint review is conducted at the end of the sprint cycle and the finalized product is released
- Performances and improvements based on previous sprint cycle is discussed before starting with a new sprint; this is called sprint retrospective
- The sprint cycle continues

VIRTUAL SCRUM

WHAT IS VIRTUAL SCRUM?

A virtual scrum is an education tool used to help teach the scrum methodology to undergraduate students in the field of software engineering. Because this has become such a widely-used agile practice in the software

industry, it’s important that students get a feel for this experience in a hands-on manner. The ISISTAN-CONICET Research Institute, located in Buenos Aires, Argentina, has developed a virtual scrum program and conducted a case study to evaluate its effectiveness.

This study provided an educational and hands-on approach to scrum methodology. This virtual training tool allowed students to familiarize themselves with programs and elements of Scrum like blackboards, web blackboards, web browsers, document viewers, charts, and calendars.

This study followed 45 students using this program to complete their capstone project, as well as those not following the virtual scrum methodology. The results of the study confirmed that the virtual tool is an excellent and efficient way to teach students the fundamentals and navigation of the Scrum methodology.

AGILE SOFTWARE DEVELOPMENT AND THE SCRUM METHODOLOGY

Annual surveys have found that agile methods like Scrum have been increasing every year. The 2007 State of Agile Development Survey found that 37 percent of respondents use Scrum. And today, more than 50 percent of surveyed companies have adopted Scrum as their main agile methodology. This methodology is a dependable, collective approach to software development that can be implemented in any office. This also leverages communication and team work to succesfully manage product development.

So while this study has established the importance of a virtual Scrum, it is important to understand how the virtual Scrum can be applied in your individual or workplace life. This study was established to help users understand the framework and capabilities involved in the scrum methodology. The virtual Scrum allows users to get a 3D experience inside the world of scrum development.

This program helps users become involved in the scrum process by taking members through an incremental life cycle or sprints. These users are also able to become avatars with specific roles, including Product Owner, Scrum Master, and Scrum team.

It’s important to familiarize oneself with the basics of this methodology before experiencing the virtual scrum world.

This secondary tool, used to help people understand the process and actions involved during the scrum methodology, helps illustrate first-hand the benefits of this methodology. While reading and writing about the scrum methodology is a good way to learn, it’s especially helpful to get real exposure to the scrum

process.

The virtual Scrum not only allows users to take on important roles inside the Scrum process, but it also helps people experience other aspects of scrum development including organizing and creating user stories, planning the sprint backlog, monitoring sprint work and finishing the sprint.

All of these key steps to the process can be understood through the virtual Scrum. This teaching tool is extremely valuable to the software engineering world and many great companies are searching for engineers well-versed in this area.

For those looking to get a better understanding of Scrum and its processes, using the virtual Scrum program is an excellent option. The simulated scrum environment may be the next major trend in scrum education.

Chapter 4 Tools

Scrum Tools

Scrum tools are responsible for facilitating planning and tracking in the Scrum projects. They provide some single place where the management of the sprint backlog, product backlog, tracking and planning sprints, conducting the daily Scrum meetings, showing Burndown charts, and holding of Scrum Retrospectives can be done.

There are various types of Scrum tools. Some of these tools are open source, while others are paid. You can also get distilled versions for some of these tools. However, for you to be able to enjoy all the features of the tool, you will have to purchase the full version of the same.

These tools will make it easy for you to complete each iteration. They can also help you deliver high-quality Increments. The following are the available Scrum tools:

1. **Acunote**

This is a fast tool, and it's easy to use. The tool can provide the team with the actual progress instead of wishful thinking. For those who are in need of data-driven management, this tool will provide you with powerful analytics. This tool can also help in facilitating collaboration across the whole Scrum Team. Small teams can use the tool to keep the items to be accomplished. Large companies can use Acutenote to track the work done by many employees. The tool can also be integrated with other tools, such as source control systems, Google Apps, bug trackers, etc. The good thing about this is it's an open source tool.

2. **Redmine Backlogs**

This tool comes with a number of features that can help you manage your team effectively. It can help you sort stories in the backlogs. The tool can also give you a mechanism to display burndown charts. It provides a task board through which you can track the tasks. With this tool, one can track down any impediments for each sprint.

3. **Scrumwise**

This is the easiest Scrum tool for you to use. With this tool, you can make the teams happy and productive. You will be able to manage the tasks done by different members and track the way things are being done. The tool facilitates the creation and management of backlogs. You can also plan for sprints and releases and know whether you will deliver an Increment at the sprint's end. You can create a task board that you can use easily, and you will be able to track the events in real time. The tool can help you know whether your progress is on the right track or not.

4. **Axosoft**

This is a good tool, especially if you are developing a complex and large project. It helps you assign work effectively to the members. This tool will help plan for your project early enough. The progress of the project can be tracked visually by the use of burndown charts. Finally, you will be able to release the right product in time, which will increase your confidence. Any deviation from the right path of the project will be detected early enough for corrections to be made.

Scrum Tools of Communication

1. Burn down chart

We've glanced upon the concept of burn down charts before, but now that we have a deeper understanding on the important concepts of Scrum. So, let's have a more rigorous discussion on this vital tool of communication.

Sprint backlogs get updated everyday to reflect the amount of work remaining in designated tasks. By adding the remaining amount of work from each task, we get the total amount of work remaining for the whole sprint. The total amount of work remaining for the whole sprint is then shown visually using the burn down chart.

Burn down charts typically trend towards zero, as each task gets tackled even just a little bit everyday. Sometimes, however, the amount of work remaining on the previous day may actually be less than that of the work remaining for the next day. This usually happens when there is an error in estimating the amount of work that needs to be done to complete a feature. Some people may get a bit confused because the team worked on a lot of features on the previous day, only to find out that the amount of work remaining on the next day is still the same. This is still a result in underestimating the time it takes to finish certain features. This error may not always be avoidable, especially if the team is not all that familiar with some of the product features to be created.

While the product owner typically cannot just add new tasks to the sprint backlog in the middle of the sprint cycle, team members may choose to add tasks that they find are actually connected to the features they're working on in the current sprint. While typically changes in the sprint cycle is discouraged, if it turns out that the team accidentally left out a feature in the product backlog that's actually needed for a certain feature to work in the current sprint cycle, then by all means the team should add that task and adjust the remaining amount of work remaining for the whole sprint.

A potential problem that arises is when teams fall into the pitfall of grabbing too many tasks from the product backlog that may not actually be necessary, therefore increasing the chances of releasing a potentially shippable product late. Like having a unified idea of completeness, the team should also have a unified idea of what's necessary. What's great about being able to add tasks to the sprint backlog as necessary is that work becomes much more realistic than other methodologies that only allow changes to happen in before product development starts. Allowing necessary changes to happen even in the middle of a product cycle will result in better-built programs.

There are actually two types of burn down charts: release burn down charts and sprint burn down charts. As you might have guessed, release burn down charts show the amount of work remaining for the whole project whereas sprint burn down charts show the amount of work remaining for the current sprint cycle. One can think of release burn down charts as "zoomed-out" versions of sprint burn down charts. Sprint burn down charts are the ones updated daily whereas release burn down charts are usually updated only when sprints are completed.

Release burn down charts allow the team to see whether the product being developed will be released on time or not. If the product is in danger of being released late, the team will have to make necessary adjustments, whether it means working faster or removing certain tasks from the sprint backlog. Sprint burn down charts allow the team to see whether a potentially shippable product is ready at the end of the sprint cycle or not. Like release burn down charts, should they notice a potential delay in release, they'll have to either work faster; but unlike release burn down charts, they most likely cannot just remove tasks from the sprint backlog.

In either case, burn down charts are very useful in predicting the possible release of the product being developed. The more work days that have elapsed, the more accurate predictions can be made. Using the same charts, the team could also create a target number of hours in each day to make sure the product is released on time. Saying "we'll need to finish x amount of work in the next

sprints to release the product on time" is a lot better than saying the non-constructive "we're going to be late!" statement.

If you've noticed, the charts and logs used always show the time remaining rather than just time elapsed working on the product. In most cases, time elapsed is useless data because time spent working on something does not exactly reflect the amount of concentrated work done. It could also lead to a false sense of accomplishment (teams could mistakenly correlate elapsed work hours with completed work). By instead showing the amount of work that's still unfinished, the team can have a more accurate gauge on the remaining work they still have to do and spread them out evenly.

2. Burn Up Chart

The problem with burn down charts is that they don't show scope changes very clearly. Scope change is when tasks are either added or removed in the product backlog or sprint backlog. This usually happens when the team discovers tasks that actually need to be done prior to other already planned. Clients, customers, and product owners may also think of additional features along the way. Burn down charts may oversimplify data and lead to estimation and prediction problems in the future.

A burn up chart shows the work that is already completed and the total work with separate data points, as compared to the burn down chart, which combines the two. The total work data will show the amount of work available, which may change as tasks are added or removed. The work that's already completed will show the amount of work that's been done. As I've mentioned before, in burn down charts you may see past days that contain less work to be done than succeeding days, and that's often because of added tasks. Burn up charts show these added tasks clearly, as well as any removed tasks. They also make it easier to see why products take too long to finish or if too many tasks are added.

Just as burn down charts could be represented in hours spent or in story points, burn up charts could also be represented in these two modes. You can use either, depending on whether you're particular about the hours spent in tasks or not.

Burn up charts and burn down charts both have their pros and cons – burn down charts are much simpler to create but won't really tell you as many things as burn up charts will. Burn up charts will, for example, show you that the team had to remove tasks in order to finish a sprint, while burn down charts will only show a sudden downward fall in work remaining.

3. Task board

The task board is a visual representation of the progress of the sprint backlog. At a glance, you'd be able to tell the progress of the tasks quite easily.

Team members will update the task board all throughout the sprint. This is where tasks are either added or removed as necessary. The amount of work remaining is then updated in the daily Scrum and will be reflected in the burn up or burn down charts.

In task boards, product backlog features to be tackled in the sprint is updated to have more details, with mini-tasks necessary to accomplish the tasks. The columns usually used in the task board are:

- **To Do:** All uncompleted work generally starts here. Team members choose tasks to work on in this column and move them to the “Work in progress” column.
- **Work In Process:** All uncompleted work that people are already working on generally goes here.
- **To Verify:** If there are no separate tasks in testing the completed work, the theoretically-completed work is generally placed here to be verified - usually by the Scrum Master or other team members.
- **Completed:** Tasks that have been verified are moved to this column.

Chapter 5 Core Roles of Scrum

Core Roles In Scrum

There are various roles to be filled in the Scrum team, each carrying specific duties and responsibilities. They include:

1. **The Scrum Product Owner**

The role of the product owner is at the heart of the Scrum framework. A product owner is a doer and a visionary who paves the way for the product that meets the needs of the market. The product owner is fully aware of both the development process and the customer. They collaborate with both parties and help them make precise decisions that bring the product to life. Specific qualities of a product owner are:

Visionary

The product owner has the ability to envision the final product and communicate this vision to the team and the customer. He or she also carries the responsibility of seeing through the completion of the project. These responsibilities call for the capturing of requirements, ideas, and working closely with the team. This demands spending a considerable amount of time with the customers and understanding their needs and expectations well. After this, the product owner envisions an elegant and simple solution to resolve all these needs. He or she also receives and analyzes feedback received from the users and the stakeholders. Therefore, the product owner must keep a close eye on the project, tracking the progress made so far, and calculating the progress expected.

One thing that the product owner needs to master is the ability to embrace change and to deal with it gracefully. Scrum’s framework is built as it is to help users embrace change more easily. Since he or she is in contact with the customer, they get to speak or call the client about their needs. Sometimes, these requests will be incessant and annoying, but it is the duty of the product owner to explain the change to the client in a friendly manner. They must then come up with a simple but elegant solution that will resolve all customer needs. Changes do not only come from the customers, but the market and other forces also generate change and the product owner must come up with ways to deal with it.

Negotiator and Communicator

The job description of a product owner requires that they develop effective communication and negotiation skills. They must communicate with different entities such as the users, marketers, management, engineering, development, sales, and operations, and align them towards the common goal. They become the voice of the customer. This role calls for an occasional ‘no’ and a compromise when the suggested change could jeopardize the flow of the project in an irreparable way.

Qualified and Available

A product owner must be qualified and available to carry out the job. The work of a product owner is full-time and therefore, they require adequate time to carry out their roles without distractions. If the person is overworked from carrying out other roles besides the project, the progress and quality of the project at hand will suffer. Qualification for this role simply means that the person has full knowledge and is able to understand the market, the customers, and the team. He or she also has to be passionate about delivering the proper user experience, able to manage a budget effectively, comfortable working with a self-organizing cross-sectional team, and has the ability to guide the team through steps of development for the project.

Decisive

The product owner is looked upon and trusted to make decisions during times of uncertainty, ambiguity, and when dealing with unknowns. He or she should be comfortable with trying out new things and taking educated risks. It is also important that they be right most of the time to secure the confidence of the team and the customer. The best product owners have remarkable instincts that address the sensitivities of their target customers and when faced with critical decisions, they are able to pick the right ones, especially those that have to do with pricing, designing, and positioning.

A product owner will only do these things effectively if he or she is empowered, with the authority of making decisions single-handedly. In turn, the individual has to prove that they are qualified and committed to earning that trust.

