
SCRUM-INSTITUTE.ORG
SCRUM PROCESS FLOW
Scrum Team
Scrum Master
Daily Scrum
Product Owner
Shippable Product
Continuous Improvement & Customization
Sprint planning
Sprint backlog
Sprint
Sprint Review
Sprint Retrospective
Product Backlog
Borrowing the idea of scrum from rugby, the agile approach uses a scrum framework to manage effectively the development of products
BORED WITH DRY THEORY?
WE HELP PROFESSIONALS LEARN SCRUM IN PRACTICE SO WELL AND AVOID MISTAKES AS IF THEIR CAREERS DEPEND ON IT!
THE SCRUM FRAMEWORK
INTERNATIONAL SCRUM INSTITUTE ™
THIRD EDITION
© COPYRIGHT INTERNATIONAL SCRUM INSTITUTE ™
WWW.SCRUM-INSTITUTE.ORG

THE SCRUM FRAMEWORK

TRAINING BOOK

THIRD EDITION

BY INTERNATIONAL SCRUM INSTITUTE™

WWW.SCRUM-INSTITUTE.ORG

Dedication 

To all of the International Scrum Institute™ students, thank you for inspiring us, keeping us focused, and making sure we do our best to help you grow in your career with your skills and knowhow.

Without you, your engagement and your loyal support, International Scrum Institute™ could not come where it is today.

TABLE OF CONTENTS CLICKABLE

WELCOME

Hi! My name is Yeliz.

First of all, thank you very much for getting your copy of The Scrum Framework. I love that you are taking the time to read it.

I want to briefly share with you the backstory of why we wanted to write this book for you and how you can get the best use out of it.

Within the context of our Scrum training and Scrum certification programs, we did thorough research in the Scrum education space.

The conclusion was: We failed to find one single reliable study book, we could sincerely recommend to our students!

We surveyed and talked to our successful students who have successfully passed their Scrum certification exams, and we found out a remarkable and yet indisputable piece of information.

Almost none of the Scrum books in the market did help them learn Scrum and make a smooth beginning to deploy and profit with the Scrum Framework. They did end up with literally zero return on investment. Both for their professional objectives as individuals and the financial goals of their organizations.

A significant number of Scrum books in the marketplace claim that they cover all details of the Scrum process. However, what they are not telling is that: They don't have a logical, to-the-point, and digestible structure, and time-tested and proven contents.

So these books were unable to help our students comprehend and most importantly love Scrum!

In summary, to remove this significant impediment in the Scrum learning space, we took the liability to write for you The Scrum Framework and brought it to your service!

We are absolutely confident that **The Scrum Framework** will make you proficient in the Scrum process and its practical use in your career and businesses.

So you will have an unprecedented opportunity to love Scrum and keep on taking the tangible benefits of being a Scrum professional who knows how Scrum should work.

Take some coffee to enjoy and some paper to take your notes, and spend some quiet time to read The Scrum Framework!

Afterward, you will have a great understanding of the Scrum domain and be prepared to pass your Scrum certification exams.

You will be ready to deliver great products and services to your clients and employers and to build your bright career and future!

It already seems to me that you're a person who is keen on adding new skills to your toolbox. Otherwise, you wouldn't be reading these sentences today.

I am delighted that you're giving us your time and attention to learn Scrum. Let me assure you that we'll never take this responsibility lightly. It's our duty, obligation, and at the same time, our pleasure to accompany you on your journey to learn Scrum.

You can count on me whenever you may need any help. I will be always pleased to assist and serve you!

Thank you very much again for your trust in our services and engaging with **The Scrum Framework** today!

Yeliz Obergfell
Vice President - Student Experience
International Scrum Institute™
https://www.scrum-institute.org

SCRUM-INSTITUTE.ORG

"It has been a fantastic experience with getting this certification. The Scrum Institute provides you all the information and tips to get Scrum knowledge successfully.

Watching Scrum Institute in action is a valuable learning experience by itself because most of the champion companies in their domains apply these types of professional services that drive benefits to their clients.

And Scrum Institute is the best and only education provider which continuously cares about its students. They go the extra mile to teach us and prepare us the life with Scrum process rather than repeating dry theories and boring dogmas!"

Francisco Gonzalez, IBM

At International Scrum Institute, we are calling this an Achievement!
And we are proud of being part of it!

Register Your Scrum Certification Program
https://bit.ly/2LNv7xW

ABOUT INTERNATIONAL SCRUM INSTITUTE™

International Scrum Institute™ is an independent institute. We help organizations and professionals get certified with worldwide renowned and valid Scrum certification programs and prove their competence in the Scrum domain. We empower professionals globally to build their careers, and organizations to create and sell their outstanding products and services that their clients will love.

Your renowned Scrum certification programs have proven their worldwide recognition by being the choice of more than 594,000 Scrum professionals in 143 countries.

The term "Scrum" was first used and published by Harvard Business Review in January 1986. Hirotaka Takeuchi and Ikujiro Nonaka coined the term "Scrum" with their article: **The New New Product Development Game**. So, the Scrum process was initially meant to be an open project management framework.

Nonetheless, things didn't work like that. The Scrum process has been heavily commercialized. Worst of all, Scrum has been subtly dogmatized, so it has started contradicting its very own "inspect and adapt" spirit which you will later learn more about it in this book. Professionals and organizations have wasted enormous amounts of training, certification, and then recertification fees for literally zero return on investment.

Before International Scrum Institute™ was established for you, there used to be pressing challenges for Scrum professionals like yourself.

You didn't possess a reasonable alternative to get your Scrum certifications and prove your competence in the Scrum domain. Scrum professionals had to pay expensive fees for the one way profit-driven Scrum certification programs of other certification entities. Moreover, they had to pay hefty prices for classroom training, recurring

certification renewals, and various additional recurring subscriptions and memberships.

International Scrum Institute™ aims to remove these barriers set in front of the Scrum professionals in developed and emerging markets. We are here to save you from paying unreasonable fees for Scrum classroom training and Scrum certification programs before you certify your knowhow in Scrum.

International Scrum Institute™ provides ten major online Scrum certification programs. These programs have been designed by our consortium of renowned business and people leaders, coaches, mentors, experts, and authorities from all major industries.

Here is an overview of our Scrum certification programs we have created for you:

- **Scrum Master Accredited Certification™**
- **Scrum Product Owner Accredited Certification™**
- **Scaled Scrum Expert Accredited Certification™**
- **Agile Scrum Leadership (Executive) Accredited Certification™**
- **Scrum Trainer Accredited Certification™**
- **Scrum Coach Accredited Certification™**
- **Scrum Team Member Accredited Certification™**
- **Scrum Certification for Web Developer™**
- **Scrum Certification for Mobile App Developer™**
- **Scrum Certification for Java Developer™**

Moreover, feel free to check out the articles specified below to read why we perform and serve you far better than our competitors.

- **Featured on LinkedIn with Hundreds of Likes: Scrum Master Certification Made Economical: Step-by-Step Plan**
- **8 Reasons Why International Scrum Institute™ Serves You Far More Better Than Its Competitors!**

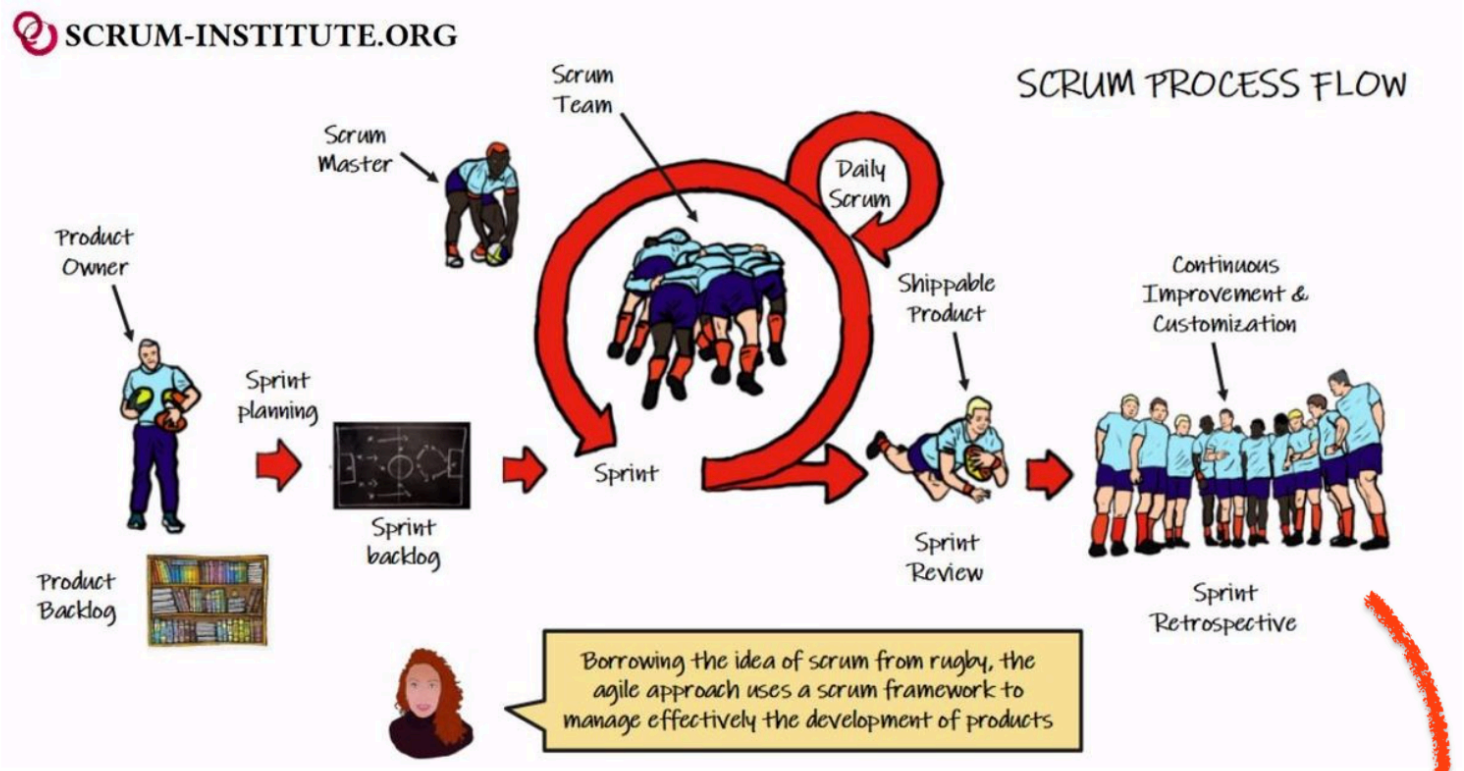


Image Credits: Our Good Friend Eric Lefebvre, Switzerland (Author of Understanding Insurance)

Scrum Master Certification Made Economical: Step-by-Step Plan

Like Comment Share 252 · 54 Comments

Scrum Master Certification Made Economical: Step-by-Step Plan Featured on LinkedIn https://bit.ly/31R5wd2

WHAT IS SCRUM?

What is Scrum? Well, without making things too complicated, the Scrum framework can be defined as the following:

Scrum is an iterative software engineering process to develop and deliver software.

Although the software is the main focus of the Scrum framework, iterative and agile Scrum process can be and is already being applied outside the software industry as well.