Doer

Besides generating good ideas, the product owner should be able to see ideas and their visions through to the end.

Educated

A product owner must be educated in the ways of the business and possess the necessary skills to do the job properly. It would prove impossible for a product owner to generate ideas and alternatives to help address the client's interest without having a clue about how the system works. This understanding is also essential for interpreting and analyzing feedback. In addition, as a person who works directly with the development team, it is necessary for the owner to have a clear understanding of the development process and have a passion for working to create the most friendly experience for the users.

2. The Scrum Master

A Scrum Master's primary duty is to ensure that all members of the team stay focused on their tasks and are unhindered. The Scrum Master's role lies between being a team leader and a management role, which makes defining their traits or qualities somewhat difficult. Added to this is the fact that the Scrum Master is chosen from the employees and it may take a while for them to develop the qualities needed to fill that role.

In that regard, here are some characteristics the right Scrum Master ought to have:

A servant leader: The Scrum Master is not given charge of the team to command them with regards to what to do. Instead, they are meant to provide support to the team members by helping them recognize and solve impediments. It is not easy for an employee to transition to a Scrum Master, but a born leader will have the natural drive to help, enable, and encourage others, which is one of the critical necessities for a team.

Knowledge of the products, the market, and the domain: The more that the Scrum Masters are acquainted with their production processes, markets, and products, the easier it is for them to take note of even the smallest issues that crop up in the team and to resolve them early on. For example, the Scrum Master will be able to offer an alternative development plan if the current one is failing.

Relates well with the team and can influence them: The Scrum Master is drawn directly from the team and may take on the team leader role very easily but face challenges taking on management responsibilities. However, the Scrum

Master must have the natural ability to command the respect of fellow team members to influence them to take on specified actions.

Relentlessly pursues excellence: Success on the Scrum Master role is mainly about causing the team to improve the manner in which it carries on with its activities. The best approach is through a reflective analysis of the procedures and processes of the team before optimizing and streamlining. Many employees focus on the current task, and it is the Scrum Master's role to ask them to take a step back and help with the review, by taking note and calling out anything they might have encountered throughout the year.

Takes note of and acts on team conflict: A good Scrum Master is able to recognize team conflict in its early stages, is equipped with proper strategies to address the conflict, and employs strategies to prevent future conflict.

Occasionally allows the team to fail: An able Scrum Master knows better than to allow his team to fail. However, there are times when the Scrum Master cannot prevent it and allows the team to fail and learn. The lessons the team will learn from mistakes, sometimes, is better than the motivation the team gets from successes and good advice.

Does not mind being disruptive: A Scrum Master knows that some changes require disruption. They understand when it is necessary to be disruptive to the extent that they cause change without causing damage.

Knows how to develop a team: A Scrum Master understands the phases a team goes through, which are storming, forming, performing, norming and adjourning. If the Scrum Master is able to monitor the progression of a team, they will be able to maintain the stability of the team.

Principles over practices: Scrum practices are only viable if they are supported by agile principles. Otherwise, the plans and practices become an empty shell. A Scrum Master should understand that agile principles are at the root of successful practices and that their team must adhere to them religiously.

Encourages self-ownership: An able Scrum Master encourages the team to accept self-ownership of projects, the environment, and the task wall.

Observant: A Scrum Master keeps a keen eye on the team's daily activities.

Although he/she does not play an active role in every activity that takes place, they observe all the sessions and take note of what is being discussed in the daily Scrums. They also monitor the role that everyone plays in the stand-up.

Lead by example: A team will be quick to learn if under a Scrum Master who inspires them and allows them to unleash their abilities and potential, while still modeling desirable behavior. In a difficult situation, the Scrum Master will lead them through, showing them how to act on it, calmly and without panicking. The main focus is to find a solution to the problem and not to look for someone to blame.

An Enabler: Enabling is second nature to any good Scrum Master. The daily Scrum and other meetings become fun to attend because the team is assured of progressive thinking and ideas that will be well prepared and useful. In the end, the meetings become a source of thought stimulation and encouragement.

Averts impediments: A good Scrum Master will not only work on resolving current problems, but also works on preventing the rise of others in the future. With the experience garnered over the years, they should able to read the situation and predict the outcome, which gives them the opportunity to ward off issues even before they appear.

Able to relate with the Product Owner well: A Scrum Master works in partnership with the Product Owner, which makes it necessary for them to have an outstanding relationship. Although they have different responsibilities where the Product Owner incites and provokes the team into aggressive and creative production, the Scrum Master plays the role of protecting the team from exhaustion and fatigue. When each plays his role, the team will be challenged to excel in its activities, and at the same time is protected from exhaustion and burnout.

Is open about their experiences: Scrum Masters are quick to share the experience they have had with other people in the organization and in talking forums in conferences and seminars. These events present an excellent opportunity to educate and mentor others, while still learning from them. Writing down ideas also helps.

3. The Team

The team is made up of professional developers whose sole responsibility is to convert the action items listed on the Product Backlog into functional pieces of the final product. The 5 to 9 people size of the team is to make it manageable and to ensure that it brings together all the skills needed per project.

The responsibilities of the Scrum team include:

- The Scrum Team bears responsibility for all activities taken towards the achievement of the sprint goals.
- The Scrum Team has the duty of keeping itself organized; this is not the work of the Scrum Master.
- Once the Team is committed to delivering certain results, it has the responsibility of fulfilling its promise, making sure to deliver quality results in the agreed time.
- The Team liaises with the Scrum Master in deciding the order of priority of the items listed in the product backlog when coming up with a sprint backlog.
- Members of the Team have the obligation of attending every project-related meeting of whichever nature, and to ensure that the agendas of the meetings are addressed properly.
- The Team has to remain agile in their workstations and have to attend standups and any other events when called upon to do so.

4. The Stakeholders

Traditional methodologies used in project management take stakeholders to be all parties that are affected by the project. In Scrum, however, only three roles are recognized; that of the Scrum Master, the product owner, and the development team. Other interested parties, such as the project sponsors and the end-users, have their voice in the project through the product owner.

Among the stakeholders, Scrum specifically puts emphasis on the role of the end-user. This is because, for agile projects, the team has to ensure that each step they conduct has benefits and value for the end-user. This causes the product owner to involve the end-user in almost every step so that the user can validate the claims made by the development team. The foremost goal of an agile project is to maximize the customer’s satisfaction.

Therefore, in Scrum, the roles of the stakeholders include:

- Maintaining a healthy relationship with the product owner to learn all the details regarding the on-going project.

- Communicating their concerns and wishes to the product owner, particularly in regard to the project duration and quality.
- Be a source of regular input by answering the questions the product owner asks.
- Help the product owner to effectively prioritize some tasks to ensure that the project goes on smoothly.
- Continuously take and provide updates regarding any necessary changes to the plans.
- Provide insight, inspiration, and motivation in the project development process.

Other roles the stakeholders take up differ from one project or one organization to the next.

Non-Core Roles In Scrum

Just as there are essential core roles in Scrum, there are also non-core roles. While these roles are not mandatory for a Scrum project and might not even be as involved as the other roles, they are still very important because they can play a significant part in the projects. These roles include the Stakeholder(s), the Vendors, and the Scrum Guidance Body.

Stakeholder

Stakeholder is a term that collectively includes customers, sponsors, and users who frequently collaborate with the Product Owner, Scrum Master, and Scrum Team. It's their job to come up with ideas and help start the creation of the project's service or product and provide influence throughout the project's development. The customer is the specific person who buys the project's product or service. It's entirely possible for an organization's project to have customers within that same organization (internal customers), or customers outside of that organization (external customers). A user is an individual or organization that uses the project's service or product. Just like customers, there can be both internal and external users. It's even possible for customers and users to be the same person. The sponsor is the person or organization that provides support and resources for the project. They are also the person that everyone is accountable to in the end.

Vendor

Vendors are outside persons or organizations. They provide services and products that are not usually found within the project organization. They help

bring things in that might not have been there otherwise.

Scrum Guidance Body

The Scrum Guidance Body is optional and is made up of either a group of documents or a group of expert individuals. It's their job to define government regulations, security, objectives related to quality, and other parameters seen in the organization. It's these guidelines that help the Product Owner, Scrum Master, and Scrum Team to carry out their work in a consistent manner. The Scrum Guidance Body is also a good way for the organization to know what the best practices are, and which ones should be used in all Scrum projects. It's important to note that the Scrum Guidance Body doesn't actually make any decisions related to the project. It's instead used as guidelines and a structural way for everyone in the project organization to consult the portfolio, project, and program. It's especially useful for the Scrum Teams, who can look at or ask the Scrum Guidance Body for advice whenever they might need it.

Chapter 6 Phases of Scrum Process

The Process

Unlike the more conventional waterfall version that concentrates on a stringent step-by-step process of development stages, the iterative design is best viewed as a cyclical process. After a preliminary planning stage, a tiny handful of stages are repeated over and over, with each conclusion of the cycle incrementally improving just iterating on the program. Enhancements can rapidly be recognized and carried out throughout each iteration, making it possible for the next iteration being a minimum of marginally better than the previous.

Planning and Requirements: Like most any improvement project, step one is enduring a preliminary preparation phase to map out the specification documents, create software or maybe hardware requirements, moreover typically get ready for the upcoming phases of the cycle.

Design and analysis: Once preparation is done, an examination is conducted to nail down the correct internet business logic, database models, and the way will be needed at this time in the venture. The design phase in addition happens here, starting any technical requirements (languages, services, data layers, etc), which will probably be used in an effort to meet up with the requirements of the evaluation phase.

Implementation: With the preparation along with analysis into position, the real implementation and coding process could now start. All preparation, specification, along with design docs up to this point are coded and also implemented into this original iteration of the venture.

Testing: Once this power build iteration is coded and implemented, the next thing is going through many testing procedures to determine and find some potential issues or bugs which have cropped up.

Evaluation: Once all prior stages were finished, it's time for a comprehensive analysis of advancement up to this particular point. This enables the whole team, and also customers and any other external people, to check out where task is at, where it must be, what can or even should change, so on.

Today the real fun starts! This is the crux of entire iterative design, by which most recently constructed iteration of the software program, in addition to all responses from the evaluation procedure, is brought to the planning; development stage at the upper part of the list, so the process repeats itself all once again.

The Actual Process

The Scrum Process

The Scrum procedure is rehearsed in different gatherings and accordingly gives a few ancient rarities which are depicted underneath.

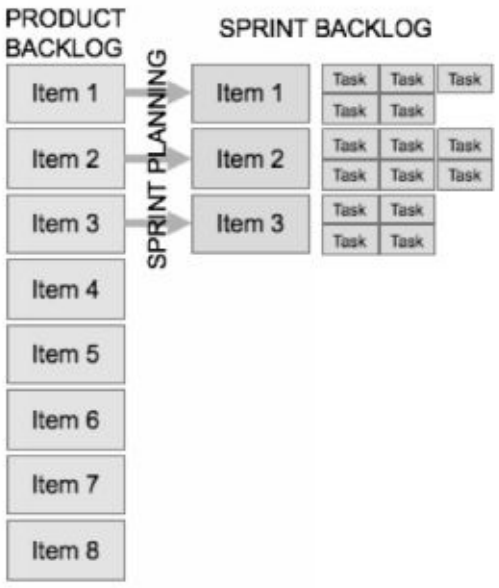
The Product Backlog

The Product Owner makes a rundown of prerequisites dependent on the portrayal of advantages – the Product Backlog. The entirety of the Stakeholders adds to its substance. This not just occurs toward the start of an improvement and at specific occasions, however as an on-going procedure. When change demands or new prerequisites rise, they are embedded into the Product Backlog. This principal receptiveness to acknowledge steady changes and view them not as deterrents; however as circumstances, prompts high adaptability in the advancement. The prioritization is frequently founded on the altruism estimation of the business or on a hazard associated with the necessity. This takes into consideration snappy response to changing conditions without risking the venture.

A Product Backlog can, for instance, have the accompanying structure:

Model Scrum Product Backlog

Table 1: Example Product Backlog



The Sprint Planning Meeting and the Sprint Backlog

The Sprint Planning Meeting happens toward the start of a Sprint and all undertaking individuals take part. This is the place the Team and the Product Owner settle on the substance of the following Sprint.

The Product Owner characterizes the most elevated need sections ("Items") of the Product Backlog and clarifies their practical use. The Team assesses the exertion of the "Things".

The Team chooses the "Things" for a Sprint from the Product Backlog as per the prioritization and afterward places them into the Sprint Backlog.

When the Team and Product Owner concur with the execution of the Sprint Backlog in the following Sprint, the Sprint Goal is characterized.

From that point forward, the Team includes the Sprint Backlog "Things" to a usage plan. These are for the most part explicit advancement exercises.

The Sprint at that point begins. For dependability, the necessities of the Sprint Backlog ought not change during a Sprint.

The Impediment List

When the main Sprint has begun, each Team Member can include the supposed hindrances (Blockers) to a rundown. Each Team Member reports their Blocker for the execution of an assignment when it emerges and puts it in the rundown of Blockers. It is the assignment of the Scrum Master to dispose of these Blockers. A Blocker might be a system condition, yet could likewise be the sit tight for an incomplete assignment. The Blocker is passed on to the next Team Members in the Daily Scrum Meeting and recorded in the Impediment

List.

Case of an Impediment List:

Table 2: Example of an Impediment List

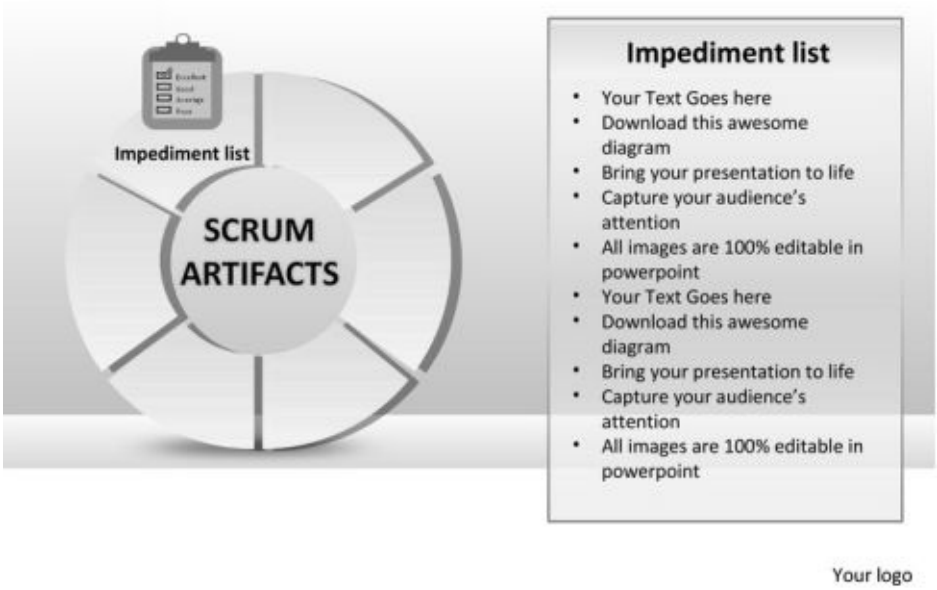


Table 2 speaks to a case of an Impediment List. Further sections can obviously be included. Where appropriate, a status for every line could be beneficial if the rundown ought not just to incorporate remarkable Blockers.

The Daily Scrum Meeting

When a Sprint has begun, the team meets at the Daily Scrum Meeting. The Daily Scrum Meeting is a casual gathering enduring greatest 15 minutes. In this gathering, each colleague needs to respond to the accompanying three inquiries:

What did I do yesterday that helped the Development Team meet the Sprint Goal?

What will I do today to enable the Development to group meet the Sprint Goal?

Do I see any hindrance that anticipates the Development Team or me from meeting the Sprint Goal?

This gathering isn't a broad exchange, however the groups short every day arranging a meeting to facilitate the vital undertakings to arrive at the run objective. It is the Scrum Master's assignment to guarantee this occurs.

The Scrum Master's primary assignment is to determine developing Blockers. His undertaking isn't to control the Team. The Team is self-sorting out.

The accompanying rules apply to the "Day by day Scrum":

The gathering consistently begins on schedule.

The gathering ought to consistently happen in a similar area and simultaneously.

The gathering will be held transparently, everybody can take an interest, yet the real jobs have select talking rights.

The gathering is constantly restricted to a limit of 15 minutes, paying little heed to the size of the Team.

The members should remain during the gathering. This should help keep the gathering brief.

The Sprint Review Meeting/Retrospective

Toward the finish of each Sprint there is a "Survey Meeting", in which the Team shows the outcomes to the Product Owner and different Stakeholders. Everybody can take an interest in this gathering and give criticism on the outcomes and their improvement. An audit of the Sprint (the review) likewise happens toward the finish of a Sprint. The entire Scrum Team is welcome to this. What ought to be improved later on is talked about at this gathering.

The Goal of the Sprint

A chief estimation of Scrum is the capacity to incorporate new prerequisites into the framework after every emphasis and Sprint. With each progression, the client gets an executable framework which, over the span of time, turns out to be progressively similar to the final result. Not at all like other iterative advancement models, Scrum centers around executable frameworks, yet additionally on really usable frameworks. In this way, the improvement of a total framework ought not be isolated into singular modules that can't reasonably be utilized without the total framework. This dodges the client not getting an item that can really be utilized by the clients until the finish of the advancement procedure and afterward having change demands emerge. The prioritization of the necessities, as indicated by the altruism estimation of the business, additionally guarantees that the client can utilize his most significant capacities at a beginning period.

What is Scrum Project Management?

Scrum is a coordinated undertaking the board philosophy or structure utilized fundamentally for programming improvement ventures with the objective of conveying new programming ability each 2 a month. It is one of the methodologies that impacted the Agile Manifesto, which expresses a lot of qualities and standards to manage choices on the most proficient method to

create better programming quicker.

What is Scrum in Relation to Agile Project Management?

Scrum is a sub-gathering of lithe:

Lithe is a lot of qualities and rules that depict a gathering's everyday collaborations and exercises. Light-footed itself isn't prescriptive or explicit.

The Scrum philosophy pursues the qualities and standards of spry, yet incorporates further definitions and determinations, particularly with respect to certain product improvement rehearses.

Albeit produced for lithe programming improvement, coordinated Scrum turned into the favored system for nimble venture the executives as a rule and is now and again basically alluded to as Scrum venture the board or Scrum advancement.

Scrum Structure (Paths, Phases, and Activities)

Planning

Planning is the first phase in Scrum. Its purpose is to determine the vision of the project, project expectations and ways to fund the project. The vision needs to describe in what ways the product will change and influence the business environment and what are the benefits to the customer. It also needs to explain how the product will be introduced to the market. Sometimes planning needs to have phases too. We can take an example of healthcare software that needs to add equipment for digital imaging such as X-Rays and MRI. The vision for the project was an adaptation of radiology operations in which radiologists can discuss and review the condition of their patients having the same digital image. Additionally, they could mark those images and access them whenever they need it.

Vision has a purpose of determining the common context for the product, thus in which environment decisions will be made. The vision of the project is also a resort used to get the necessary funding. A project proposal must show what the benefits are and how the funds will be implemented most efficiently. Vision can be demonstrated through models, words, spreadsheets, and most often it is not too long. Once the vision is presented to the possible fund sources, they decide if they want to financially support the project. After planning, the highest prioritized work is getting the funds, creating an initial product backlog along with the release plan. After that, in this phase team needs to determine

technicalities, architecture in business environments and expectations from the project.

Staging the project

Every project has its own characteristics, some projects are large or require big and numerous teams, and others are small but complex, with many teams that work together to build software. Some of these projects need a big amount of work to determine the environment that is satisfactory for product development, while others are immediately ready to go. In the phase called staging, a team assesses different aspects of the project. In this phase, the team also creates additional product backlog needed for further development.

Staging has many sub-phases. Every sub-phase represents one aspect of the project and it is assessed as such. During the staging phase, many different paths can be used in each sub-phase. Activities needed for each of these paths are not an actual projection of work. In fact, activities, in this case, are used to prioritize or add requirements for the product backlog. The actual work is a result of these requirements, and that work is performed during the development phase and by the project team. Staging is important because it helps with building mechanisms for coordination and communication between all Scrum units, or between self-organized teams.

Final sub-phases are significant for project initiation in staging. These sub-phases get the project started. This means that there are enough staff members in each team and that the team already got together with the Product owner to get project details. In these sub-phases products, the owner gives as much information about the project as possible shearing the vision and everything else that is acquired during the planning phase. Sometimes, this means that the team needs to get out and become familiar with the targeted environment. It is a part of the experience needed for the team if they want to understand the purpose of the desired system and its implementation. If, by any chance, team members never worked in Scrum before, it is highly recommended that the Scrum methodology is introduced to the team through concrete training.

Development

All activities connected to the complete process of product development are done in the development phase. This phase is divided into sprints that have the purpose of developing increments of working functionalities for each sprint individually. Sprints consist of sprint planning, daily meetings, and daily

reviews. If the projects are complex, then the architecture of the software is significantly complex too, which means that there can be more than one team working together on software development.

In this case, sprints are used to build a product backlog that will be coordinated by models and designs that were implemented in the first sprints done by one team. This team distributes further context and information about the product to every other team.

If the development environment is upgraded and that upgrades are needed before all teams can deliver potentially shippable increments, the work is scheduled for the next sprint as highly prioritized work by the common product backlog.

This kind of work functions in parallel with the actual development of required functionality. Although the architecture of the work environment is requested in some of the first sprints, every other sprint in this phase also has to deliver working functionality with a suitable increment.

Release

When it comes to the release vision of the project, its functionality is based on a combination of benefits that are expected from the project along with the overall costs, time frame, and availability.

After every sprint in the development phase, product owner reviews if the desired functionality is achieved and what is the right time to release the product.

After all these inspections and reviews, the Product Owner establishes if the conditions for release are met and the project enters the Release phase.

This phase is made of two different kinds of activities. The first one is the activity that sets additional requirements for the product backlog. These requirements should polish the product and convert it from potentially shippable to an actual releasable product. If the development phase failed to meet any of the requested assignments, the product owner addresses more sprints to fix this point.

Another activity in the release phase refers to one or more sprints that need to be added as a consequence of inadequate increments created during the development sprints.

The release phase can be established only after the development team creates working functionalities that suffice final product backlog requirements.

Chapter 7 Scrum Artifacts

Product Backlog, Sprint Backlog & Increment

You will most likely think of archeology when you read or hear the word ‘artifact.' As the traditional definition refers to an object or tool made by man or a ‘work of art,' so Scrum also uses this term to refer to the main tools used within the framework. The three main artifacts for Scrum are the product backlog, the sprint backlog, and the product increment. These artifacts, or tools, all operate with the same objective in mind, which is to increase transparency and always ensure a shared understanding of the work. Scrum artifacts provide insights to a team, which facilitate understanding of a particular project and the tasks or activities to be completed. Let's begin with the three main artifacts as well as additional artifacts used in many Scrum projects.

Product Backlog

The product backlog is a prioritized list of requirements, functions, and features, which a final product needs to include. This list is versatile in the sense that it is constantly updated by the product owner, and these changes are always communicated to the Scrum team. As a product makes progress and is delivered to a client in increments, the backlog continues to grow as feedback is received. The product backlog is the single source of agreed requirements for a final product, and the content and validity of this list is controlled by the product owner. Items listed on a product backlog usually comprise a description, estimate, and value and often described as ‘user stories.'

Backlog grooming, also known as product backlog refinement, is the practice of a Scrum team meeting after every sprint is completed and running through all the content of the backlog to ensure it is both relevant and useful. These meetings include every member of the Scrum team, including the Scrum Master and Product Owner. The objective of these reviews is to ensure the project is aligned with the product roadmap constantly. Backlog grooming may include some of the following actions:

- Review of the most important elements at the top of the prioritized backlog list
- Removal of elements which are discovered through the most recent sprint, to no longer be relevant
- Adding additional features, requirements or necessary functions which

the final product needs to include
- Re-ordering items on the list as priorities of the overall project change
- Formulate answers for questions posed by the client based on the most recent increments delivered
- Ensure that all team members are up to date on the overall product roadmap and any necessary changes, which have been made or are still required to be made.

The Product Owner will make use of the product backlog during sprint planning to outline what items within the project are a priority. From this point, the team will define which priorities they can tackle in the next sprint. If you're wondering what a product backlog looks like, it is more than just a to-do list. Items on the backlog should always have the same characteristics.

- All entries should always add value to the overall product and to the client.
- All the entries must be prioritized in terms of importance to meet the overall objectives of the project.
- Items that appear first on the list will have greater detail than those towards the end of the list.
- Entries into the backlog are all estimations
- The backlog is a living document in the sense that it is constantly changing as the needs of the project or client change.
- The backlog does not include action items or routine tasks.

Let's delve into what each of the above characteristics involves:

Entries should add value

Every entry in the backlog must relate or refer to some kind of customer benefit or consideration in relation to the final product. Any entries without a clear client value should be removed. Instead, entries could include the exploration of client needs, various technical possibilities, any tasks needed to launch the product or any other functional or non-functional needs of the client or project. This could also include tasks relating to creating the environment, which the product needs to operate within.

Living document

The Scrum product backlog is continuously updated throughout a project. When necessary, new items are added to the list; additional detail may be added

to existing items, items may be re-prioritized or even deleted. Although this differs from traditional methodologies of software development, this approach allows for maximum value for the client.

Detail differences

Depending on where an item features on the product backlog, so it's level of detail will differ. Items towards the bottom of the list have fewer details than those at the top of the list, which is of a higher priority. The items, which are to be tackled in the next sprint, are the items that will appear with the most detail on the product backlog. The reason for this is due to the ever-changing nature of a Scrum project. It makes little sense to update requirements that are not part of the next sprint as their requirements will change before they become more relevant.

No action items or routine tasks

The backlog should not include items that are routine or considered action items. It only includes items related to creating product value to the client.

Items are Prioritized

All entries in the backlog are prioritized. This prioritization is done by the product owner and Scrum team collectively, although it is the product owner who owns this process. When it comes to this task, value-added, costs involved as well as risks are the most common factors taken into consideration. The Scrum master uses this backlog and the way it is ordered to determine what item should be dealt with next.

All entries are estimated

Items in the backlog are estimations by virtue of the fact that they are ‘user stories' and, by definition, need to be estimations. In many cases, Scrum teams do not have the information readily available to provide specifics for user stories, and so estimations are considered efficient. This ties into empiricism on which Scrum revolves around.