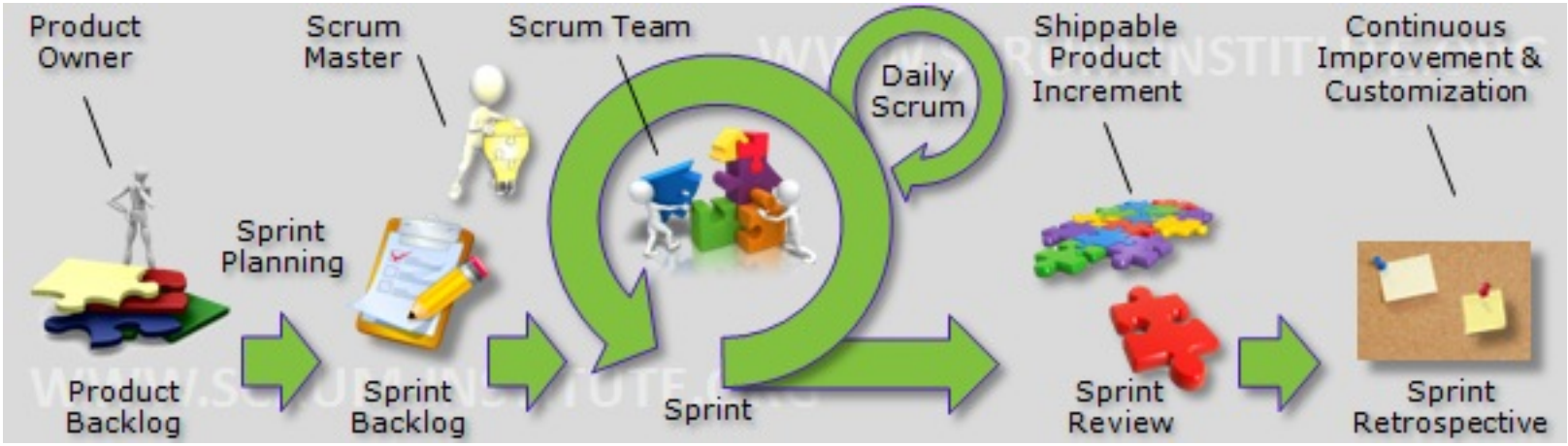
Most people in the IT industry believe that the term "Scrum" was coined early in the 2000s as a parallel track of emerging agile software development and delivery trends. That is a piece of incorrect information!

The term "Scrum" was first used and published by Harvard Business Review in January 1986. Hirotaka Takeuchi and Ikujiro Nonaka coined the term "Scrum" with their article: **The New New Product Development Game.** (Yes, two News) You should have a look at "The New New Product Development Game" to see how everything all about Scrum got started!

Scrum can be used in all kinds of software development projects. To develop and deliver complete software packages or only some modules of larger systems — both for products and services of internal and external clients.

The Scrum Framework is a lightweight process. It focuses on increasing the productivity of teams while reducing wastes and redundant activities.

Scrum defines some general guidelines with a few rules, roles, artifacts, and events. Nevertheless, all of these components are critical, serve for specific purposes, and they are essential for the successful use of the Scrum framework.



Overview of the Scrum Framework

The main components of Scrum framework are:

- **Three Scrum Roles:** The Scrum Product Owner, the Scrum Team, and the Scrum Master.
- **Five Scrum Events (Scrum Rituals) or Ceremonies:** Scrum Grooming (Backlog Refinement) Meeting, Sprint Planning Meeting, Daily Scrum Meeting, Sprint Review Meeting, and Sprint Retrospective Meeting.
- **Product Backlog (Scrum Backlog) or Scrum Product Backlog:** An artifact that is used to manage and prioritize all of the known requirements of a Scrum project.
- **Sprints:** Cycles of work activities to develop shippable software product or service increments.
- **Sprint Backlog:** An artifact to keep track of requirements committed by the Scrum teams for a given Sprint.

Self-organization and unconditional collaboration are critical elements of the Scrum framework. Scrum Teams do no longer require a project manager in a classical sense. With the Scrum framework, the Scrum Master and the Scrum Product Owner share the role and responsibilities of a typical project manager.

Nonetheless, a Scrum Master or a Scrum Product is never allowed to overrule the democratic decision-making capability of a Scrum Team. For instance, only the Scrum team members can jointly commit which ones of highly prioritized Backlog items they will deliver in a Sprint as a software increment.

Another central element with the Scrum framework is the continuous improvement that we enable with "inspect & adapt". A Scrum Team continuously monitors, inspects, and assesses their artifacts and their use of Scrum framework to adapt and optimize them. These continuous efforts for optimization maximize quality, efficiency, client satisfaction, and therefore minimize wastes and overall project risks.

The Scrum framework understands that the requirements are likely to change and they are not entirely known, especially at the beginning of projects.

Every project has unknown unknowns. Sometimes a few, sometimes a lot. The Scrum framework helps us embrace that we can discover and deal with these unknown unknowns only while we are running our projects.

The Scrum Team first fine-tunes and granularizes the lower-level or low priority requirements before it implements them. During Scrum Grooming (Backlog Refinement) and Sprint Planning Meetings. Openness for change, continuous optimization, and learning from errors are now becoming integral elements of the whole software engineering lifecycle.

Another cornerstone of the Scrum framework is transparency and direct communication. The Scrum Product Owner works closely with the Scrum Team to identify and prioritize requirements. These requirements are written down as user stories and stored in the Scrum Product

Backlog. The Scrum Product Backlog consists of all tasks that need to be implemented to deliver a working software system successfully.

A Scrum Team is empowered to select the user stories with which they are confident to deliver within the 2-4 weeks of Sprints. Because the Scrum Team commits its own goals, the team members feel more engaged, and they know that their opinions are listened to. This inclusion of Scrum team members to the natural flow and planning of software projects increases the team morale and subsequently augments the team performance.

Scrum Masters possess another vital role in the Scrum Framework as they work as servant leaders for and with their Scrum Teams.

Scrum Masters are trained facilitators to ensure flawless operation of their Scrum Teams. Sometimes they are master negotiators to protect their Scrum Teams from interruptions and fictive priorities of their stakeholders. Other times they are master communicators to remove or prevent known or anticipated impediments before these impediments bring their teams to dead-end streets. To only call a few of the responsibilities of Scrum Masters. We will cover more about the duties of various Scrum roles later.

The Scrum Framework, in its pure form, is best suitable for highly independent, one-team greenfield or brownfield projects.

However, the practical common sense of Scrum professionals did not stop there. With the introduction of additional roles and addendums such as "Chief Scrum Product Owner" and "Scaled Scrum", it can be used within different project configurations too, including multi-team and geographically distributed project setups. We will cover more about these as well.

For now stay tuned and keep on enjoying the lecture!

AGILE MANIFESTO

When the IT industry talks about the Scrum framework, It's also often we hear the term "Agile Scrum" along the same lines as "Scrum". It led some of us in the industry to think and look for differences between the terms "Agile Scrum" and "Scrum".

Here is good news for you. "Agile Scrum" and "Scrum" terms do both refer to the same thing. They both refer to the Scrum software engineering process. Then why do we sometimes use the word "Agile" in front "Scrum"?

It's because the scrum framework fully embraced and embedded the Agile Manifesto (Manifesto for Agile Software Development) to its core process, principles, and underlying philosophy. That brings us to understand the agile manifesto and the values of the scrum process better before we deep-dive the technicalities of the scrum process.

Agile manifesto values:

- Individuals and interactions over processes and tools,
- Working software over comprehensive documentation,
- Customer collaboration over contract negotiation,
- Responding to change over following a plan.

While the factors on the right-hand side do still possess significant values, the agile manifesto appreciates and prioritizes the factors on the left-hand side higher.

The elements favored by the agile manifesto have been carefully time-tested and chosen to:

- Serve clients and stakeholders better and create value for them with software,
- Enhance the profession of software engineering regardless of your role, title, and career level.

SELF-ORGANIZATION

The scrum team organizes itself. Scrum team members decide in consensus about tasks they need to execute to deliver the goals of a sprint. A self-organized team doesn't require a manager or a team leader.

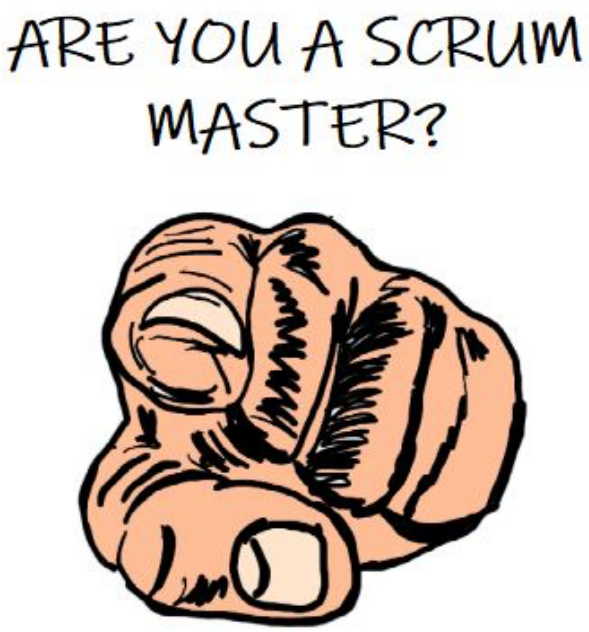
Self-organization in the scrum framework is very disciplined.

Sprint Backlog, Sprint Burndown Chart, and Daily Scrum Meetings which you are going to learn more about them later in this material build the foundation of self-organization.

Organizing the work by themselves requires for the most teams a learning phase. Competent scrum masters who own scrum master certifications support their scrum teams to excel with self-organization quickly.

Self-organization also includes the ability to work together despite different opinions and possible conflicts among various scrum team members. Self-organization requires compliance and trust in joint decision-making processes.

Those decision-making process in the scrum framework includes, but not limited to, planning, estimating, implementing, reporting, and reviewing the work the scrum team is jointly responsible.



Yes? Then you need to bring up a team that can self-organize its own work!

INSPECT AND ADAPT

Scrum Inspect and Adapt is a straightforward concept to comprehend, but the hardest to properly implement and master.

Companies have finally confirmed that none of their project managers can fully foresee the big picture of complex systems. They were unable to do reliable end-to-end planning. It was evident for them that they needed to try something different.

Together with lean manufacturing (also known as lean movement), companies needed to develop a process to empower them strategically. They needed a standard operating procedure to help them learn and fix their courses of action while they're running their projects and even operations.

That was the birth of Toyota Improvement Kata, which we today call "Inspect and Adapt" while we talk about scrum software development and delivery framework.

According to "Scrum Inspect and Adapt":

- **Step 1. Inspect:** We do our best to grasp the current status of the project with our current level of knowhow and understanding about it.
- **Step 2. Adapt:** We define a direction and vision about the next steps of our project and then strategize and execute our vision.
- **Step 3. Learn:** We carefully observe, learn, and teach each other while we do so. We continuously log what works and what doesn't work while we do our work.
- **Step 4. Restart:** Start over from Step 1 again.

Note that those four steps described above are analog, but not limited to the following Scrum rituals (Scrum events).

- **Step 1. Inspect** is analog to Sprint Review Meetings and Sprint Retrospective Meetings.
- **Step 2. Adapt** is analog to Sprint Planning Meetings and Backlog Refinement Meetings.
- **Step 3. Learn** is analog to Daily Scrum Meetings.
- **Step 4. Restart** is analog to the closure of a sprint and the start of a new sprint.

FIVE KEY VALUES OF THE SCRUM FRAMEWORK

We have already mentioned that the scrum framework is not only a software engineering process. It also has a robust set of underlying principles.

In fact, most of the professional business domains can apply and utilize these principles.

It’s not enough to get a scrum certification to be hugely successful with the scrum. You should possess a firm grasp for scrum values to succeed with the Scrum framework.

So that you’re going to deliver a great job and fantastic software that your customers and employers love. Let me now tell you more about those principles of the scrum process.