The collaboration needed to maintain the Scrum Product Backlog creates buy-in from all team members and ensures clarity of exactly what needs to be achieved.

Sprint Backlog

Definition:

While the product backlog refers to the product and end desired result of the

project, the sprint backlog refers to the list of objectives for a specific sprint. The sprint backlog is the outline, completed by the team, of what specific increment or function is going to created or developed in the upcoming sprint. The backlog includes exactly what tasks need to be completed for a project to be considered successful.

The sprint backlog is useful to team members on a daily basis as it provides a clear representation of the progress the team is making, and a real-time picture of what work is still remaining. Every sprint backlog should include at least one high priority item, a user story, which has been identified in a previous sprint's retrospective.

Updating the Sprint Backlog:

The sprint backlog should be updated on a daily basis without fail. All items for a project will be contained in the product backlog. A team will select a high priority item and break it down into tasks that can be completed in one day, and the task selected should be achievable in one day. As a team member, with a task you know, is achievable in one day, you will have an indication of your progress and whether you are ahead or behind rather quickly. This progress is then reported on in each daily Scrum.

Should the sprint backlog not be updated daily, it will become evident rather quickly that you are not able to see movement on the backlog and that work is being completed. You may also lose track of whether or not your team is falling behind. With the constant updates on progress, a Scrum team is able to react to slower progress and course correct rather quickly. Scrum teams who do not exercise discipline in updating the sprint backlog may not pay as much attention to the backlog and will be more likely to fail or veer off course.

What is in a Sprint Backlog?

In simple terms, a sprint backlog is a plan or guideline for the sprint. It will usually contain user stories from the product backlog, which the team aims to complete. Each item will be broken down into a prioritized task list for the team to execute. The Scrum team will usually use their current rate of production, or how much work they know they can complete within a given time, to predict what they can accomplish during a sprint. Once this has been estimated, the development team can compare the total hours for every task against the capacity of the team to do a final audit on expectations for the sprint.

When is the sprint backlog finalized?

Once the items on the sprint backlog have been selected from the product

backlog and finalized, the sprint backlog is locked in, meaning no changes can be made. Once sprint planning is complete, a team can add but not remove items from the backlog. The product owner is the only Scrum member who can remove items from a sprint backlog if they decide that it no longer adds business value. During a project, items that are removed cannot be replaced by additional tasks.

What Is the Difference between Product Backlog and Sprint Backlog?

A product backlog is the list of user stories or tasks to be completed for a project as a whole. The sprint backlog is a subsection of the product backlog as it contains selected user stories, which are further broken down into tasks for the team to address in an individual sprint. During sprint planning, a product owner will select an item on the product backlog and clarify its specific detail. The team will then evaluate and determine if that item can be executed in the next sprint. What the team deems to be achievable is then added to the sprint backlog.

Who Manages the Sprint Backlog?

The sprint backlog is managed predominantly by the development team. It is considered the best practice to update this backlog on a daily basis so that the team constantly has a live reminder of progress, which is visual and keeps the team motivated. Should any obstacles or impediments to work occur, the development team should discuss this with the product owner as soon as this becomes apparent. Should any obstacles become apparent, the communication between the team will include managing expectations of internal and external stakeholders as well as possible solutions.

Product Increment

What is it?

Increment refers to gradual steps taken towards the completion of a goal. Product increment, in relation to the Scrum framework, refers to the sum of all the product backlog components completed during a sprint. These are combined with the increments of all other previous completed sprints. A sprint requirement is that the increment produced should be in working order and provide tangible value to the end product.

The specific definition of increment for a Scrum team needs to be created, and a consensus reached between team members of exactly what amounts to a completed increment of product. This will usually vary from project to project and team to team. As team members start a new project with the previously

conceived idea of what an ‘increment' should be defined as, it is important for the whole team to agree on what they believe this amounts to. Once the defining parameters have been agreed upon, this definition is used as the yardstick to determine whether or not the work completed during a sprint is classified as ‘increment.' The practice of determining a collective definition or standard that the whole team ascribes to is a common theme in the Scrum methodology and is also used to guide the team when determining how many product backlog items it could reasonably complete during a sprint.

At the end of each sprint, as we know, the objective is to produce or release functionality in the form of an increment, which contributes to the overall product. Once the product owner has assessed the quality of the increment, they can decide to release the increment immediately or not. This decision will often be determined by whether the whole Scrum team believes it amounts to an increment of the required standard as required by the project. As Scrum teams work together over multiple sprints and mature as a team, coming to these conclusions is usually a much quicker process. In addition, the collective definition of increment may become more stringent, resulting in higher quality outputs by the team.

Who delivers a Product Increment?

It is the development team that will deliver product increments on a regular basis. This product increment should reflect the product owner's overall project road map as well as support the Scrum team's definition of "Done." Although it is the development team that delivers product increment and hold the majority of this responsibility, it is still a shared objective carried by the whole Scrum team.

Delivering Product Increments: Challenges

In some cases, Scrum may be implemented across companies with team members across departments. Breaking down silos that may exist, by virtue of differentiating departmental work practices, is often a big challenge and may take considered effort to address. It is important to get buy-in from the key stakeholders within each department so that these silos may be broken down. In addition to these silos, many companies will have their own work practices and values, which may overshadow a Scrum team, regardless of the Scrum's specific work parameters or values. Should any of these work practices be counter to Scrum methodology and how the framework needs to operate to be successful, the product owner will need to determine how to overcome these.

How is Product Increment created?

The product increment is created during a sprint as a result of all related development tasks, including design, analysis, testing, integration, and build. These actions result in creating a partial addition to what is envisioned as the final product. When using Scrum, teams will work on one feature of a final product at a time. It is planned in such a way that at the end of each sprint, a workable and value add feature is produced. This increment of product is then tested and integrated with previously delivered features. Prior to this testing and integration, a product is not usually considered to be "Done." Once testing is complete, the product increment is delivered to the client for feedback, which is taken on board before the next sprint commences.

This feedback allows the Scrum team to adapt and course correct. It provides useful input for a sprint retrospective and allows the team to decide, based on the client's needs, what the next sprint should involve. The feedback also allows for quality control checks and for iterative improvements, a key element of the Scrum and agile approach. The key principle is to use this continuous feedback to carry out effective iterations

What is the outcome of the Product Increment?

The product increment is useful to all team members for a Scrum, as well as external stakeholders. It allows a product owner to assess the current return on investment (ROI) from the final functionality that will be delivered to the client at the end of each sprint. Deliver increment at regular intervals also creates a sense of unity between team members as they receive regular feedback and recognition. The feeling of making progress and achieving success is a lot more ‘real' than traditional project management methodologies where feedback is only received at the end of a project.

Definition of "Done"

When a product backlog point or increment is described as "Done," the entire Scrum team must reach a consensus on what "Done" means in the context of their specific project. This definition and what it means to the team, may need to be discussed at considerable length to ensure all team members, with varying degrees of experience and expertise, agree. This definition of "Done" for the Scrum team is then used to assess when work is actually complete on the product increment.

Just as consensus on the definition of increment will influence sprint planning, so the same consensus is needed amongst the team to determine what they collectively deem as "Done." Each sprint's purpose is to produce and deliver increments of functionality that adhere to the specific Scrum team's

definition of "Done." As with product increment, as teams mature their definitions of "Done" are usually expand to include more stringent criteria for higher quality. The longer a team works together in a solid team, the higher the quality that can be achieved.

User Stories

The word "Done" is used most often by the delivery team of a project. In general terms, this means that a product owner has received an increment of the final product, or the final product as a whole, reviewed and accepted it. Once it has been accepted, this user story is marked as "Done" and is added to the team's progress. Examples of milestones or increments of product that could be declared "Done" are the acceptance of criteria which are met, the passing of functional tests or having code reviewed and approved.

Chapter 8 Scaling Scrum

The term scalability refers to the process of taking a defined process or framework and expanding upon the process so as to create a larger impact. Some processes or practices are easier to scale than others, and often the question is raised as to whether or not Scrum is scalable, and if so, how is this best done. Scaling Scrum, for example, could involve taking mechanisms from one Scrum team and implementing them in multiple teams for larger projects.

So, the question is, ‘Is Scrum scalable?' Initially, Scrum was thought only to be applicable to teams who work on smaller projects, and that is was not suitable for application across multiple teams for larger projects. However, this was only based on the fact that Scrum had not yet been used on larger scale projects, and since its inception Scrum has been applied and successfully scaled.

So, when and how should a team make a move to scale a Scrum project? The answer to this question usually depends on the nature of the project and at what level a team would like to scale. Scaling usually occurs at one of three different levels as scaling can take place across projects, programs, or portfolios. Depending on what level a team would like to scale at determines how much coordination is required. When it comes to scalability, additional resources and project managers may be necessary to ensure that development stays on track.

When it comes to Scrum teams, it is usually recommended that teams stay under ten members. In the event that an organization wishes to scale their Scrum projects, it is recommended that a bigger team is divided into smaller groups who meet regularly to discuss their progress and report any issues or concerns. Keeping cadence with these meetings and ensuring they happen at regular intervals is crucial and could be managed by a project manager.

We previously discussed the Scrum of Scrums, which is where multiple teams synchronize their projects. Each Scrum team would select a team representative who join the Scrum meetings and update on the team's progress, challenges they may be facing, breakthroughs they may have had, as well as coordinate any future activities with other teams. When it comes to deciding how often a Scrum of Scrums should meet, it is the size of the project, level of interdependency, complexity, and recommendations from upper management, which should be taken into consideration.

As we know, Scrum recommends that meetings and collaboration take place face to face. Although not impossible to implement Scrum over different geographical locations, it does take a lot more coordination and effort. When it

comes to scaling a project with teams in different offices, the Scrum of Scrum meeting scan takes place using video conferencing tools. When larger projects are deployed, a chief of Scrums will need to hire, and this person is responsible for facilitating all the sessions between the Scrum of Scrums. The chief of Scrums will determine exactly when meetings should take place and outline their agendas. These meetings, like other check-ins, will involve the sharing of updates on progress, challenges, and recognized dependencies across projects. Once teams receive an agenda from the chief Scrum master, they should prepare their updates ahead of the meeting. Should any particular members of a team be facing challenges, these should be raised in these meetings, as it is often likely that other teams may experience the same challenges. This allows teams to share in problem solving and overcome obstacles at a quicker rate.

When these meetings take place, each team representative will usually provide an update that answers four main questions. These include what the team has been working on since the last meetings, what the team plans to work on between now and the next meeting, asking other representatives if there are any other elements of development the other teams dependent on them for, and finally, what could the team be working on that would directly impact the other teams.

The outcome of these Scrum of Scrum meetings is usually better coordination of work, which is carried out across teams. This is specifically true when there are tasks that run across different teams, and there are high levels of dependency. This ensures that if there are any obstacles, discrepancies between expectations, or change in deliverables, they are exposed and addressed as soon as possible. These meetings also operate as an open forum when representatives can provide honest feedback and receive recommendations or input from other representatives.

In the event that a project is scaled above the capabilities of a Scrum of Scrums framework, an additional meeting framework is created where a representative from each Scrum of Scrums is sent to a larger meeting known as the ‘Scrum of Scrum of Scrums’. This allows all projects that are related to each other to be coordinated in such a way that allows for maximum quality and timeous output. What is important to note is that this type of coordination, especially should the larger teams be distributed across geographical locations, will require much larger coordination and management effort.

Scrum of Scrums

When dealing with larger Scrum teams, a technique called a “Scrum of Scrums” can be used in order to manage the daily tasks of such a large group.

Unlike the previous example, a Scrum of Scrums does not have multiple teams with multiple Scrum Masters. This concept considers having one large team which is divided into subgroups. This plays off the initial consideration we had about maintaining ever-numbered groups.

Let's consider the following case.

A Scrum team consists of 16 members, given the size and the scope of the project to be completed. The budgetary restrictions do not allow for individual Scrum teams, say, two teams of eight, or even three teams of six. So, there is one Scrum Master and one Product Owner looking after all 16 members.

In order to make the division of labor much easier, the Scrum Master has divided the 16 members into four subgroups of four. Each subgroup is intended to cover a specific amount of work during each sprint.

Since the Scrum team has a Daily Standup Meeting, the Scrum Master has to make a choice: does the Daily Standup Meeting include all 16 members of the team, or could the meeting include less members?

There is one problem with each approach.

In this first approach, having all 16 members may drag out the meeting far longer than necessary. Each member may have something to say and that, in itself, may lead to some disorganization in the meeting. So, the meeting could ultimately become counterproductive.

The second approach, working with fewer individuals poses a problem since it would be difficult to choose who is in the meeting and who isn't. One workaround could be to designate a "group lead" and have this person attend the daily meetings. That would reduce the number of participants from 16 to 4. But that poses another issue: Scrum does not advocate "leaders" or folks "in charge" within each group. Bear in mind that Scrum has a flat structure meaning that everyone has equal participation.

So, what's the solution?

A Scrum of Scrums.

In a Scrum of Scrums, each of the members of the subgroup gets a chance to represent the team at the daily meeting. Since everyone is equally involved in the process, everyone would be equally qualified to represent the group at the meeting. A good rule of thumb is to rotate reps every day. Unless sprints were unusually long, rotating reps every day makes the most sense.