Scrum Value #1. Courage

There are times when doing the correct thing to serve the best values and benefits for our clients are not the easiest. In such moments, scrum master, scrum product owner, and the scrum team members should remember their duty and obligation.

That's to build the best possible products and services in their particular business and information technology domain. To be better than mediocre, **a scrum team should sooner or later face difficult decisions that won't make everyone happy in their particular ecosystem of stakeholders.**

To deal with this, all members of the scrum team should remember what they learned during their scrum certification training.

They should remember to be courageous, and they should master to decide and act courageously.

Scrum Value #2. Focus

With the scrum framework, when you hear the value focus, you should be thinking about two things:

- **Identification of correct work:** What tasks are necessary to deliver the goals of my sprint? What are essential to developing the best software products and services for my clients so that they will be pleased with my work?
- **Prioritization:** What tasks should I be working on next?

Each moment in time, there is one critical question that the entire scrum team, including scrum master and product owner, must be answering.

This question is: "What are the most important things we should be doing at the moment to fulfill reasons of why an employer hired us in the first place?"

Scrum framework has several built-in events (rituals) to ensure the reasonable prioritization of user stories and tasks. According to the scrum process, the prioritization of user stories and their associated tasks should have a continuous priority.

So we make sure that the scrum team works on the right things in the correct order.

Some of the built-in scrum ceremonies (scrum events) to prioritize our work and adjust our focus are:

- **Scrum Grooming (Backlog Refinement) Meeting:** Grooming Meeting solely focuses on prioritization for Product Backlog to prepare it before the upcoming Sprint Planning Meeting.
- **Sprint Planning Meeting:** These meetings help us see the dependencies and correct order of work to deliver our user stories.
- **Daily Scrum Meeting:** Daily Scrum (Daily Stand-Up) Meeting supports us to set the tone of an upcoming workday. We must direct our focus on where it's most required.
- **Sprint Review Meeting:** Sprint review meeting indirectly shows us where the emphasis of the

scrum team must be channeling to have more successful reviews in the future.

- **Sprint Retrospective Meeting:** These meetings support the scrum team to prioritize what aspects of their engineering process must be first improved.

Here in this section, I covered scrum rituals only from a focus point of view. You can find a more detailed explanation about the scrum ceremonies later in this material.

Having read all these, it must be evident for you now how essential prioritization and focus for the scrum framework are.

Scrum Value #3. Commitment

Without the commitment of scrum master, scrum product owner, and the scrum team, there is no possibility to deliver outstanding results with software.

In the world of the scrum software development process, most people translate the commitment value as the agreement and confinement of goals of given sprint deliverables.

Although this entirely makes sense, that understanding is not flawless. **Whenever you hear the word "commitment" within the context of scrum values; what you should remember is the word: "obsession".**

To be successful in software engineering and, in life and business, you should become obsessed with your goals. So in the context of the scrum process, **you should become obsessed with creating marvelous software for your clients to solve their problems.**

Why are commitment and the associated obsession with scrum goals so important? Because without the obsession with the team's mission to build and deliver astonishing software, each time the scrum team encounters a non-trivial impediment, your work will slow down and stall.

Then the scrum master and the scrum team will start creating explanations to justify and legitimize for scrum product owner why they're unable to deliver sprint goals. Excuses should have no more room in your team if your goal is to become a better than an average scrum team.

Only with an enormously high level of dedication, it's relatively more comfortable and fulfilling to solve the problems of our clients and help and build value for them with software.

Scrum Value #4. Respect

Regardless of their age, gender, race, belief, experience, competence, opinions, and work performance, every member of a scrum team must respect and count on each other.

This respect is not only confined within the boundary of the scrum team. Moreover, every internal or external IT and business stakeholder who interacts with the scrum team is utterly respected and welcomed by a scrum team.

Experienced team members must pay attention in order not to invalidate the willingness of the contribution from less experienced team members.

It's particularly crucial to properly receive and answer opposite opinions that the majority of the group do not agree with.

Scrum Value #5. Openness

The scrum value "openness" is often one of the primary differentiators between an average and high-performer scrum team. It would help if you resembled the openness capability of a scrum team to the vast ability of a collection of openminded individuals.

They're creative, innovative, intellectual, honest, direct, and humble. In the scrum software engineering and delivery process, there is no inappropriate opinion, decision, and action.

The only condition is that they must be transparent, and they should aim to contribute to the joint mission of the scrum team.

It doesn't mean that every decision and action must necessarily accelerate the outputs of the scrum team, and they should result in substantial success stories.

Thanks to openness and courage values, the scrum software development group is not afraid of making mistakes. **They see their errors and less than optimal outcomes as vital chances to meaningfully improve their overall productivity and quality of work.**

Courage, Focus, Commitment, Respect, Openness are the vital Scrum Values you always keep in mind.

INTRODUCTION TO SCRUM - A REAL WORLD EXAMPLE (CASE STUDY)

Before Starting The First Sprint

Alex works as the Scrum Product Owner of a new software development project. One of his first tasks is to assess and find out requirements **to deliver business value his client is looking for**.

He needs to make sure that his client will get **the correct software to achieve tangible business results**. He writes down the essential use cases and discusses them with the architects, client representatives, and other stakeholders from IT and business units.

After assembling the high-level use-cases and requirements, he writes them into the **Scrum Product Backlog** and initiates an estimation and prioritization session with the Scrum Team. As a result of this session, all items in the Scrum Product Backlog get an **initial rough estimate and priority**.

During those sessions, Anna, the Scrum Master, ensures that everyone speaks the same language. So, the Scrum Product Owner, the Scrum Team Members, and their stakeholders are **aligned with the anticipated goals**. So they have an **adequate understanding of potentially new concepts** for them, such as Use Case, Backlog, Sprint, and so on. And most importantly, **the Scrum software development and delivery process is correctly applied in the store**.

Now Alex, the Scrum Product Owner, begins to break down the high-level requirements into the first draft of smaller-grained user stories. With this list, he then calls for the first Sprint Planning Meeting.

Sprint 1 - Day 0

During the **Sprint Planning Meeting**, Alex presents the Scrum Product Backlog items from the highest priority to the lowest. The Scrum Team asks and clarifies open questions. For each item, the team discusses if they have enough

capacity and the required know-how to develop and deliver it. The Scrum Team needs to ensure that **all required human and technical resources are in place before the start of the Sprint.** They need to confirm that all prerequisites and dependencies are fulfilled, which could be critical to delivering certain software features successfully.

During **Sprint Planning Meeting (What-Part)**, the Scrum Team commit to complete the user stories 1,2,3,6,7 and 8 until the end of the Sprint. So these user stories are now moved from the Scrum Product Backlog to the **Sprint Backlog**. The user stories 4 and 5 cannot be accomplished in this Sprint, as some prerequisite technical infrastructure is not yet in place.

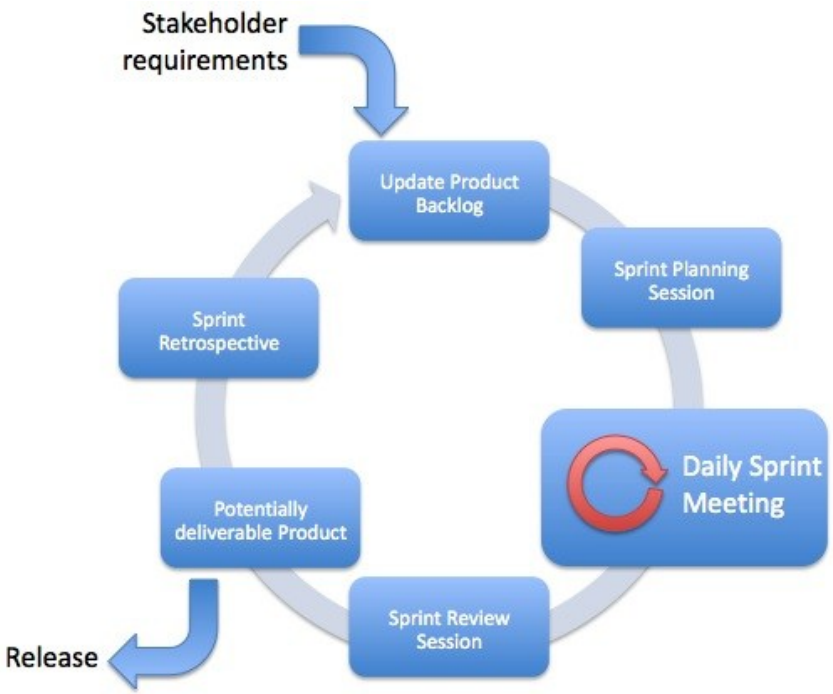
After the What-Part of the Sprint Planning Meeting, Anna, the Scrum Master, calls the Scrum Team to **drill down how the team is going to implement the committed user stories (How-Part)**. The emerging tasks during the How-Part of the Sprint Planning Meeting are written down on the cards, and the team store them into the Sprint Backlog. Now all members of the Scrum Team are ready to select a task to begin to work on.

Sprint 1 - Day 1

In the morning, the whole team gets together for their **Daily Scrum Meeting**. Everyone gives a **brief and concise statement** about what he or she has done so far, updates the estimates of remaining work on the cards of the Sprint Backlog. Everyone tells what he or she is planning to do today, and reveals if there are any impediments which hinder them from processing any tasks.

Today one of the Scrum Team members, Melinda, informs the Scrum Team that she has a problem with the license of the integrated software development environment she is using. Anna, the Scrum Master, checks if other team members have the same problem and confirms that she'll take care of this impediment after the meeting. After about **15 minutes** of this Daily Scrum Meeting, everyone goes back to work.

After this meeting, Anna updates the **Sprint Burndown Chart** to visualize the progress of work during this Sprint. Then she calls the software vendor, orders the missing license, and delivers it to Melinda.



Introduction to Scrum
A Real World Example (Case Study) across various Scrum Phases and Sprints

Sprint 1 - Day 2

In the morning, the whole team gets together again for their **Daily Scrum Meeting**. In the afternoon, a member of the Scrum Team, James, has uncertainty about the expected outcome of one of the user stories. He calls Alex, Scrum Product Owner, and they discuss this user story to ensure that James properly understands it. After Alex gets informed and confident about how to proceed with this user story, he continues working on its implementation.

Sprint 1- Day 6

The days starts again with the Daily Scrum Meeting of the team. Anna, the Scrum Master, notices this morning that the meeting tends to take more than 15 minutes. The Scrum Team members are engaging with a discussion regarding the optimization of some database queries. Anna reminds the team that the **Daily Scrum Meetings are not meant to do the work, but formally aligning the team about**

the work and bringing them on the same page.

After the Daily Scrum Meeting, Alex (Product Owner) informs Anna (Scrum Master) that the client brought up several new requirements that may potentially impact the ongoing Sprint and the subsequent Sprints. Anna politely reminds Alex that **the Scrum Team is unable to pick up these requirements during the current Sprint as the team has already committed to the scope (user stories) of this Sprint**. And yet, Anna calls a **Backlog Refinement Meeting** for the afternoon so that Alex can inform the team about these new requirements.

During this meeting, the group supports Alex **to figure out where these user stories fit the overall development plan of the software, their initial task break-down, estimates, and priorities**.

Sprint 1 - Day 10

Finally, that's the last day of this first Sprint. Anna, the Scrum Master, invites the Scrum Team for the **Sprint Review Meeting**. The team has prepared a non-production server with the latest version of the shippable software increment they created.

Alex, the Scrum Product Owner, and Mr. Rich, one of the client stakeholders, sit in front of an instance of a graphical user interface of this software. They validate **if the implementation meets the expectations** and if the team documented details regarding the current level of application adequately.