Now, it could be that a sprint would take 8 weeks. In such a case, the team might decide to rotate reps on a weekly basis. That might work insofar as giving each member the opportunity to handle the responsibility of attending the meeting on a more permanent basis.

Now, the one consideration under this approach is to avoid having one “permanent” rep. The whole point of avoiding a permanent person representing the group boils down to avoiding having any semblance of hierarchy within the Scrum team. As such, having a rotating representation is the best way to go.

Ultimately, the point of having such a set up is to reduce the likelihood of one team member taking over the rest of the group. While there are people who are more naturally inclined to a leadership role, the fact of the matter is that by having rotating roles, you are able to make sure that everyone gets equal participation, thereby fostering a collaborative approach within the team.

Scaled Agile framework

So, when Agile is scaled up to a larger framework, keeping track of the workflow and making sure that the collaborative nature sticks get harder and harder. So, it then becomes a challenge for Scrum practitioners to enforce a discipline whereby team members have equal participation, work is conducted in as per the user stories outlined at the beginning of the sprint, and the overall nature of the project is completed within the aspects pertaining to the Agile mindset.

As such, large projects can be perfectly accommodated within an Agile framework with the basic difference that the ultimate work breakdown structure, as seen in the Product Backlog and the individual Sprint Backlogs may be divided among individual teams rather than one collective work divided among individual team members.

In order to make this concept resonate on a deeper level, consider the following situation:

A food manufacturer wants to produce a new low-calorie drink. Since the market is demanding low-calorie drink options, the management team at this corporation is looking to produce a low-calorie drink that can compete with other brands in similar categories.

Consequently, a team has been assembled in order to determine the right formula for this new drink. The company has decided to invest in research and development into producing a new drink rather than just rebadging an existing one.

So, the CEO of the company has called in one of the company's production engineers and named them "project engineer." This engineer is aware of the value in the adoption of a Scrum approach to this project. As such, the project engineer has assumed the role of Scrum Master. The client is the CEO who has been designated by the company's board as the person in charge of making sure the project comes to fruition.

Thus, the project engineer comes in a decides to build multiple Scrum teams that can work at once, especially since the company has determined that a proposal for a new drink needs to be ready before the holiday season. In this case, the project would have about three months to be completed. This means that there is no time to be wasted.

The Scrum Master, or project engineer, decides that the best course of action is to assemble a research team which can look into the actual production of the drink, a production team which can look into the logistics of bottling and distribution and a marketing team which can come up with a brand name, graphics and art for the sole purpose of producing the new drink.

All three teams will work in tandem in such a way that they are all in a concerted effort toward producing a new, low-calorie drink.

The Scrum Master has decided to coordinate each team through a Scrum of Scrums approach. Under this approach, the main idea is to ensure that all of the teams are working in tandem, thus moving the project along with multiple groups working in sync.

In every sprint, each of the groups works on the same user story, except from their own angle. So, the user story is based on a consumer who is young and is

looking for a low-calorie option that can help them cut down on their sugar consumption while maintaining the flavor of other drinks on the market.

Ultimately, the entire team arrives at the same destination with a finished drink, distribution and production considerations in check, and a solid marketing campaign in place. The final presentation of the product is presented to the company's board. The project is approved by the board and is ready for an official product release, that is, a formal launch into the marketplace.

This example underscores the fact that in order to tackle a rather large project, the project engineer, or Scrum Master, decided to find the best way to come up with a new drink and set it up in the market. Whether or not the product is a success once in the market remains to be seen. The most important thing to keep in mind is the need for the project team to find the best way to work on the project in a collaborative manner which underscores the needs and requests of the customer, thus delivering value at all times.

Coordinating multiple Scrum teams

In the previous example, we were able to see how multiple Scrum teams can be coordinated through the use of a Scrum of Scrums. While using multiple Scrum Masters might be a good solution, it might not be the most practical when taking budgetary restrictions. The previous example highlighted how the company wanted a new low-calorie drink and just tapped one of their best engineers to take on the project.

Sure, this approach would provide a more "predictable" outcome, the fact of the matter is that traditional project management methodologies are used to produce predictable results. When something such as the creation of a new product requires the intervention of a project team, then there is a clear need for the each of the members to find the best way to make their work count. That is why Scrum can be used across a number of different industries and fields.

The fact that Scrum has a cross-cutting appeal, it allows project managers to find an effective way of running a number of teams simultaneously. This implies that the project team is able to make sense of the various tasks that need to be done in such a way that the project team is able to get a firm grasp on the various objectives that need to be completed throughout the lifecycle of the project.

For instance, a traditional project management methodology would not allow all three teams to run simultaneously. The project team would have to wait until there was a successful formula developed for the drink, while the remaining teams might be able to come up with ideas but wouldn't be able to work until the previous step had been developed.

Distributed Scrum teams

One of the most important factors in a Scrum project is finding the right way to manage multiple teams working at the same time. Ideally, these teams would be distributed evenly at the same physical location. This is the concept of co-location, and according to Scrum doctrine, it is the best way in order to ensure that teams are working collaboratively.

But what happens if there are teams distributed in different geographical locations?

Is it possible to run a sprint when team members are not physically located in the same place?

The answer is yes.

Modern technology has facilitated communication in such a way that teams can work in different geographical locations and still achieve positive results. The fact that Scrum teams can communicate via the internet allows for teams to work remotely without having to bridge the distance by moving and thereby increasing the cost of the project.

However, such an approach is not for everyone.

Some folks can work perfectly fine on their own. They are capable of producing results without having someone looking over their shoulder to see what they are doing. Then, there are others who are not so adept at working on their own. They may find themselves in a situation in which they need to have someone telling them what to do in order to get things done.

While Scrum does not advocate telling people what to do, the psychological pressure of having a team working around them helps them to focus on getting work done. Nevertheless, using Scrum to work with teams located in different geographical locations is possible.

Yet, there is one limitation. Such projects need to be those which can be assembled over the internet, or which do not require the production of a physical good. In such cases, there would be a point where team members would have to come together to produce the actual, physical good.

How and when this approach is used depends largely on the nature of the project. So, it’s up to the project manager to find the right way to make sure that the project team can work despite distance and other considerations that may negatively affect the team’s overall performance.

The use of Scrum in maintenance projects

At this point of the book, we have known how Scrum is used to create value, especially when there are new products being produced. Yet, we have not known what happens in the case of such projects where there is nothing new being

created, but maintenance is the primary focus.

One such example could be an update to an existing product. This update could be commissioned by the customer or stakeholder in order to fix bugs found in a project.

Consider this situation:

An automobile manufacturer has realized that one of their recent models has been shown to have transmission problems. So, the automobile manufacturer has decided to commission a Scrum team to find a good solution to the transmission issues for this vehicle.

The Scrum team has come together, run several sprints, detected the problem, and proposed a corrective solution. The latter sprints were intended to provide more thorough testing to the new transmission so as to avoid the same problems the previous transmission had. In the end, the results were positive, and the new transmission has been cleared for new models.

Chapter 9 Benefits

Benefits of Scrum

Following are ten significant benefits that Scrum gives to associations, groups, items, and people. To exploit scrum benefits, you have to trust in observation, discover progressively about the Scrum structure by utilizing it, and persistently review and adjust your execution of scrum.

Better quality

Tasks exist to achieve a dream or objective. Scrum gives the system to nonstop criticism and presentation to ensure that quality is as high as could reasonably be expected. Scrum guarantees quality by the accompanying practices:

Characterizing and explaining on necessities without a moment to spare, so information on item includes is as significant as could be expected under the circumstances.

Joining day by day testing and item proprietor criticism into the advancement procedure, permitting the improvement group to address issues while they're new

Normal and ceaseless improvement of scrum group yield (item or administration) through run audits with partners

Directing dash reviews, permitting the scrum group to consistently improve such group explicit factors as procedures, devices, connections, and workplaces

Finishing work utilizing the meaning of done that tends to improvement, testing, joining, and documentation

Diminished time to advertise

Scrum has been demonstrated to convey an incentive to the end client 30 to 40 percent quicker than conventional techniques. This abatement in time is because of the accompanying elements:

Prior inception of improvement because of the reality the forthright documentation periods of cascade ventures (which regularly take months) are inevitable by having a committed item proprietor installed inside the scrum group to dynamically expand necessities "without a moment to spare" and give continuous explanations.

Most noteworthy need necessities are isolated from lower-need things. Steadily conveying an incentive to the end client implies that the highest-esteem and - hazard prerequisites can be conveyed before the lower-worth and hazard necessities. No compelling reason to hold up until the whole venture is finished

before discharging anything into the market.

Usefulness is swarmed to finish each dash. Toward the finish of each run, scrum groups produce working item and administration augments that are shippable.

Expanded degree of profitability

The reduction so as to showcase is one key explanation that scrum ventures understand a better yield on speculation (ROI). Since income and other focused on benefits start coming in sooner, prior aggregation implies higher absolute return after some time. This is an essential precept of net present worth (NPV) computations. Notwithstanding time-to-showcase benefits, ROI with Scrum likewise increments by:

- Ordinary input through dash surveys legitimately from partners, including clients, empowers course rectifications early, which is less expensive and tedious than later simultaneously.
- Less expensive imperfections because of mechanization and direct front testing imply less squandered work and quicker arrangements.
- Diminishing expenses of disappointment. In the event that a scrum venture will come up short, it flops prior and quicker than cascade ventures.
- Higher consumer loyalty
- Scrum groups are focused on delivering items and administrations that fulfill clients. Scrum empowers more joyful task supports through the accompanying:
- Teaming up with clients as accomplices and keeping clients included and connected all through ventures.
- Having an item proprietor who is a specialist on item prerequisites and client needs.
- Keeping the item build-up refreshed and organized to react rapidly to change.
- Exhibiting working usefulness to inner partners and clients in each dash survey.
- Conveying item to end clients quicker and more regularly with each discharge instead of at the same time at the end.
- Steadily subsidizing tasks as opposed to requiring huge in advance responsibilities.
- Higher camaraderie

Working with cheerful individuals who make the most of their occupations

can be fulfilling and fulfilling. Self-administration puts choices that would regularly be made by a chief or the association into scrum colleagues' hands. Scrum improves the spirit of colleagues in these manners:

Being a piece of a self-overseeing and self-arranging group enables individuals to be imaginative, creative, and recognized for their aptitude.

Advancement groups may sort out their group structure around individuals with explicit work styles and characters.

Scrum groups can settle on choices customized to give balance between colleagues' expert and individual lives.

Having a friend association with a business agent (item proprietor) on a similar group adjusts specialized and business needs and separates authoritative hindrances.

Having a scrum ace, who serves the scrum group, evacuates hindrances and shields the improvement group from outside obstructions.

Concentrating on manageable work practices and rhythm guarantees that individuals don't wear out from pressure or exhaust.

Working cross-practically permits improvement colleagues to adapt new aptitudes and to develop by instructing others.

Empowering a hireling head approach helps scrum groups in self-administration and effectively staying away from direction and-control strategies.

Giving a domain of help and trust expands individuals' general inspiration and assurance.

Having eye to eye discussions lessens the dissatisfaction of miscommunication.

At last, scrum groups can concede to rules about how they work to take care of business.

Expanded cooperation and possession

At the point when scrum groups assume liability for activities and items, they can deliver extraordinary outcomes. Scrum groups work together and take responsibility for and venture execution through the accompanying practices:

Having the improvement group, the item proprietor, and the scrum ace work firmly together consistently

Directing run arranging gatherings, permitting the improvement group to sort out its work around educated business needs

Having day by day scrum gatherings where advancement colleagues compose around work finished, future work, and barricades

Leading run surveys, where the item proprietor plots his prioritization choices and the improvement group can exhibit and talk about the item

legitimately with partners

Leading dash reviews, permitting scrum colleagues to survey past work and prescribe better rehearses with each run

Working in a colocated domain, taking into account moment correspondence and coordinated effort among advancement colleagues, the item proprietor, and the scrum ace

Settling on choices by agreement

Progressively important measurements

The measurements that scrum groups use to evaluate time and cost, measure venture execution, and settle on venture choices are frequently more important and more precise than measurements on conventional undertakings. On scrum ventures, measurements are progressively pertinent on the grounds that

The individuals who will take the necessary steps, and nobody else, give exertion evaluations to extend prerequisites.

Courses of events and spending plans depend on every advancement group's genuine exhibition and abilities.

Utilizing relative appraisals, as opposed to hours or days, tailors evaluated exertion to an individual improvement group's information and abilities.

In under one moment daily, designers can refresh the torch outline, giving day by day perceivability of how the advancement group is advancing toward a dash objective.

Toward the finish of each dash, an item proprietor can look at the undertaking's real cost (AC) in addition to the open door cost of future activities (OC) against the worth that the present task is returning (V) to realize when to end a venture and start another one. You don't have to hold up until the finish of a venture to recognize what its worth is.

Improved progress perceivability and presentation

On scrum extends, each individual from the undertaking group (which incorporates the scrum group and partners) has the chance to know how the venture is going at some random time. Straightforwardness and perceivability make scrum an introduction model to enable the venture to group precisely recognize issues and all the more precisely foresee how things will go as the task advances. Scrum activities can give an elevated level of progress perceivability by:

- Putting a high incentive on open, genuine correspondence among the scrum group, partners, clients, and any other individual inside an association who needs to think about an undertaking.
- Day by day scrums that give day by day knowledge into the

improvement group's prompt advancement and barriers.