At the end of the Sprint Review Meeting, Alex concludes:

- The team delivered user stories 1,2,6 and 7 as committed and expected.
- The team couldn't finish the user story 3 on time, and they didn't demonstrate this user story at all. So, the remaining tasks of this user story are shifted to this next Sprint.

- The user story 8 did not fulfill some of its Definition of Done (DoD) criteria. This user story is moved to the next Sprint, so the team can define and complete the associated tasks to satisfy the DoD of this user story later.

Alex, the Scrum Product Owner, and Mr. Rich, the client stakeholder, **shortly debrief the Scrum Team** about the upcoming changes and challenges about the software requirements and **the direction of the overall strategy about this software should be going**. Mr. Rich thanks the Scrum Team for their efforts and commitment and leaves the room.

After the completion of the Sprint Planning Meeting, the Scrum Team sits together for the **Sprint Retrospective Meeting**. During this meeting, they discuss what went well during the Sprint and what could be improved, so that the likelihood of failed commitments like it happened with user stories 3 and 8 will reduce in the next Sprints. One of the hurdles identified from the Sprint Retrospective Meeting is that the team do not know enough about the overall system architecture. Anna, the Scrum Master, takes over the task of bringing a system architect on board to coach and guide the team at the beginning of the next Sprint.

Sprint 2 - Day 1

Alex, the Scrum Product Owner, keeps on adding new requirements to the Scrum Product Backlog based on his recent client meetings. Moreover, he improves the way he articulated DoD of user story 8, so the Scrum Team can better envision the expected outcome from this user story.

Alex then invites the team for the Sprint Planning Meeting for Sprint 2. The Scrum Team discuss and commit to user stories with the guidance of Anna, the Scrum Master, and subsequently, the second Sprint begins.

THREE ELEMENTS OF CHAOS AND FRUSTRATION BEFORE THE SCRUM FRAMEWORK

To better understand the impact of the scrum framework to our software engineering practices and businesses, it makes sense to have a look at a day in the life (or a software project in life).

Therefore, I would love to briefly talk about a software project from the past before we adopted the scrum development and software delivery framework in our organizations.

A few days before I wrote these lines, we had lunch with one of my ex-colleagues with whom we used to work together almost 20 years ago.

This gentleman, Marcus has got his **scrum master certification** and **scrum product owner certification** from **International Scrum Institute™**. He currently works as a scrum master for one of the leading software houses in the agile project management software domain.

As a scrum master, Marcus is now in charge of operating an agile scrum team whose scrum team members located in geographically distributed locations around the globe.

During our lunch, Marcus admitted that there are a lot of typical challenges with distributed agile scrum teams. Some of the problems he specifically mentioned related to his software project configuration are:

- Differences in working styles among scrum team members,
- Timezone differences,
- Cultural misfits, and
- Language constraints.

Despite these difficulties, Marcus still added that running a software project with the agile scrum process is more fun, productive, and enriching than how we used to work 20 years ago. Compared to days when we used to work without scrum software development and scrum software delivery processes.

Marcus' statement was indeed a big testimonial for the credit of the scrum framework from a very accomplished and experienced manager, scrum master, and product owner.

Thank you, Marcus!

Then we explained to him one of our past software projects before we used to meet with the scrum framework. I'm sure that many scrum masters would resemble this experience to their previous projects before they've gotten their scrum master certifications.

Back in the late 1990s, we were part of a software engineering group to build a smart card-based **public key infrastructure**. Smart cards securely protected private keys of infrastructure members, associated public keys and their wrapper certifications were openly distributed (as the name public implies).

Back in the day, this was by itself a relatively complex IT project that required multiple interdependent hardware engineering and software engineering teams. We had to do massive amounts of **research and development (R&D)** to build a fully functional hardware and software system.

Remember these are days before we had the **minimum viable product (MVP)** concept to experiment, create, learn, and experiment again.

Without scrum to create such a sophisticated infrastructure that constituted numerous hardware and software elements was a real challenge.

Here are **three significant setbacks** we used to have without any scrum masters and anyone who possesses a scrum master certification in our teams.

FRUSTRATION #1. WE HAD TO PLAN OUR ENTIRE PROJECT BEFORE WE UNDERSTOOD WHAT THE PROJECT WAS ALL ABOUT

Without scrum, our teams had built and delivered entirely wrong software and hardware products that did not fulfill demands from our client.

We had times in our professional lives when some third party companies had imposed how we supposed to build our software products or software services.

Capability Maturity Model (CMM), **ISO 9001:2008** and other derivates attempted to help our companies to ensure we build our correct software in correct ways.

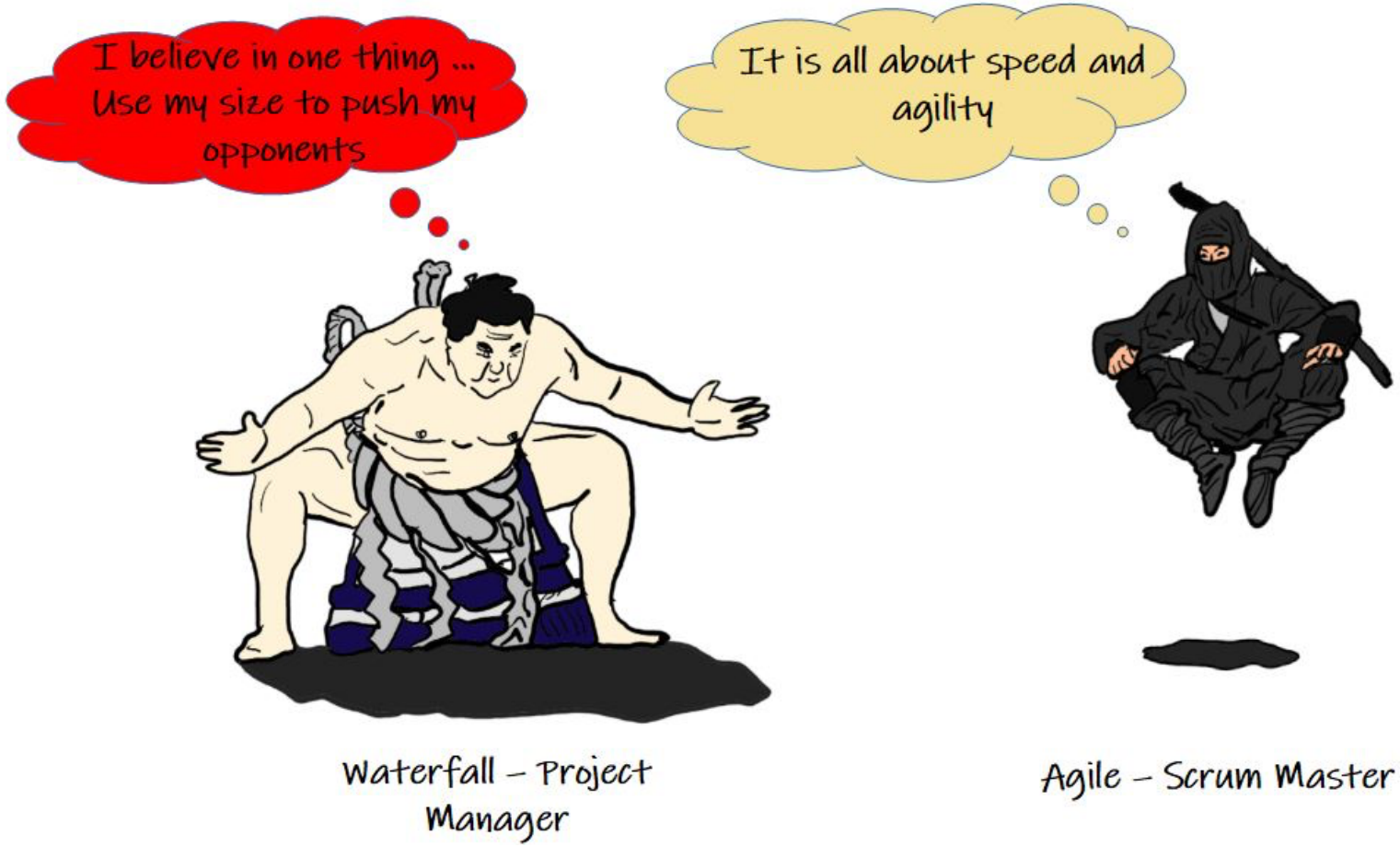
How successful they used to be is not part of this book. This book was meant to focus on the scrum process and merits of the scrum framework rather than criticizing almost extinct procedures.

However, I have to add that these process improvement frameworks before the scrum software engineering framework recommended a phased approach. They advised a phased software engineering approach which we called the **Waterfall Software Engineering Model**.

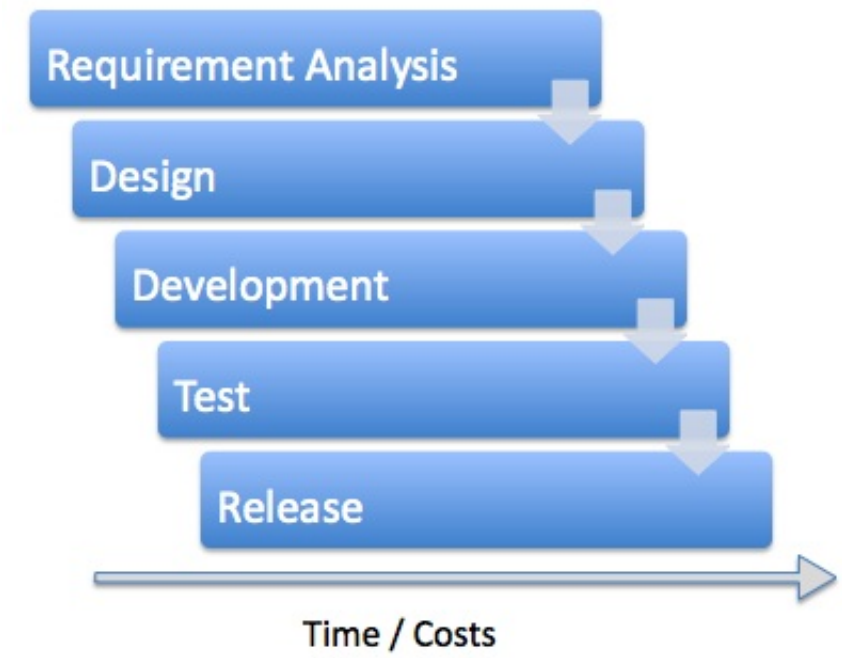
With the Waterfall Model, each software project was supposed to start with requirement analysis, where we aimed to understand what our client needed and wanted.

Then we designed these requirements, we implemented them, we tested (verified) them, and we maintained them in our software production environments. Finally, we reached to end of the software engineering lifecycle.

Nonetheless, the reality didn't play out like that!



The Waterfall Methodology vs The Scrum Framework



Phases in the Classical Waterfall Software Development Model

The adverse effects of unforeseen delays happened during a particular phase of the Waterfall Software Engineering Model were inevitable to the following software engineering phases.

Studies have shown that in over 80% of the investigated and failed software projects, the usage of the Waterfall Methodology was one of the critical factors of failure. **But why?**

As shown on the left side, when deploying the Waterfall Methodology, there is a strict sequential chain of the different project phases. A previous phase has to be completed before starting the next phase. Going back is, in most cases, painful, costly, frustrating to the team, and time-consuming.

The project timeline is planned at the start. A releasable product is delivered only at the end of the project timeline. If one phase is delayed, all other phases are delayed too.

To avoid this, project managers of the Waterfall Methodology usually try to anticipate all possibilities beforehand. That means that in one of the earliest phases of the project, they try to define all requirements as fine-grained and complete as possible. However, requirement definition in an initial stage of a project is often complicated, and therefore many requirements

change (or should change) throughout the project.