- Day by day scrums around task sheets empower engineers to self-arrange and distinguish the most elevated need errands for the afternoon.
- Utilizing the data from everyday scrum gatherings, run torch diagrams, and errand sheets permits the task group to follow progress for singular runs.
- Run reviews permit scrum colleagues to recognize what's functioning admirably and what's not to make activity arrangements for development.
- Showing achievements in run audits. Anybody inside an association may go to a dash audit, even individuals from other scrum groups.
- Expanded venture control

Scrum groups have various chances to control venture execution and make revisions as required as a result of the accompanying practices:

Modifying needs all through the task at each dash interim instead of at significant achievements enables the association to have fixed-time and fixed-value ventures while pleasing change.

Grasping change permits the venture group to respond to outside factors like market requests.

Every day scrum coordination permits the scrum group to rapidly address issues as they emerge and swarm together to complete necessities.

Every day updates to run overabundances imply that dash torch graphs precisely reflect run progress, allowing the scrum group the chance to make changes the minute it sees issues.

Up close and personal discussions evacuate barriers to correspondence and issue goals.

Run audits let venture partners see working items and give item proprietors the criticism they have to ensure the safety.

If done correctly, the Scrum framework can have the following positive effects on the team and the company:

- Better product quality that meets or exceeds client expectations
- More frequent code deployments (which means visitors to a site or users of the software or app can use the product and benefit from it through bug fixes, added features, or upgrades)
- Faster lead time from committing to the project all the way through to code deployment

- Lower change failure rate
- Reduced costs of deployments
- Improved employee motivation
- Improved organizational performance as a result of faster delivery of better or superior product
- Improved overall productivity
- Improved market penetration, increased market share, and higher profitability

Companies that utilize the Scrum framework find themselves able to adapt to the ever-changing requirements of clients and stakeholders with less chaos and confusion. They know that they're creating a product that will remain relevant even if there are sudden changes in requirements. It's a win-win situation for the Scrum Team and the stakeholders.

Chapter 10 Scrum Mistakes to Avoid

1. Anticipating that Transformation should Agile and Scrum to Be Easy

Very regularly, somebody will get a book on Agile or Scrum, start cleaving up prerequisites into client stories, start every day stand-up gatherings, create programming in 2-multi week runs, and afterward call themselves Agile. Simple, correct? They will probably observe some improvement in their capacity to react to change and may even give working programming quicker – to some time. It won't be excessively long, however, until the guarantees of Agile become less obvious, groups battle to keep up the pace, programming doesn't generally coordinate client desires, and afterward Agile is regarded a disappointment. Nimble change requires some serious energy and quite often begins muddled. Genuine change uncovered existing corporate and culture issues that must be managed – issues, for example, poor correspondence, absence of responsibility, doubt, and so on. Successful Agile change is regularly an absolute culture change. Give it time, and be all set through the torment and protection from social changes.

2. Doing the Practices Without the Principles

Accomplishing the simple things like actualizing Scrum gatherings, filling the Scrum jobs, and utilizing appropriate Scrum antiques is great, however is just half (or less) of the fight. The Agile standards are what make the practices function admirably, and make them reasonable over the long haul. Standards are a lot harder to consolidate than rehearses, which is the reason numerous organizations miss the mark – they don't do the hard parts. Utilizing procedures without understanding why you are doing them can prompt disappointment. Lithe is about individuals, collaborations, and culture, not procedures, practices, and apparatuses.

3. Muddling the Agile/Scrum Startup

Do all that you can to keep Agile new companies straightforward. Nimble tasks can be fruitful without the most recent, coolest coordinated effort or lifecycle instrument. Stickies on a divider, assignments in a spreadsheet, and a physically created torch diagram will take care of business. Investing important energy getting an instrument going as opposed to getting individuals cooperating is concentrating on an inappropriate thing. The Agile Manifesto places higher incentive on people and communications than on procedures and devices.

4. Driving a Scrum Team Like a Project Manager

A "direction and control" mindset is counter to the Agile system. A pioneer allocating undertakings and managing exertion is an Agile enemy of example. Extraordinary Agile groups are self-sorting out, the Scrum Master is a hireling chief, and groups figure out how to turn out to be better at cooperating and conveying more prominent worth all the more proficiently by normal investigation and adaption. Regularly the exercise is found out better by understanding (fortunate or unfortunate experience) than by simply being determined what to do. Scrum Masters and Agile mentors control more than they drive.

5. An Un-Ready Product Backlog

An item accumulation that isn't "prepared" is one of the most widely recognized explanations behind run disappointment and for unmotivated groups. It is likewise a main driver for low conveyance speed and not conveying high worth. Most new Product Owners aren't prepared to be profitable all alone. They need guidance, instructing, and hand-holding for the initial scarcely any dashes as they figure out how to create and keep up an item build-up that has enough important highlights assessed at a significant level, and organized by business esteem. Setting up the accumulation well in front of the following sprint(s) is an absolute necessity. You never need the group to come up short on work to do, and that work must be of most noteworthy incentive by then as organized by the Product Owner. Being a Product Owner can be tedious. Set the correct desires, give all the preparation, and help the Product Owner to keep the progression of significant worth coming.

6. Imparting "Through" the Scrum Master

Something I normally see in new Scrum groups is individuals utilizing the Scrum Master to convey their messages to other people. For instance, an engineer has an inquiry concerning a client story; rather than going straightforwardly to the Product Owner, he/she messages the Scrum Master to get the data. A key Agile rule is conveying up close and personal at whatever point conceivable. The time it takes to make the email would almost certainly have been every one of that was expected to find the solution legitimately from the partner. Be that as it may, for some, specialized individuals, up close and personal correspondence is a startling thing when they're accustomed to living in their work-space world, without conversing with individuals. This is a social or character issue that must be survived. It sits around idly and, all the more significantly, expands the danger of miscommunication.

7. A Product Owner Who isn't Available Or Involved

The Product Owner job can be very tedious. Numerous who are new to the job are not prepared for the responsibility, or simply don't have a clue about that they should be so included. Cooperation is basic in the Agile world. Businessmen and engineers need to cooperate to deliver programming that the business needs. This occurs by steady correspondence, coordinated effort, and short input cycles to approve or make course revisions. A training I love to see is the Product Owner so engaged with the everyday action of the task group that the Sprint Review is absolutely a convention in light of the fact that the Product Owner has just observed a few cycles of the highlights all through the run and has guided the group to manufacture precisely what the business needs. That is an excellent thing.

8. Careless Daily Stand-ups

The day by day stand-up meeting is significant from a few angles. It puts individuals eye to eye each day for 15 minutes, powers correspondence and joint effort, and gives perceivability and straightforwardness into the task. For such a key gathering, it's essential to set the correct desires in advance so the group pays attention to it. This may sound activist, yet participation at the day by day stand-up is rarely discretionary. Start on schedule and complete on schedule. Adhere to the three inquiries (what did I achieve for the task yesterday, what will I chip away at today, what impediments are blocking me from finishing my work on schedule). Try not to permit side discussions, talks, or critical thinking during the stand-up; those should all be possible after the stand-up is done. This gets the group in the method of regarding the group and individuals' time, and they figure out how to convey better by adhering to the targets and being brief.

9. Not Conducting Retrospective Meetings After Every Sprint

One of the twelve standards behind the Agile Manifesto is "At ordinary interims, the group considers how to turn out to be progressively compelling, at that point tunes and modifies its conduct in like manner". Lamentably the Sprint Retrospective is frequently treated like an extra or an extravagance, and performed just "if there's time". The truth of the matter is, Agile is about alterations to a great extent, calibrating and reacting to change. It's extremely difficult to alter and calibrate on the off chance that we don't delay to discover where modifications are required. Business as usual isn't Agile, persistent improvement is.

SOME COMMON PROBLEMS THAT A SCRUM MASTER FACES

The Scrum Master has the main role of facilitating this work technique and ensuring that there is no obstacle in the team's efforts at achieving their sprint target. However, little problems and unforeseen mistakes always occur, and some are so common that all scrum masters should have some ready or prospective options prepared for them. It is not only time-saving and effective, but also allows for the sharpening of one's crisis management capabilities in the long run.

One of the most common issues that occur is that the Product Owner is not able to give the group the product backlog in time. Not knowing the priorities of the Product Owner, it is challenging for an advancement group to go forth in action.

In such a situation, the scrum master can select from among different alternatives, relying on the exact scenarios. Either the whole team could be allowed to take a break from running, especially if the hold-up is supposed to be by a day or more. On the other hand, a development group could also continue with its planning conferences without giving concern to stockpile, specifically if the group has already finished some effective sprints.

The team can then develop an outline of concerns if they recognize with the general direction the product is taking and present it to the Product Owner for approval or change. Apart from this, the group could take this break as a chance to examine their work and gather feedback to more enhance the scrum procedure.

Though, to guarantee that the scrum pattern is followed, a strict policy must be followed where such breach of the process is inappropriate. If extreme situations are dealt with, the sprint completion can work upon a design of rewards as well. Apart from this, in most situations, overseas units are normally uninformed of Scrum and how it works. For that reason, correct training, albeit a concise, brief one must be conducted for, otherwise the sailing will not be smooth.

So many scrum masters also face the question of whether it will be better for a whole development group work on a specific aspect, finish the matching sprint and move on to the next, or the group be shared various different groups handing different elements at the exact same time.

The solution to this problem depends on numerous aspects like the size of the team, the time constraints, the nature of the task, and so on etc. Often a master might choose to try both the strategies before selecting one as the requirement or choose not to have a basic approach at all.

A knowledgeable scrum master usually does has some such solutions at hand to deal with such hindrances. Still, being prepared ahead of time is always best,

for it displays foresight and effectiveness that are both very important for business matters.

WHAT MAKES A TERRIFIC PRODUCT OWNER?

Here are a few traits that I feel are very important to be an efficient Product Owner:

Overall Commitment - complete engagement with the Scrum group is necessary. While it may not be possible to be physically present in a Scrum team room all of the time, the PO ought to be readily available via email, social media, messenger, and so on. The idea is to appear friendly so that the group members don't think twice to voice their questions.

Product Backlog owner - the PO owns the Item Backlog. This is a purchased list of all the functions that are preferable in the product. The PO continually orders this according to changing business top priorities and must be on hand to answer any questions concerning the backlog.

Subject Matter Professional - As an expert who must know the business and the product inside out, the PO is the go-to person for the Scrum group to answer any item associated questions. He is the sole authority on the item and must be able to clarify any business reasoning or in-depth performance that needs to be built in, and hence, should assist the group as they work on their Sprint tasks.

Collaborative - Being collective and being a great communicator is vital for the PO. He needs to have the ability to position himself 'at par' with the Scrum group and be open to deal with them at a peer level. A significant change in mindset might be needed here! Despite the fact that the person who is the PO is much higher in the organizational hierarchy, he/she needs to mix into the group and work shoulder to carry to attain the typical goal.

Scrum Vs Extreme Programming

Agile Process

The Agile Process or software application development refers to a set of software application development methods that are based upon iterative development. In this procedure, the options and requirements both evolve mutual partnership between cross functioning groups. These teams are self-organizing in nature.

The Agile software development technique normally promotes a regimented kind of job management process which encourages:

1. Frequent adjustment and inspection.
2. Self-organization and accountability.
3. A management philosophy which promotes teamwork.

4. A firm approach which brings into line the development with client needs and company objectives.

5. And a group of best engineering practices having an intention to allow for fast delivery of good-quality software application.

Extreme Programming (EP)

It is a software development methodology with an intention to boost software responsiveness and quality to the unstable requirements of consumers. This introduces checkpoints and enhances the performance in a way that the new requirements from customers can be adopted.

The benefits of Extreme Programming are:

1. Unit testing of all code.
2. Avoiding programming of features till needed.
3. Programming in pairs or carrying out extensive code evaluation.
4. Clearness and simplicity in code.
5. Unstable customer requirements better understood.
6. A flat management structure.
7. Regular communication between the developers and even with the client.

The disadvantages of Extreme programming are:

1. No documented compromises of user conflicts.
2. Unsteady requirements.
3. Lack of overall design file or specification.
4. Incorporates insufficient software application design.
5. Demands meetings at recurrent intervals at high cost to clients.
6. Can expand the risk of scope creep as a result of the lack of comprehensive requirements documentation.
7. Needs excess of cultural change to adopt.

Differences between Scrum and Extreme Programming(EP):

1. The time period for iterative sprints is different in both approaches.
2. Changes aren't enabled by the Scrum groups throughout their sprints. Whereas Extreme Programming teams need to be far more reasonable to changes.
3. Work is done by EP teams in strict concern order. Whereas in the case of Scrum, the Product Owner focuses on the set of activities.
4. EP does prescribe some engineering practices; Scrum doesn't.

Chapter 11 Scrum Certification

PROFESSIONAL SCRUM

As was noted, getting educated about various aspects of Scrum is a good way to ensure your Scrum projects end in success. However, not every team member might be interested in pursuing certification-level Scrum training. For them, perhaps attending one or two-day Scrum appreciation workshops or seminars may be a good starting point.

For those team members that are interested in going further with their Scrum training, there are a number of certification options available as proposed by the Scrum Alliance. Here are a few to consider:

- **Certified ScrumMaster® (CSM)**

A Certified ScrumMaster® serves as a vital resource to help Scrum teams make proper use of Scrum, thereby enhancing the chances of a project's success. As a CSM, you will gain a deep understanding of Scrum values and practices, and their application, and will provide the Scrum team expertise and knowledge that typical PMs don't usually have.

As a CSMs, you will learn how to act as a "servant leader", which empowers you to lead the team without formal authority over them. You will also master the principles of the Scrum framework, and learn how to help the team navigate them. Most importantly, you will gain valuable skills on how to insulate the Scrum team from internal and external distractions that often serve to derail Scrum projects.

- **Certified Scrum Product Owner® (CSPO)**

The Product Owner role is a pivotal position in the broader Scrum Team. As a CSPO, you will acquire in-depth knowledge of all necessary terminology, principles and practices that will allow you to successfully fulfill your responsibilities as a Scrum Product Owner. As noted earlier in the book, Product Owners wear dual hats - that of a conventional Product Manager as well as a Project Manager. As such, the CSPO designation will be of immense value to anyone desirous of stepping into that challenging role.

- **Certified Scrum Developer® (CSD)**

Scrum product development is unlike the traditional Waterfall phased-project cycle approach. The CSD course will provide developers all the knowledge to sharpen their Agile development skills. In addition, developers will also master the science behind incremental development as advocated by Scrum, instead of a delivery at end-of-project lifecycle approach.

CSD has the edge over non-Scrum colleagues in that they not only learn Agile engineering, but are also exposed to the basic principles and practices of the Scrum framework.

- **Certified Scrum Professional® (CSP)**

Certified Scrum Professionals (CSPs) are in-practice CSMs, CSPOs or CSDs that wish to take their Scrum certification to the next level. Every project delivery methodology can be stretched to its limits, and that's when organizations see additional benefits. This training will enable you to acquire additional skills and knowledge to help you challenge your Scrum Teams to extend their current boundaries of Scrum practice.

- **Certified Scrum Trainer® (CST)**

When Scrum practice has made you perfect in the science of Scrum, it's time to learn the art of teaching Scrum. As a CST, you'll learn everything there is to know about translating your wealth of Scrum knowledge and experience, and passing it on to others.

Every organization that's committed to Scrum should consider having at least one CST on board. CST's will not only help Scrum practitioners in the organization keep their skills current, through continuous training, but could also wear the hat of Scrum Master or Product Owner, if required.

- **Certified Scrum Coach® (CSC)**

For CSP's that wish to elevate their Scrum credentials, CSC might be the answer. As a CSC, you must be able to demonstrate that you have all the practical and theoretical knowledge of Scrum to qualify as a Coach to others - individuals, groups, and organizations.

One pre-requisite to attaining the CSC designation is that you must be able to prove that you have helped at least one organization successfully adopt Scrum in the implementation of real-life projects.

Chapter 12 Applying Scrum

Staying up with the pace may be difficult. It requires to win a battle against yourself and to become able to gather and analyze evidence to challenge yourself. Timothy Gallwey introduces the concept of the “inner game” in which your first self is always judging and not trusting your second self (the inner one). It’s vital to find tools that will help you in keeping the pace by becoming able to analyze the facts in a way that is free from judgment. Scrum can really help you in building the structure that can help you in reaching that objective.

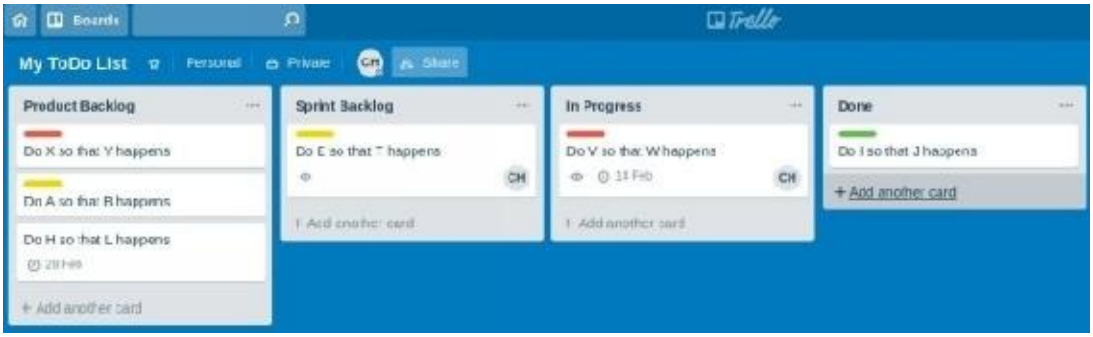
The first thing to apply Scrum in the SSD manner is to define your sprint length. This timebox will evolve according to what works for you. The main criterion is to find out what is the shortest amount of time that it takes, on average, for you to reach a set of tasks that are oriented to a specific goal. The sprint should anyway be no longer than 1 month. If you decide for a 1-month sprint, it means that you will have 12 sprints in a year and every month you will follow the 4 main events of Scrum to manage your progress. In SSD, your team may be physically built only upon yourself, so there will be only one sprint backlog for all the products that you will have to handle. Remember that for some of them, the product owner maybe someone else (e.g. your partner) and this means that when planning you will have to handle any dependency and priority that you cannot judge alone. Involving the other product owners in your planning is key to prepare a good sprint backlog that will help you in reaching goals.

Remember that you may be working alone to your sprint backlog, so be honest, try to focus and to avoid putting too many things that will make you feel overwhelmed. Again, your day is made of 24 hours and you cannot change that fact. You need to sleep, to eat, and to do many mandatory things just to live, so your capacity is fixed.

At the beginning of your sprint, you will have to decide what your goal will be. That’s the sprint planning. Things like “I want to pass the exam X”, or “I want to lose 1 kg” can be goals for a sprint. Having to handle more products, you may try to put more than one goal for unrelated products, given that you are sure you can pursue both without interferences. For example, if your goals are to pass an exam and at the same time lose 1 kg, you will have to be sure that the time you use to get fit, does not get in the way of the time you use to study. Moreover, if you will have ambiguous situations in which you are not sure of what you need to do, you will have to give your goals an order (not just a

priority, since they may have the same value). Having a clear order will help you in deciding how to spend your time in case of ambiguities.

Once you have defined your goal, you will have to pick the things you want to do to achieve your goal and move them from your product backlog to your sprint backlog. The sprint backlog will be the main tool to visualize what you expect to do in the sprint. Having those items clearly visible will help you stay motivated, see your progress, and keep concentrated. You can build your backlog with whatever tool that works for you. It can be an app like Microsoft One Note or Evernote, where you can create pages or sections where to register to-do items for each product, or more advanced applications that help you handle to-do lists with attributes or even Kanban boards like Trello. Remember that the backlog is a living document so it should be easy to modify and to sort things out. You must be able to clearly see priorities and, in case, to show what's in your pot to someone else (e.g. to raise transparency).



Once you have your sprint in place, you will do a daily Scrum with the objective of checking if you are progressing towards your goal, if there are any impediments and what will be your plans for your next 24 hours. There is no best moment for taking those 15 minutes. Some may find it useful to do it in the morning while others during the day. I found that the best moment for me is in the evening, just before going to bed. I look at my day and search for what I learned from what happened. I try to find out what blocked me from progressing and what things worked or made me grow. Finally, I look at my sprint backlog and see what I want to achieve the next day.

Managing the backlog is something that happens at any moment. Adding a Trello card or writing a note on Evernote is something that you can do whenever a new product backlog item comes into your mind. There will be time to refine the backlog and get rid of what you put there that may no longer fit your purpose.

At the end of the sprint, you must find your way to do a review and a

retrospective. It’s vital not to skip those two moments. The first one will be the event in which you will check with yourself, free from any biased judgment, if you achieved your goals and what is the result. Passing the exam, checking your weight or counting a certain number of occurrences will be your proof. Here you will be using the definition of done to understand if you really made progress or not. Your objective during the review is to refine your backlog and to add new items (or to remove others) according to what you achieved. For example, let’s say you did not pass an exam, you may put in your backlog an item that remembers you to study an additional book. If possible, invite other stakeholders to your review or simply ask for feedback. If you had a goal that other could observe, simply have a casual chat with them and ask if they noticed anything that can prove that you succeeded. Learn to deal with feedback of any type, from constructive to destructive and gather everything that can help you in refining your backlog.

Finally, find at least one hour per sprint to do a retrospective. This is probably the most important moment for your growth and your path to being happier. The retrospective is a moment in which you will look at what you have done and find out ways to improve your processes and your relations. It may vary from finding the best transportation medium to get to work, to improving your relationship with your partner. The key element is to keep the focus on one problem at a time and to find actionable items. It’s important to distinguish between action points and actionable items. An action point is when, by looking behind your back, you see that you may have required a better communication to improve your life at home. An actionable item is something tangible that you will do the next sprint, like “leaving a post-it with the reminder of what to buy” to make it easier for your companions to go grocery shopping.

A good retrospective will pass through 5 stages: setting the stage, where you will break the ice and start discovering what’s your purpose for that moment. Gather the data, so, looking at what you tracked that can help you understand what did not work in relation to the identified problems. Generate insights, where you will start giving a meaning to the data you gathered. Find what to do, in which a list of actionable items will be generated. Closure, where you will wrap up what you just decided and commit yourself to improve. Even if you do a solo-retrospective, it’s very useful to visualize your thoughts. A retrospective could last even one hour, so you may generate data and insights that after 20 minutes of self-reflection may just disappear from your vision. So, it’s important to find your way to keep everything you start analyzing in front of you. Using sticky notes on a wall may be a solution. There are many ways to implement the 5 phases of a retrospective, but you will have to find what’s working for you. In

general, the most effective way is to start by looking for the right questions instead of the right answers. Once you find a good question to start, you can begin writing down your thoughts into the cards and then post them onto the wall so that you can visualize whatever you are thinking and realizing.

Chapter 13 The Scrum Framework

Delivering a project in accordance with the requirements of various stakeholders is an inherently challenging task. However, Scrum has made that challenge easier to navigate by prescribing a "framework" for conducting the project management process.

The Scrum framework revolves around some basic "Scrum values" that practitioners must adhere to, including:

- Commitment
- Openness
- Focus
- Courage and
- Respect

Within these value parameters, complex projects can be delivered through collaboration and effective teamwork.

The main focus of successful Scrum is to try and effectively manage a Product Backlog of deliverables that represent the "whole". This, in turn, is done by breaking up the "whole" into smaller deliverable backlogs known as Sprint Backlog. As each Sprint is completed, it adds value to the deliverables produced by previous Sprints, so that the final Sprint marks the culmination of the project.

Within this framework, Scrum uses Scrum Teams that are governed by predetermined Roles, Events, Artifacts, and Rules.

Scrum Teams

In the real world, Scrum Teams are a collective name given to several sets of roles that are performed by individuals and groups working on a Scrum driven project. The key to a successful Scrum project is to ensure that you put the right team in place, and that everyone:

- Agrees to pursue common goals
- Embraces a common set of rules and norms
- Respects each and every member on the team
- Values all input given by every member

Without these prerequisites in place, a Scrum project is doomed to failure

from the outset.

While many novice PMs think of Scrum (Agile) teams in the context of "hitting the ground running", the truth is that because of their cross-functional nature, that might often not be the case. Veteran Scrum PMs will, however, always build flexibility in their plans for the Forming, Storming, Norming, Performing and Adjourning cycles (as defined by the Tuckman model, and adapted subsequently - more on this later!) to play out before the team gets down to productive business.

Scrum teams are, by definition, cross-functional in their setup. That means you will likely have people on the team with diverse skill sets, many of who have never worked together (as a coherent team), and some that will probably be part of a team for a very short duration, and then be replaced by others.

Scrum Roles

Scrum, as an agile method of handling project development, decentralizes the responsibilities and authorities once held by the Project Managers. The aim, as has been explained previously, is to create a project development environment that is more flexible to change and capable of spontaneously reacting to this change. This does not mean that the Scrum Project Management is, or can possibly be, devoid of reporting authorities.

Arguably, every PM model has to be bound and governed by certain rules, which maintain its functional integrity. And by rule, these Rules have to be enacted by one or more authority/reporting figures. Therefore, even though the new model decentralizes the authority once held by the project manager, it cannot simply eradicate it.

The fundamental responsibilities needed to create, retain, and manage sufficient structure in the team are redistributed to various new designations.

It is in this context, therefore, that the Scrum Team must be organized around specific team roles.

Scrum Master: This role ensures that the Team follows Scrum best practices, and adheres to the Norms and Rules that the team agrees to follow. Typically, this role is one of a: - Protector (who shields the team from external disruptions) - Coach - Facilitator - Moderator - Cheer Leader At project initiation, the Scrum Master is usually unable to directly contribute to actual Sprint goals. His/her main goal will be to shape the team, and to get them to focus on delivering on committed goals. However, with the passage of time, as the team settles into the "performing" stage, the Scrum Master could potentially start doing "real" work.

Ideally, the Scrum Master should have the confidence of all team members, and must not have any conflict of interest (real or perceived), when it comes to his/her relationship with any of the team members.

Product Owner: This is a combination of the role performed by the traditional "Product Manager" and "Project Manager". The Product Owner acts as the go, between to the Scrum Team and other stakeholders, and is responsible for making sure that the Team is working on the right deliverables at the right time.