Studies have shown that in more extensive and complex projects, about 60% of the initial requirements do change throughout the course of projects. Other requirements are implemented as defined, but some of them are not really needed by the customer. So those implementations consume time and money that could have been better used to implement functionality with a higher added value for its clients.

The separation into different project phases forces project managers to estimate each phase separately. The problem is that most of these phases usually are not separate. They are working together and in parallel.

For instance, no reasonable human-being can assume that the development phase finished before the testing phase started. And yet, this is precisely and unfortunately how the Waterfall Methodology used to work.

The Waterfall Methodology for developing software can be used for implementing small and straightforward projects. **But for bigger and more complex projects, this approach is highly risky, if not insane.** It's often costlier and always less efficient than Scrum software development and delivery framework.

This was the life before the Scrum framework.
Sending our software back and forth between various teams, without the guidance of professionals with the Scrum skills, **made our work bureaucratic, complex and unproductive**.

Finally, it wasn't only the product which suffered, but also employee morale and commitment to our organizational mission have wholly disrupted.

FRUSTRATION #2. LACK OF COMMITMENT, CHANGE MANAGEMENT, AND WORKING TOGETHER DISCIPLES AMONG DIFFERENT TEAMS

The most significant weakness of process improvement frameworks used before Scrum was that: **They mainly focused on self-serving organizational demands of leadership.**

Some of these demands are monitoring, compliance, and predictability. **There was no focus on serving clients well and increasing employee morale at all.**

Thus members of software management teams and various other internal and external stakeholders attempted to have a fixed deadline for software delivery projects and easily monitor the progress of software engineering phases.

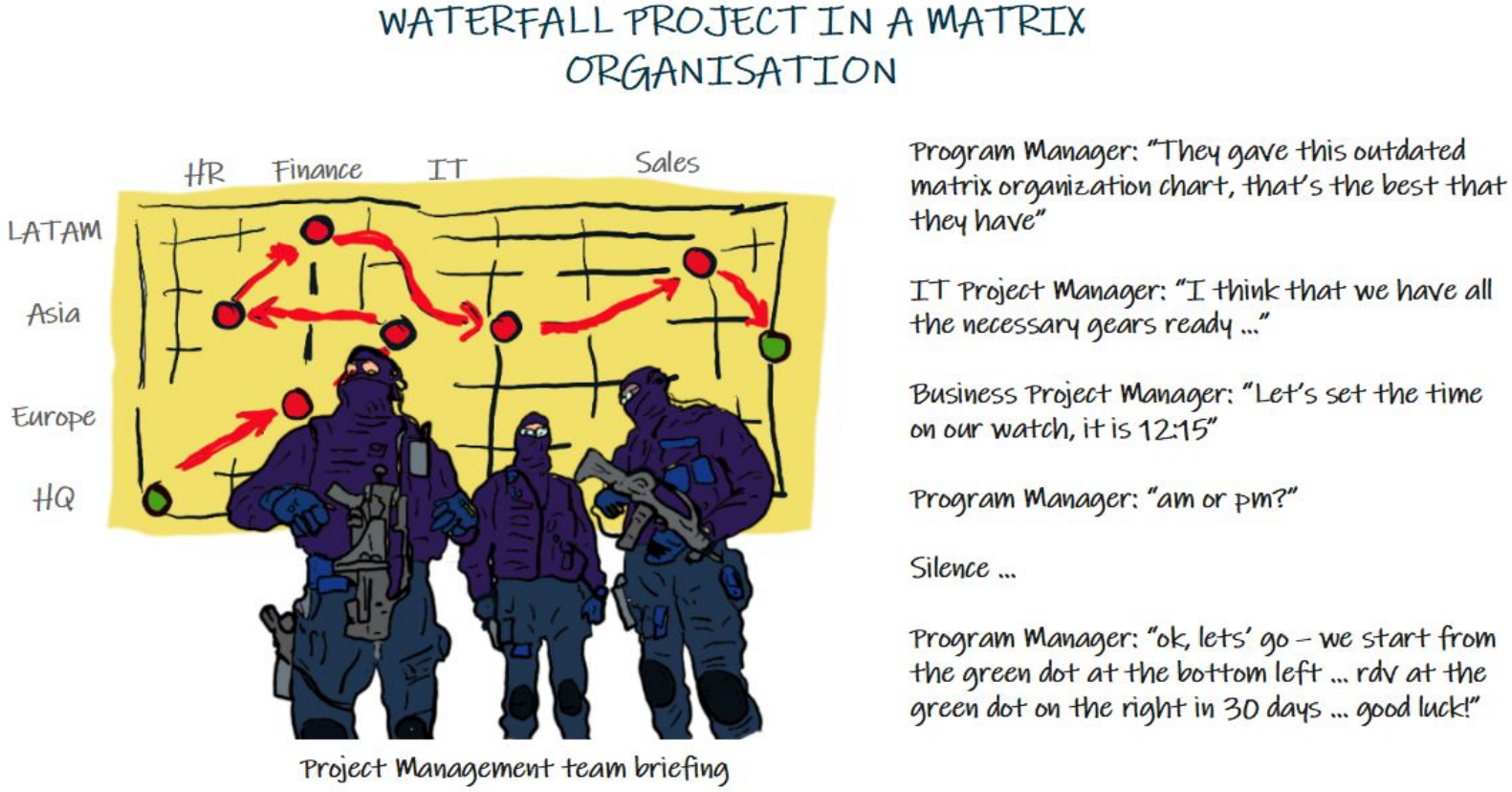
They penalized their people if something was outside the planned track, and they hoped to fix emerging issues before the scheduled date of project completion.

Furthermore, independent silos realized entirely separated software engineering phases. As an example, the development team was completely independent of a software testing (verification) team. Most people who supposed to work for the same business mission didn't even know each other by their names.

Have you got a guess about the reason for this silo-mentality in our organizations rather than focus on business missions and professional (business) maturity of employees?

The reason is simply **the matrix management**. Matrix management is an organizational management and employee structure, and it has been in our businesses since the 1970s. At first glance, the differentiating idea behind a matrix organization or matrix management seems to be smart.

The Leadership creates an organizational structure by bringing together employees with similar kinds of functional and technical



The Waterfall Project Delivery Model in a Matrix Organization

skillsets into the same or at best neighbor silos.

Back in times, it was quite popular to see the so-called "Center of Competences" in our companies where each center of competence represented an independent and autonomous silo.

One silo for C++ developers, another silo for database administrators, and another entirely separate quality assurance silo in oversees and it goes on and on. Go and figure!

The biggest challenge with the matrix organizational structure was that: To deliver a software project without the scrum framework and scrum masters, project managers had to borrow employees from silos temporarily.

These employees did not even physically position with their project teams, but they still located in the rooms of their particular center of competences.

Up upon completion of projects, these temporary project teams dissolved and project participants moved on their next assignments to serve for other projects.

Therefore, the targeted business values of these ongoing software projects have never been the utmost priority for these independent silos.

They tend to see their work as checkboxes they ticked for one project over here and another project over there.

Leadership and matrix organizational model didn't teach them how professionals should commit their business to improve the bottom line, including sales, revenue, and profit.

A McKinsey Quarterly article written by McKinsey & Company has also clearly illustrated this **illusion of cost optimization beyond the matrix organization**.

Gartner has estimated that organizations worldwide have been yearly **spending 600 billion USD to recover their IT systems** from non-scheduled maintenance work and defects.

Now let's take a short moment to visualize how the change management and impediment handling of software projects played out. How they played out in a project configuration with the waterfall model, with the matrix organization, and without the scrum process.

Yes. You're right.

Management and employees treated change management, impediment, and error handling as if they're ill exceptions which shouldn't have happened in the first place.

Therefore, changes in a software project have been the synonym of delays. They usually created a domino effect of cascading delays. Teams required someone to blame and finger-point for defects and impediments.

Last, but not least, because silos did not have a mechanism in place to process, fix, and learn from their errors, they kept on repeating the same mistakes.

Furthermore, they kept on **augmenting the amount of technical debt** while they passively attempted to deal with their problems.

FRUSTRATION #3. AUTOCRATIC DECISIONS OVERRULED DEMOCRATIC DECISIONS

Steve Jobs once said:

"It doesn't make sense to hire smart people and tell them what to do. We hire smart people, so they can tell us what to do."

However, this is precisely opposite of how most of the mainstream leadership used to operate to make decisions before the scrum era.

Before we had the scrum process in our organizations, autocratic decisions from leaders overruled the combined intelligence of their teams.

They invalidated the democratic decision-making ability of groups who were in charge of doing the real works which spanned the entire software engineering lifecycle from the conception of software to its operations.

The remoter a decision was shifted away from work centers (teams) it impacted, the more difficult it was to give a correct mission-critical decision.

The judgments from leaders used to be usually impulsive, not thoroughly thought-out, mostly late and tentative, but sometimes even too early.

These **autocratic decisions** imposed from the top made employees feel undervalued. They entirely hindered the ability of their organizations to come up with creative and innovative solutions to handle competitive business and software-related challenges.

Furthermore, they discouraged software engineering teams from giving their inputs at the times when they're asked to contribute decisions.

It was a brief overview of how we used to develop and deliver our software services and

service products before we adopted the scrum framework in our organizations.

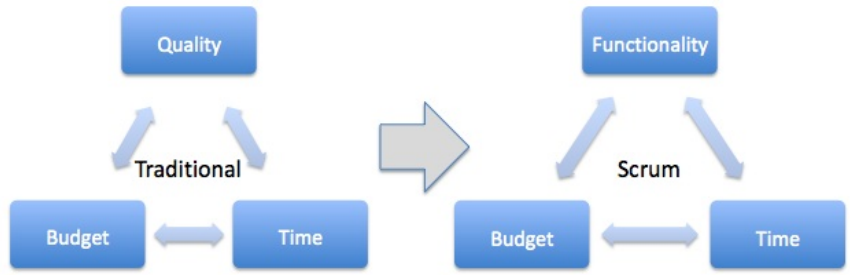
Now let's have a look at how we sorted out these chaos and frustration elements with the help of the scrum process.

Zen, keep calm ... it can only get better ... tomorrow they will start their training with the Scrum Institute!

WHAT MAKES THE SCRUM FRAMEWORK SUCCEED?

The Scrum Framework changes the classic triangle of project management.

Organizations do no longer need to sacrifice one of the time, budget, or quality. The new triangle is now emerging between the budget, time, and functionality. And none of these project success elements have to be endangered.



Triangle of Project Management

According to the Scrum framework, quality is no longer optional. To deliver what clients are paying for to flourish their businesses, the Scrum Teams strive to provide the best possible software they're jointly able to build.

In the Scrum framework, the factors which define when a feature is complete and when it meets the required quality standards are set by Definition of Done (DoD). DoDs specify the expected outcome in terms of functional and non-functional requirements, design, coding, unit testing, end-user validations, documentation, and so on. DoDs are defined in the levels of both user stories and tasks. DoDs of user stories focus on functional and non-functional client requirements, whereas DoDs of tasks focus on the desired working activities from the Scrum Team members.

The Scrum Team is not allowed to close the user stories, and obviously, the tasks that do not fulfill their DoDs. Scrum Product Owner and the Scrum Team define user stories and their tasks through-

out the course of the Scrum software engineering process incrementally.

This incremental development allows the team to remain adaptive and adjust their next best actions in a controlled manner without the additional costs and risks of jeopardizing large chunks of previous work.