In addition to working closely with the Scrum Team on other agreed roles, the Product Owner primarily: - Manages the Scrum Product Backlog - Decides what deliverables must be produced, and when - Controls the Release Schedule of the completed deliverables - Seeks any clarifications (from external sources) for the Scrum Team about Product Backlog Items (PBIs) Communication, between the Scrum Team and diverse groups of stakeholders, is a primary role of the Product Owners. The role should therefore be filled by someone with strong written and verbal communication skills. Additionally, perform this role requires someone with good negotiation and stakeholder management skills.

The Team: In addition to what has been said earlier about Scrum Teams in general, these teams have certain other characteristics that set them apart from teams using other project management methodologies (like Waterfall). The core Scrum Team: - Is often relatively small, usually between 5 to 8 individuals. Additional Subject Matter Experts (SMEs) can be called upon as needed - Does not have sub-teams - it functions as an entity unto itself - Works with great degree of autonomy as a self-organized unit - Is very much self-governed, taking their direction from within the team and the Rules and Norms they have agreed to adopt - Works best when staffed with full-time resources - Should preferably be collocated As a group, the team is responsible for: - Producing and maintaining certain Scrum artifacts (more on this later), primarily the Sprint Backlog - Organizing, running and participating in the Daily Sprint Meeting - Working collectively to produce a shippable deliverable at the end of each Sprint - Ensuring that status updates to team dashboards, specifically the Sprint Burn down Chart, are regular and accurate.

Each of these roles is linked to one another across multiple layers of management ***i.e.*** no one Scrum Role is accorded all the responsibilities of the Project Manager. Rather, many responsibilities of the traditional PM are left unassigned ***i.e.*** to no one in particular. Hence, the Scrum Roles prioritize and assign them amongst one another.

The Scrum Model disseminates the responsibilities across the Roles, and by keeping each other co-dependent instills the need for greater collaboration, openness in communication, and transparency across all stages of project development. In order to win, sports teams need to be nimble, just as project teams must be agile to be successful. When we look at how "typical" projects work, we see striking similarities to the sports world. Individuals (Players) form groups (Teams) and work on specific tasks (Plays) to deliver successfully (Win) on commitments made to stakeholders (Fans). Periodically, there are frictions (Infringements) amongst individuals and groups, and project members have to regroup (Scrummage) to sort things out.

Management experts took all of those similarities, wrapped them around a proven body of project management knowledge, and created an agile approach to managing and delivering projects, which they called "Scrum".

In developing the Scrum methodology, experts realized that there needed to be a certain discipline behind the science of project management. What they discovered is that projects can consistently be delivered to spec, on time and within budget, if PMs:

Organize the business into smaller self-governing, cross-functional teams;

Organize work into smaller chunks of deliverables;

Rank, prioritize and estimate completion and delivery;

Organize delivery of small "working components" into shorter fixed-timeframe (1 to 4 week) Iterations (or "Sprints")

Consult with customer’s/end users and organize release plans based on inspection results of each iteration

Optimize the entire process based on retrospective review following every iteration

This is the central idea from which Scrum evolved.

Scrum Events

Scrum projects are carried out through a series of events, which are geared towards producing all of the project deliverables. Additionally, there are other Scrum Events which serve to manage timelines and quality of the deliverables, as well as conduct self-appraisals of the overall Scrum (and Sprint) processes that were followed, and adopt measures to streamline them where necessary.

The Sprint: within the Scrum Framework, all project activities designed to deliver items in the Scrum Product Backlog are performed via an event known as the Sprint (or "Iteration"). Sprints are usually confined to time-boxed durations of between 1 and 4 weeks.

The objective of the sprint is to create the ideal conditions so that the project team "sprints" to finish all of the deliverables that are on the Sprint Backlog.

Sprint Planning: Each Sprint commences with two sets of planning sessions - a WHAT and a HOW. In the WHAT session, the Scrum Team reviews the Product Backlog and agrees upon what items from there will be delivered in the current Sprint.

Team members are allowed to freely express their views on whether the commitments can be met, or if they see potential hurdles in delivery. Once the team has agreed upon what is to be included in the current Sprint, they commence discussing HOW to deliver on the WHAT.

The HOW session takes all of the WHAT deliverables and breaks them down into specific tasks, estimates timelines for them, and assigns responsibilities for each task to individuals (or groups) on the project team.

Daily Scrum Meetings: Now that the Scrum Team has received it's "marching orders" for what needs to be delivered by the end of the Sprint (in 1 to 4 weeks' time), the team meets daily to manage its workload and ensure progress is being made.

The Scrum Master facilitates the Daily Scrum Meetings, and participation of all team members is mandatory. These meetings are "stand up" sessions, and should be no more than 15-minutes in duration. If there are significant items flagged during the session, they should not be ignored due to lack of time. "Off line" meetings could be scheduled to deal with such items separately.

During the Daily Scrum Meeting (also called the Daily Stand Up Meeting), the Scrum Master will review the following with each team member: - What was accomplished by him/her since the previous Daily Scrum Meeting?

- What prevented him/her from meeting their goals?
- What do they plan on accomplishing prior to the next session?

The Sprint Backlog is updated as a result of these daily sessions to reflect the current status of the Sprint.

The Sprint Review: Since each Sprint is meant to culminate with the completion of certain deliverables, this implies that the Sprint Review will be held once those deliverables have, in fact, been completed.

During these sessions, held at the end of a Sprint, the Scrum Team demonstrates all of the Sprint Backlog items completed. The Product Owner is responsible for accepting (in line with predetermined acceptance criteria) or rejecting any Sprint Backlog item. Should an item be rejected, the Product

Owner will remove it from the Sprint Backlog and add it (back) to the Product Backlog. Such items are then reviewed and reprioritized for possible inclusion in subsequent Sprints.

The Sprint Review should be kept as informal as possible, with less emphasis on ceremony and more on substance - such as decisions indicating the status of Sprint Backlog items: "accepted", "rejected", "needs some improvement".

The Sprint Retrospective: As its name implies, this is a retrospective look at the recently completed Sprint. The Sprint Retrospective meeting should be conducted immediately following the completion of a Sprint, and must focus on the following three things: - What the Scrum Team did well during the Sprint?

- What didn't go as planned during the Sprint?
- What improvements need to be made so things can go even better in the next Sprint?

The Scrum Master facilitates these sessions, and it is mandatory for the entire Scrum Team to attend and provide feedback. The Scrum Master documents these sessions and updates the Scrum "lessons learned" and best practices documentation for future reference.

Scrum Artifacts

In archaeological terminology, an artifact describes a man made-object, like a vase and tool. Essentially, an artifact is something that we people make to be able to fix an issue or even produce something.

Like any good process or methodology, Scrum prescribes certain tools, called Scrum Artifacts, which help practitioners document the overall project. The Scrum Team and other stakeholders also use these artifacts as visual aids to manage and keep track of the progress being made on the project as well as individual Sprints (Iterations).

Product Backlog: All project teams, regardless of the methodology they follow to manage the project, need some kind of "list of features" to work towards. In its simplest form, the Scrum Product Backlog may be considered as the entire universe of features (User Stories) that must be delivered in order to consider the project completed.

Sprint Backlog: Once specific Product Backlog Items (PBIs) are selected

for inclusion in a Sprint, the challenge is to monitor and track all of the tasks needed to deliver those PBI during the Sprint. The Sprint Backlog is yet another powerful tool that helps Scrum Teams stay on top of what needs to be done to accomplish their goals.

The Sprint Backlog provides a visual picture of:

- The PBI (or User Stories) that are scheduled for delivery during the current Sprint
- The list of open tasks that must be completed in order for each PBI to be considered "done"
- A subset of those "open" tasks that have been assigned to a developer and are actively being worked upon
- A list of development tasks that have been completed ("done") When a team member is assigned a task, the task is removed from the "Open" queue and entered in the "Dev" queue, with his/her name against it.

As and when a task is finished, it is removed from "Dev" and added to the "Done" queue.

Scrum Teams own and manage the Sprint Backlog, and it is updated daily (often several times a day). It is reviewed during the Daily Stand-up meeting, and updates made for everyone to see. Usually, it is helpful for the team to also estimate "time to complete" and "time left to finish" in person-hours so that all team members and stakeholders have a clear indication of work in progress and effort remaining.

Burn Down Chart: Managing projects requires keeping a keen eye on what's happening, in terms of how work is being finished, and what is expected to happen, in terms of forecasting future outcomes. The Burn Down Chart is an excellent tool in the arsenal of the Agile PM to accomplish both those objectives.

This tool helps PMs and other stakeholders see exactly how quickly the Scrum Team is "burning" through the Product Backlog (User Stories). The Burn Down chart: - Is a visual aid that helps show progress made on completing various milestones to deliver the final product - It compares how the team is doing (Real Burn Down), in terms of effort in finishing deliverables - It projects what completion will look like (Estimated Burn Down) if the current rate of progress is maintained - Plots the progress rate (Velocity) of the Scrum Team PMs can use various formats of Burn Down Charts to focus on various metrics, such as "Total Effort versus Work Done"; "Total Effort versus Work

Remaining"; "Effort versus Velocity". These views will help the Scrum Master and Product Owner decide how to manage slippage (if any), or reprioritize PBIs in a more realistic way.

The Burn Down Charts should be reviewed during each Daily Scrum Meeting, and Team Members should be allowed to comment on why things aren't progressing as planned.

Conclusion

Scrum can help teams deliver great products on time if the team members, the Scrum master, and the product owner already have the right skills and abilities to create the product. Scrum is not a magical set of rules that any organization could just follow like a cookbook recipe and expect instant results. What I've given you is a basic understanding on the essentials of Scrum and how to use it to tap into the skills of the team members, Scrum master, and product owner, turning those into powerful leverages in creating innovative products on time.

Scrum is flexible in a sense that after several projects, it can morph into a completely different framework, perhaps with more effective tools, artifacts, and roles. Nevertheless, Scrum has no marked finish line. There is no end goal, that means you can stop learning.

Being good in the implementation of Scrum is never the end goal of companies. In the same way that studying is not the end goal of students, rather, to learn more effectively, learning to be more proficient with Scrum means you'll be able to help your company reach its goal better.

Some methodologies actually have end states, which is why they have certain levels to reach. Scrum, however, does not make an assumption that there is a state wherein you can no longer transcend the current state. It assumes that you will always find new and better ways of achieving goals. After all, this is the real world, where the best does not stay the best for too long.

No Such Thing as a Perfect Start

I've probably mentioned this quite a few times in this book one way or another, but that's because a lot of people still can't quite get this concept right. A lot of people try to implement Scrum, only to delay the start of implementation because they can't seem to perfect the concepts of Scrum. Ironically, this is exactly what Scrum is against: waiting for the perfect moment before you begin.

We live in a non-ideal world. Ideal conditions only exist in abstract mathematical notations that only a few of us would live to see. Scrum allows changes to happen because of the fact that people who try to implement Scrum will inevitably make mistakes before, during, and after the development process. A team's concept of perfection before product development will inevitably be different from its concept of perfection during and after.

If you're worried that you don't have everything perfectly planned, you should stop worrying! Perfection means no longer being able to learn new things, and Scrum emphasizes the need to continuously learn and grow. In most cases, the first few sprints may be somewhat disappointing or even downright ugly, but that's all right. The important thing is that the succeeding sprints start to become better than the previous ones, and in most cases, they really will.

Get started! By starting as soon as you can, you give your company a lot of time to grow.

The first few sprints won't be perfect and neither would the sprints in the distant future, but no Scrum implementation is ever problem-proof. All companies have problems implementing Scrum. Remember that since Scrum helps companies discover hidden kinks and bottlenecks, the companies that find these problems may associate difficulties from the problems to the Scrum framework itself. This misconception is understandable because a lot of methodologies sometimes make work seem a lot easier than it is by hiding potential problems, only to let them pop-up somewhere near the end.

Scrum is a bit more thorough, letting the teams see potential problems ahead in the beginning, middle, and end of the production. The thing is, though, Scrum doesn't tell you how to solve those problems. It can only tell you so much; the Scrum master, product owner, and team member all have to work together to find a solution.

I've said before that one can change some aspects of Scrum should he find more effective solutions, but beware that a lot of people tend to change Scrum, thinking that it'd lead to more efficient operations only to find out that they were slowly reverting back to their old methodologies. If there are dysfunctional people in the organization, the introduction of such a powerful framework that exposes bottlenecks, kinks, and other problems may make them rebellious to the idea of its implementation.

There will be a lot of impediments, especially in problematic organizations, before Scrum could be implemented. It takes patience, consistency, and diligence to properly cement the foundations of Scrum into a weakly managed organization. A lot of people, even the ones with good intentions, will rebel. People naturally resist change, especially ones that force them to change their way of thinking. Help the people involved by easing them into the principles of Scrum and give them a concrete view of the goals you're trying to achieve through this change in methodology and framework. The more people understand Scrum, the less they'll resist its implementation. The less people who resist new implementations, the better your company will be at implementing Scrum.

I don’t claim that this book has all the answers you’ll ever need. Far from that, I encourage you to keep learning and keep asking questions. Keep challenging existing ideologies in a healthy manner. I hope that this book has given you a lot of insights and ideas about the Scrum framework to help you in your journey in creating and delivering great and innovative software. Good luck and may your visions for your company and your teams come true!