The Scrum Team builds a potentially shippable software product increment until the end of each Sprint. The team demonstrates and discusses these increments with the Scrum Product Owner and client stakeholders **to get and incorporate their feedback towards the next steps of their project.**

This flexibility applies to not only software delivery but also the operational processes. So, the Scrum Framework allows **the optimization of the use of resources (human, time, budget, material) and the minimization of wastes.**

Studies have shown that Scrum has the following positive effects in practice:

- More frequent code deployments,
- Faster lead time from committing to deploying code,
- Faster mean time to recover from downtime,
- Lower change failure rate,
- Better product quality,
- Reduced or identical costs compared to pre-Scrum deployment,
- Improved productivity and throughput,
- Improved code and operational reliability,
- Enhanced organizational performance and client satisfaction,
- Improved market penetration, market share, and profitability of organizations,
- Improved market capitalization growth,
- Improved motivation of employees.

Introducing and adopting the Scrum Framework is non-trivial. And yet, the adaptive and iterative approach of the Scrum Framework handles this initial burden, and it copes with ever-changing client and business requirements better.

Thus, the Scrum Framework is, in most cases, a better alternative to the classical software engineering methodologies.

SCRUM ROLES - THE SCRUM TEAM

The Scrum Framework recognizes three roles:

- **The Product Owner,**
- **The Scrum Team Member,**
- **The Scrum Master.**

In addition to other programs it's providing to its worldwide students, **International Scrum Institute™** provides three primary training and certification programs for these three roles.

These programs are namely **Scrum Product Owner Accredited Certification™ (SPOAC™)**, **Scrum Team Member Accredited Certification™ (STMAC™)**, and **Scrum Master Accredited Certification™ (SMAC™)**.

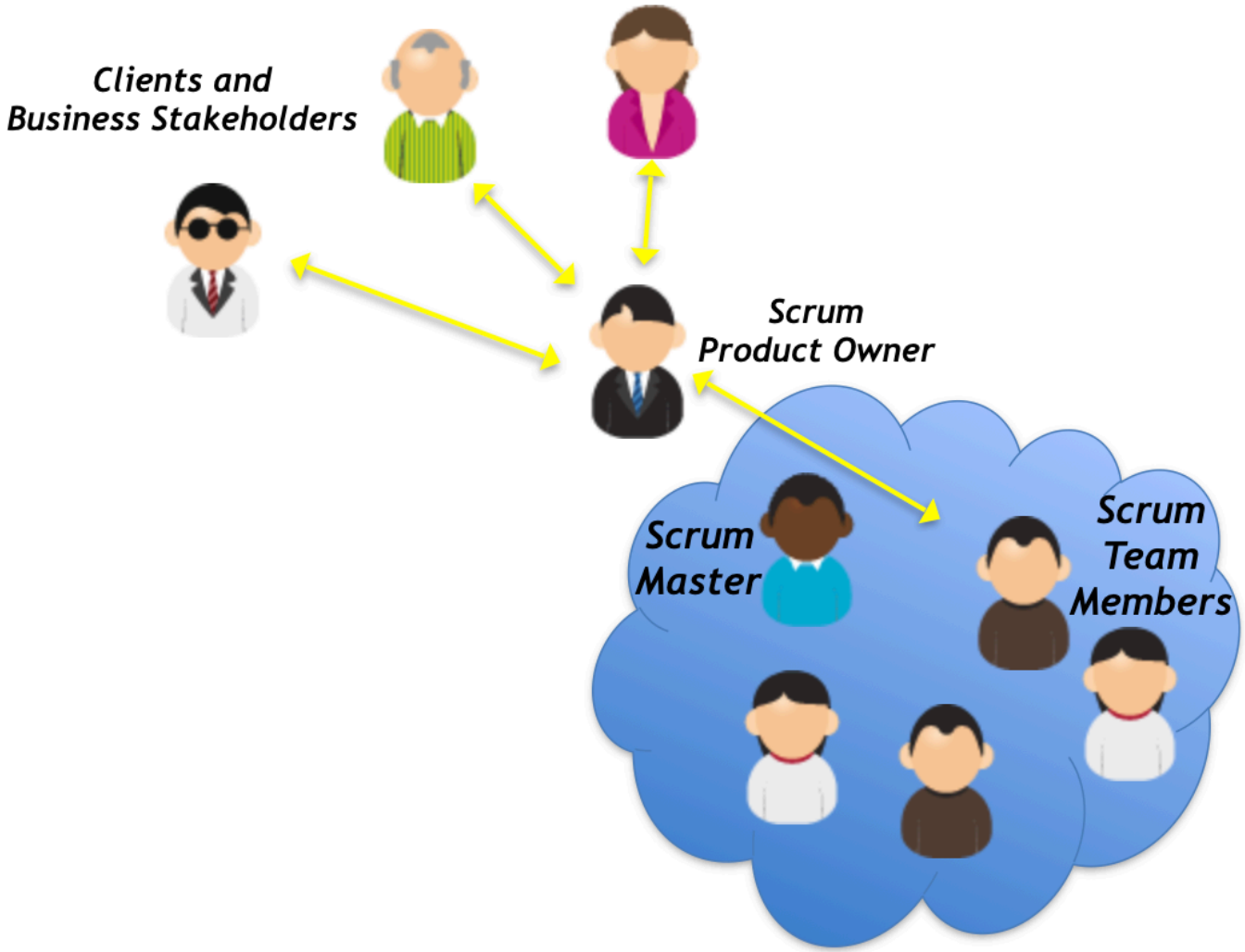
A proper scrum organization must adequately possess people from all these three skillsets. That's particularly essential to succeed with the scrum software development framework.

None of these roles is dispensable or replaceable. They aren't combinable with the other scrum roles and functions.
Each Scrum Product Owner typically works together with one scrum team. Each Scrum Team has its own Scrum Master, and each Scrum Master cares and works with one single Scrum Team.

Please don't underestimate the importance of understanding the purpose and function of these roles and employing them with adequate talents.

Many times we observed that the root cause of difficulties of a scrum team is either because these roles are not understood or they don't employ the right people.

Each of these roles has a defined set of responsibilities. Only if the owners of these roles fulfill these responsibilities, closely interact, collaborate, and work together, they can finish a Scrum project successfully.



Scrum Roles & Stakeholders

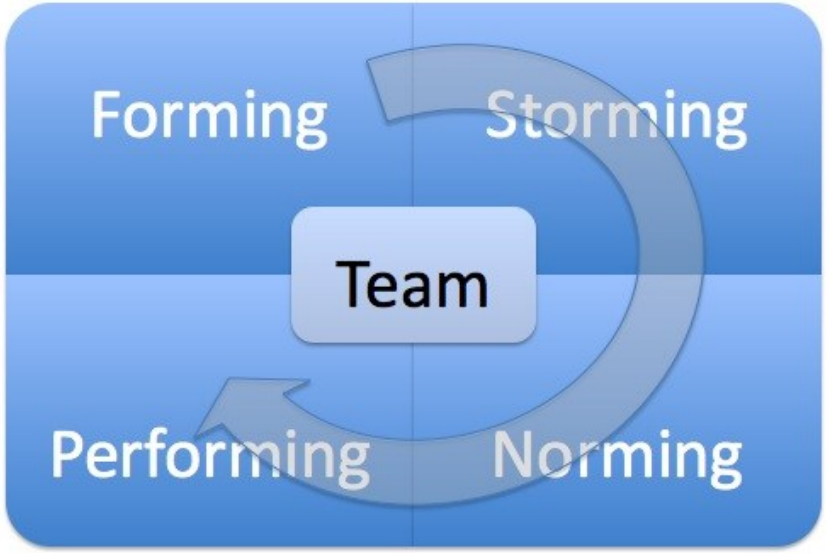
The Scrum Team

Within the Scrum Framework, dedicated Scrum Teams do all work delivered to the business clients. A Scrum Team is a collection of individuals working together to provide the requested and committed product increments.

To work effectively, it is essential for a Scrum Team that everyone within the team:

- Embraces values of the Scrum Framework such as Courage, Focus, Commitment, Respect, and Openness,
- Adheres the same norms and rules,
- Follows the common goal, which wires them to both IT and business outcomes.

When setting up a new Scrum Team, you always need to keep in mind that no new team will deliver with the highest possible performance right from the beginning. After setting up the team, it has to go through certain phases as described by the **Tuckman-Model**: Forming, Storming, Norming, Performing.



Tuckman's Stages of Team Development

How long it takes until the Scrum Team reaches the Performing Phase varies from team to team. Hiring good basketball players for the same club will not make a good basketball team as soon as they start to play together. They first need to learn and adapt their playing styles, their strengths and weaknesses to assist each other, and to play in harmony. Scrum teams are not that different. Therefore, it's vital to keep in mind that it usually takes about 3 to 5 Sprints until the

team becomes mature enough to deliver its results effectively and predictably.

Characteristics of a Scrum Team

Scrum Teams have the following characteristics:

- Team members share the same norms and rules,
- The Scrum team as a whole is accountable for the delivery,
- The Scrum Team is empowered,
- The Scrum Team is working as autonomous as it is possible,
- The Scrum Team is self-organizing,
- The skills within the Scrum team are balanced,
- A core Scrum Team is small and has no sub-teams,
- The Scrum Team members are dedicated to their teams with 100% capacity,
- Scrum Team members are collocated, and they ideally share the same room.

Rules & Norms

The environment, business, IT, and geographical ecosystem of Scrum Teams invisibly define some of the norms the teams follow. And yet, to become a truly successful Scrum Team, some rules and norms should be explicitly developed and exercised during the Norming phase.

These common standards are essential, and they can't be overemphasized to deliver smooth gameplay, IT, and business results. Otherwise, the Scrum Team members would have to continually switch back and forth between different value systems and rule sets, and they waste their valuable time. Just a few examples of such norms and rules are:

- The goal, scope, duration, location, participants and outcomes of Scrum Rituals (Events),
- Required level of details to write clear, concise and unmistakable Definition Of Dones (DoDs),
- Guidelines to prioritize and estimate user stories and tasks,
- Guidelines, procedures and the level of details to create living documents,

- Tools to use and tools not to use (remember, sometimes less is more),
- Coding standards,
- Tools and guidelines to build/perform manual/automated tests and ensure quality,
- The process to resolve bugs,
- The process to handle change requests,
- The process to prepare to product increment demonstrations during Sprint Review Meetings,
- The process to handle the outcomes of each Scrum Ritual (Event),
- Frequency, depth, and duration of Backlog Refinement Meetings.

Accountability

The Scrum Team as a whole is responsible for delivering the committed user stories in time and with the highest possible quality.

A good result or a failure is never attributed to a single team member but always the result of the Scrum Team.

Empowerment & Self-organization

The Scrum Team has to be empowered to define

- What the team commits to deliver at the end of the Sprint,
- How the committed user stories will be broken down into tasks,
- Who will perform a specific task and in which order the tasks are implemented.

Only if the Scrum Team is empowered to decide these and similar internal decisions, the team members will work with higher performance and motivation for the interest of their client stakeholders.

Balanced set of skills

Each Individual within the Scrum Team will most certainly have specialized skills, focus, and personal preference of interests. However, to achieve the best possible performance, your Scrum Team needs to have a balanced set of

skills. Only then the Scrum Team will be able to deal with the ever-changing IT and business challenges, and they can act as autonomous as it is possible.

That means a Scrum Team should be multidisciplinary (designers, developers, testers, architects, etc.) right from the beginning. On the other hand, this also means that each team member should learn a little bit from each other's specialization. For instance, to be able to finish a committed user story until the end of the Sprint, a developer should willingly write and execute tests, and consult the tester whenever necessary.

The roles of the Scrum Team members are not compartmentalized like the architect, the developer, the tester, and so on. They all share the same title, "Scrum Team Member" regardless of their core personal competencies.

Size of the Scrum Team

Scrum Teams are small. The ideal size is 7 +/- 2 people.

Note that if the Scrum Team contains more than nine members, your team will most probably suffer due to excessive overhead of alignment and communication. And yet, there is no one size fits all answer. Your Scrum Teams may still productively function even if they have less than five or more than nine members.

The only way to find this out is to test, learn, and adapt. If you find out that a team of 13 people cannot perform well enough, then these Scrum Teams need to be split into two teams. These Scrum Teams should closely align, and they correlate their goals and user stories. Besides that, they work independently.

Collocation

To minimize unnecessary communication overhead, each Scrum Team should be collocated. If the work has to spread over multiple geographical locations, independent Scrum Teams need to be created. These teams need to align and correlate their goals and user stories.

Responsibilities of the Scrum Team

The Scrum Team has specific responsibilities they need to fulfill:

- They have to breakdown the user stories, create tasks, define priorities and estimates, and they self-organize the implementation. In other words, they have to create, process, and deliver the Sprint Backlog.
- They have to perform Daily Scrum Meetings.
- They have to ensure that at the end of the Sprint, potentially shippable product increment is delivered and demonstrated.
- They have to update the status and the remaining work efforts for their tasks to allow the creation of a Sprint Burndown Diagram.

Learning Scrum can be a challenge. But don't worry about it. We built our products to help and serve you!

THE SCRUM MASTER ROLE

The Scrum Master serves all participants of a Scrum Project and the external stakeholders to comprehend and apply the Scrum Framework correctly.

He or she supports the Scrum Team to execute the Scrum Framework successfully and contributes them to improve their productivity and performance continuously.

The role of the Scrum Master is to establish the Scrum Process in its organization, the new way of thinking and acting.

Furthermore, the Scrum Master acts as a change agent. He or she coaches the team to develop new team norms and standards. The Scrum Master has its desk somewhere very close to the rest of the scrum team.

Essential tasks of a Scrum Master are:

- To establish the Scrum Framework in his or her business and IT ecosystem,
- To act as a change agent and support the adaptation of existing processes to maximize productivity of the Scrum Team.
- To coach the Scrum Team to understand and live the values of the Scrum Framework,
- To ensure efficient and close collaboration between the Scrum Product Owner and the Scrum Team,
- To remove impediments which hinder the continuity of work,
- To lead progress of work by serving,
- To moderate the Scrum Rituals (Scrum Events).
- To guard the Scrum Team from external interference and interruptions while the team does work it has originally committed for a Sprint.

Easily Learn Scrum and Officially Prove Your Knowhow

To effectively do this work, a Scrum Master needs to possess savvy moderation and coaching skills. He or she needs to be a continuous learner to inspire others to learn, change, and grow.

To learn more about Scrum Master's duties as a facilitator, I recommend you to have a look at this article: **If I had 5 Minutes to explain Scrum Master As a Facilitator**.

The Scrum Master is part of the Scrum Team and acts as a servant-leader for the Scrum Team. In the beginning, this will be a full-time job so that the Scrum Master will not be able to contribute to the Sprint results directly. However, after a few Sprints, while the Scrum Team approach to the Performing phase of the Tuckman model, the initial workload as moderator and coach will reduce. So, the Scrum Master could actively contribute to the Sprint goals.

Since there must be trust between the Scrum Master and the Scrum Team members, it can always be a good idea that the Scrum Team chooses its Scrum Master. However, in reality, the management usually imposes who the Scrum Master will be. To get the required trust, the Scrum Master should have no line management responsibility above the Scrum Team members. Otherwise, open communication in the Scrum Team and joint ownership of work and decision-making ability of the Scrum Team can suffer.

Guarding The Scrum Team, Removing Impediments

An essential job of the Scrum Master is to safeguard the Scrum Team from a false sense of urgency. Line management and the Scrum Product Owner often attempt to add unplanned user stories to the Sprint Backlog while the team focuses on the work of a planned Sprint.

However, one of the critical aspects of the Scrum Framework is that all user stories are known and committed only during the Sprint Planning Meetings. The Scrum Team cannot be forced to take over new user stories. The job of the Scrum

Master is to ensure that until the next Sprint Planning Meeting, these new user stories are stored in the Scrum Product Backlog.

Alternatively, if the ongoing Sprint does not make any business and/or technical sense to continue, it can be canceled, and a new Sprint can be planned.

Scrum Team members should only concentrate on delivering client value by building potentially shippable product increment. The Scrum Master helps by removing impediments that block or slow down the progress of work.

Examples of removing impediments could be:

- To arrange support, resources,
- To find missing knowhow, and
- To do hands-on work to help the Scrum Team Members.

Scrum Master as a Change Agent

One of the cornerstones of the Scrum Framework is the continuous improvement through Inspect & Adapt.

The Scrum Master hosts and moderates the Scrum Retrospective Meeting, and his or her job is then to facilitate, control and measure the change of the identified shortcomings.

Facilitation of Scrum Rituals (Events)

The Scrum Framework defines several meetings that have to be organized and facilitated by the Scrum Master:

- Scrum Grooming (Backlog Refinement) Meetings,
- Sprint Planning Meetings,
- Daily Scrum Meetings,
- Sprint Review Meetings, and
- Sprint Retrospective Meetings

THE SCRUM PRODUCT OWNER ROLE

The Scrum Product Owner is a central role within the Scrum Framework. That role unifies product and project management tasks, and it's also firmly integrated with software development and delivery.

The product owner's role is far broader than traditional project management, program management, or product management roles.

He or she represents the end customers and/or other stakeholders and is responsible for maximizing the value of the product by ensuring that the Scrum Team delivers the right work at the right time. The Scrum Product Owner decides the software requirements provided for a specific software version, and when the software will be released. She represents functional and non-functional demands from end-users.

That means that the Scrum Product Owner has to work very closely with the Scrum Team and coordinates their activities over the entire lifecycle of the project. No one else is allowed to impose the Scrum Team to work for a different set of priorities.

Essential tasks of a Scrum Product owner are:

- To manage and clarify project requirements,
- To guide releases and to ensure return on investment (ROI),
- To closely work with the Scrum Team and enable it to deliver the correct work on time,
- To manage stakeholders and their expectations,
- To manage the Scrum Product Backlog.

The Scrum Product Owner can delegate certain activities (like physically maintaining the Scrum Product Backlog). However, he or she still owns the accountability of his or her tasks.

Managing the Product Backlog

The Scrum Product Owner is the only person allowed to own the contents of the Scrum Product Backlog. That means he or she needs to:

- Create, maintain and clearly describe user stories in the Scrum Product Backlog,
- Prioritize user stories to accomplish business goals and fulfill the mission of software product,
- Ensure that the Scrum Team correctly comprehends and implements the user stories in the Scrum Product Backlog.

Release Management

The Scrum Product Owner is responsible for reaching the project goals. He or she creates and maintains the release plan and decides about deliveries, end-user functions, and the order they need to be delivered. Scrum Product Owners often manage the costs and budget of Scrum Teams too. They collaborate with the Scrum Team members to fine-tune, prioritize, and estimate user stories.

Stakeholder Management

External stakeholders should not directly bring their demands to the Scrum Team members. Instead, the Scrum Product Owner should collect and assess required functionalities with the stakeholders (for instance, with internal clients, representatives of external clients or end-users). The Scrum Product Owner combines, filters and initially prioritizes these user stories before he or she discusses them with the Scrum Team.

Collaboration With The Scrum Team

For a successful project, the Scrum Product Owner and the Scrum Team must work very closely. The Scrum Product Owner is responsible for ensuring that the Scrum Team members are informed and aligned about the aimed goals of software they're building.

During Sprint Review Meetings, the Scrum Product Owner is responsible for inspecting, accepting, or declining deliverables of the Scrum Team.

THE SCRUM TEAM MEMBER ROLE

The Scrum Team Members implement the software. They jointly decide the number of requirements that they can undoubtedly deliver during a particular product increment called **"Sprint"**.

A high-performer scrum team has most of the software engineering skills typically in it. Software developers, architects, testers, database administrators, and team members from all other roles work together.

They jointly build and deliver great software their client is paying for.

Scrum team members do no longer belong to a functional silo of a matrix organization. Developers do no longer belong to software development competence centers, and testers do no longer belong to the software testing competence center, and so on.

Regardless of their past coordinates in the organization, members of a scrum team belong to their particular scrum project.

Now their job is to build the best possible software to deliver the requirements of their scrum product owner.

Characteristics of scrum teams are:

- Empowered and Autonomous,
- Cross-functional,
- Self-organized and small,
- Full-time participants,
- Working in the same room,
- One for all, all for one.

It's an excellent time to remind that the Scrum Team members follow Scrum values persistently.

- Courage
- Focus
- Commitment
- Respect
- Openness

HOW DOES THE SCRUM FRAMEWORK WORK WITHOUT A PROJECT MANAGER?

In a traditional project, typically, a Product Manager defines the requirements. Then the Product Manager delegates the realization to a Project Manager. Afterward, the Project Manager coordinates all activities needed to deliver the requirements of the project.

The Scrum Framework does not define a "Project Manager" role in the classical sense.

And yet, the tasks of this role are still required to deliver a project successfully. Within the Scrum Framework, the responsibilities of a Project Manager are distributed over the functions of the Scrum Product Owner and the Scrum Master.

The definition of the role and responsibilities of the Scrum Product Owner takes possible conflicts that may arise between the Product Owner and the Project Manager into account. Decisions about functionality, release planning, and budgeting can be made much more comfortable, quicker and better if one person is responsible for the execution, controlling, and documentation of those activities. **Otherwise, there would always be a constant tension between the Scrum Product Owner (not responsible for the project) and the Project Manager (not responsible for the execution of work).**

The goal of any reliable process should be to avoid this potential conflict that impacts the functional and tactical administration of the work that the Scrum Team performs. And that's precisely what the Scrum Framework aims to accomplish. **Therefore, the typical responsibilities of Project Managers and Product Managers are merged into the Scrum Product Owner role.**

Nevertheless, the Scrum Master takes over some responsibilities of a traditional Project Manager. **Tracking the tasks in Sprints and facilitating the resolution of impediments are among his duties.** Since he or she is part of the Scrum Team, it is much easier and much more efficient to handle such activities directly in the team.

SCRUM USER STORIES

The entries in the Scrum Product Backlog are often written in the form of User Stories. A User Story tells a short story about the requirements of someone while he or she is using the software product we are building.

It has a name, a brief narrative, and acceptance criteria for the story to be counted as completed. The advantage of user stories is that they precisely focus on what the user needs and wants without going into the details on how to achieve them. How to achieve them will be the job of the Scrum team at a later stage,

There are different recommendations on how to define User stories. A well known and reliable template is:

As an [actor], I [want|must] [action] so that [achievement]

Or in a shorter version:
As an [actor], I [want|must] [achievement]

The Actor is the owner of the user story. That is often a user, but it is advisable to be more specific. By using particular actors such as an administrator, logged in customer, or unauthenticated visitor or so on, user stories become distinctive. So they set requirements into a proper context everyone can understand.

The Action is what the Actor wants to do. If it is a mandatory requirement, it can be prefixed as **must**. Otherwise, it's prefixed as **want**.

The Achievement is what the Actor wants to achieve by performing the Action. That's the Actor's envisioned business result or a functional technical component that emerges once the Action is completed.

SCRUM EFFORT ESTIMATIONS – PLANNING POKER®

All user stories within the Scrum Product Backlog have to be estimated to allow the Scrum Product Owner to prioritize them and plan releases. That means the Scrum Product Owner needs a reliable assessment of how much the delivery of each user story will take.

Nevertheless, it is recommended that the Scrum Product Owner does not interfere with the estimations that the Scrum Team performs. So the Scrum Team delivers its estimates without feeling any pressure from the Scrum Product Owner.

The Scrum Framework itself does not prescribe a way for the Scrum Teams to estimate their work. The teams who rely on the Scrum Framework do not deliver their estimates of user stories based on time or person-day units. **Instead, they provide their estimates by using more abstract metrics to compare and qualify the effort required to deliver the user stories.**

Common estimation methods include:

- Numeric sizing (1 through 10),
- T-shirt sizes (XS, S, M, L, XL, XXL, XXXL), or
- the Fibonacci sequence (0, 1, 2, 3, 5, 8, 13, 21, 34, etc)

```
As a <customer> I want to <see the catalog of salable items>
                  so that <I can order one of them>
As an <administrator> I want <be able to disable accounts>
As a <trainee> I must <answer all the questions>
```

An Example User story

The Scrum Team members must share a common understanding and consensus of the unit of estimations they use so that every member of the team feels and acts comfortable with it.

Planning Poker® / Scrum Poker

One commonly used method for the estimation process is to play Planning Poker® (also called Scrum Poker).

When using Planning Poker®, the social proof influence among the Scrum Team members are minimal. Therefore, the Scrum Team produces more accurate estimation results.

What you need to play Planning Poker® game is straightforward:

- The list of features to be estimated
- Decks of numbered cards.

A typical deck has cards showing the Fibonacci sequence, including a zero: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89. Other similar progressions are also possible. The reason for using the Fibonacci sequence is to reflect the uncertainty in estimating larger items. It would be a waste of time to discuss if a user story should have a size of 19, 20 or 21. **And yet, it's relatively easier to decide if the user story fits better to the size 13 or 21.**

A high estimate could mean that the Scrum Team members have not very well understood the user story yet. So they can attempt to secure extra capacity for contingency before their commitment to this user story. Alternatively, that could also mean that the user story should be broken down into multiple smaller user stories. **Scrum teams can usually estimate smaller and clearer user stories with higher confidence.**

Finally, the Scrum Team plays Planning Poker® by adhering the following steps:

- **The Scrum Product Owner presents the story to be estimated.** The Scrum Team asks questions, and the Scrum Product Owner articulates the user story in more detail. If the Scrum Team has to assess many user stories,

estimates can be time-boxed in a way that the Scrum Team does not spend more than a few minutes for each user story. If the team cannot still estimate a user story in a given time-box, then this could be a signal to which the Scrum Product Owner needs to pay attention. This signal indicates that either the end-user requirement or the user story or both of them are not clear enough, so they need to be rewritten.

- Each member of the Scrum Team privately chooses the card representing the estimation.
- After everyone has selected a card, all selections are revealed.
- People with high and low estimates are asked to explain their assessment because they may have thought something that the majority of the Scrum Team members have been unable to see.
- Another estimation is done for this particular user story until the team reaches a consensus for an estimate.
- The Scrum Team repeats the game until they estimate all user stories.

Planning Poker® is a registered trademark of Mountain Goat Software, LLC.

DEFINITION OF DONE (DOD)

In the Scrum framework, the factors which define when a feature is complete and when it meets the required quality standards are set by Definition of Done (DoD).

As it was clarified before in this material, DoDs specify the expected outcome in terms of functional and non-functional requirements, design, coding, unit testing, end-user validations, documentation, and so on. DoDs are defined in the levels of both user stories and tasks.

DoDs of user stories focus on functional and non-functional client requirements, whereas DoDs of tasks focus on the desired working activities from the Scrum Team members.

The Scrum Team is not allowed to close the user stories, and obviously, the tasks that do not fulfill their DoDs. Definition of Done (DoD) is used to decide whether a User Story from the Sprint Backlog is complete or not. DoD is a comprehensive checklist of required activities to ensure that only truly completed features are delivered, not only in terms of functionality but in terms of quality as well. The norms which a Scrum Team uses to define DoDs may vary from one team to another, but it must be consistent within a given Scrum Team.

There are usually different DoDs at various levels:

- DoD for a Project/Product (In the project goals)
- DoD for a Release (In the release goals)
- DoD for a Sprint (In the sprint goals)
- DoD for a User Story (In the User Story)
- Dod for Tasks (In the task)

One more essential thing to keep in mind here is that **a DoD is neither static nor indisputable.** During the course of a project, a release, or a sprint, a DoD can be challenged by anyone from the Scrum team or other business and IT stakeholders. As long as the proposed changes of a DoD makes sense and they're brought up to bring the project to success, the Scrum Team and the Scrum Product Owner should be open-minded to listen to those proposals and implement them when and where necessary.

THE SCRUM PRODUCT BACKLOG

Product Backlog (Scrum Backlog) or Scrum Product Backlog is the central element to manage all of the known requirements of a Scrum Project.

It consists of all customer requirements and work results that are needed to execute and finish a successful project. **As requirements, you count functional and non-functional requirements and other features related to user experience and user interface design.**

The Product Backlog contains feature requests and their high-level user stories. These can also include pre-requisite or complementary project requirements such as building test and development environments. Moreover, other user stories required to resolve known bugs or reduce technical debt or improve certain software features have also their places in the Product Backlog as well.

Every Scrum Project has its Product Backlog. The Scrum Product Owner is responsible for the creation, maintenance, and fine-tuning of a Product Backlog. **The Product Backlog doesn't and shouldn't answer the question of how these requirements will be developed and delivered.**

That's the duty of the Scrum Team. The Scrum Team decides and documents the required tasks to address these requirements in Sprint Backlogs. **Note that Product Backlog and Sprint Backlog are physically separate entities although entries in the Product Backlog drive the contents of the Sprint Backlog.**

The owner of the Scrum Product Backlog is the Scrum Product Owner. The Scrum Master, the Scrum Team, and other Stakeholders contribute it to build a broader list of user stories to bring the product to success.

Working with a Scrum Product Backlog does not mean that the Scrum Team is not allowed to create and use other artifacts to manage work. Examples for additional artifacts could be a

summary of the various user roles, workflow descriptions, user interface guidelines, story-boards, or user interface prototypes. However, note that these artifacts do not replace the Scrum Product Backlog but complement and detail its contents.

The Product Backlog is a living document. Similar to how the software incrementally improves, the Product Backlog grows in time as well. **The Scrum framework doesn't require a separate change management process per se. The Scrum Product Owner creates the first versions of the Product Backlog based on his best initial understanding of the product.**

While the Scrum Product Owner closely observes how the product emerges from sprint to sprint and while the knowledge about client requirements augments, he or she adds, removes and fine-tines requirements in the Product Backlog.

The Scrum Product Owner prioritizes requirements in the Product Backlog. The more priority an element in the Product Backlog has, the more details it should contain. So the Scrum Team can easily make sense of these high-priority requirements and create the required tasks to build them.

Furthermore, by using story points, the Scrum Team regularly estimates the requirements in the Product Backlog. These estimations should be fine-tuned and improved for high-priority user stories to make them ready for Sprint Planning Meetings.

The Scrum Product Owner uses the Scrum Product Backlog during the Sprint Planning Meetings to introduce the highest priority user stories to the Scrum Team. The Scrum Team then determines which items they can complete during the upcoming Sprint.

Each Scrum Product Backlog has specific attributes that differentiate it from a simple to-do list:

- A user story in the Scrum Product Backlog always add a business or technical value to its client, business owner and end-users,

- All user stories in the Scrum Product Backlog are prioritized and ordered from highest to lowest priority,
- The level of details stored in a user story depends on its position within the Scrum Product Backlog,
- User stories are estimated,
- Scrum Product Backlog is a living document,
- There are no action items or low-level tasks in the Scrum Product Backlog.

User Stories Add Value To Client, Business, and Systems

Each user story in the Scrum Product Backlog must offer some client value. User stories without any client value are a pure waste of resources, and they should be eliminated. The Scrum Product Backlog can include user stories for:

- The exploration of client needs or various different technical options,
- The specification of functional and non-functional requirements,
- The summary of high-level break-down of the work necessary to launch the software product,
- Setting up non-production development and demonstration environments,
- Remediating defects.

ToDo List

ID	Story	Estimation	Priority
7	As an unauthorized User I want to create a new account	3	1
1	As an unauthorized User I want to login	1	2
10	As an authorized User I want to logout	1	3
9	Create script to purge database	1	4
2	As an authorized User I want to see the list of items so that I can select one	2	5
4	As an authorized User I want to add a new item so that it appears in the list	5	6
3	As an authorized User I want to delete the selected item	2	7
5	As an authorized User I want to edit the selected item	5	8
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9
8	As an administrator I want to see the list of accounts on login	2	10
Total		**30**	

Example Scrum Product Backlog

Some tasks may not add direct value to the functionality of software system and business clients. Nevertheless, they should add value by increasing quality, reducing technical debt, and increasing maintainability of the product during the long run.

Product Backlog Is A Living Document

The Scrum Product Backlog is a living document. It changes and evolves throughout the entire course of a project. If needed, new user stories are added, existing user stories may be altered, defined in more detail, or even deleted. Requirements of Scrum projects are no longer frozen early on like we used to have them with the Waterfall Methodology.

Instead, the final set of requirements within the Scrum Product Backlog are developed iteratively, together with the emerging software increments. That is different from traditional requirements engineering. Still, this new way of handling client requirements allows the Scrum Team to maximize client value and minimize waste of resources.

Different Level Of Details

The requirements in the Scrum Product Backlog can have varying depths with their granularities. **Only those requirements that will be implemented during the next few Sprints should be defined with greater detail.** All other user stories should remain coarse-grained; they should be only processed "just in time" before the Scrum Team needs to know more about them.

It does not make sense to invest time and resources into the specification of these requirements, as some of these requirements may change or disappear until they are needed for implementation. **"Just in time" handling of requirements is one of the most excellent benefits the Scrum Framework offers to deal with "unknown unknowns" in your projects.**

No Low-Level tasks In The Product Backlog

The Scrum Product Backlog should not contain detailed task break-down of user stories. The Scrum Product Owner defines the requirements together with the business clients and stakeholders before he or she brings them to the Backlog Refinement or Sprint Planning Meetings. Detailed task break-down and distribution of these tasks among the Scrum Team Members are the responsibility of the Scrum Team.

The Scrum Product Backlog Is Ordered Based On Priority

All user stories are prioritized, and the Scrum Product Backlog is ordered based on the priority of user stories (from highest to lowest). The Scrum Product Owner performs the prioritization with the support of the Scrum Team. During this prioritization exercise, the added value created for the business of the client, costs, risks, and dependencies are the most common factors which are taken into account by the team. Thanks to this prioritization, the Scrum Product Owner can decide what the Scrum Team should subsequently build and deliver.

All User Stories Are Estimated

All user stories within the Scrum Product Backlog have to be estimated according to the agreed norm of story point units such as Fibonacci number or S/M/L/XL/XXL, etc. More about this comes later in this material. These estimations then impact the capacity planning of Sprints, contents of Sprint Backlog, and release plans.

Working With The Backlog

The Scrum Product Backlog needs regular care and attention. It needs to be carefully managed because **it's the source of truth to understand what your software product is all about.**

At the beginning of a project, it's filled with a lot of high-level stories that may or may not be highly relevant to contribute to the success of the project. **While the project progresses from one Sprint to another, the Scrum Product Owner and the team learn more about the project.**

Subsequently, the contents of the Scrum Product Backlog will become perfectly reasonable to reflect your product better.

After this initial setup, the Scrum Product Backlog has to be continuously maintained. The maintenance of the Scrum Product Backlog stands for:

- As new requirements are discovered, they are described and added. Existing requirements can be changed or removed as appropriate,
- Ordering (Prioritizing) the Scrum Product Backlog. The most important (highest priority) user stories are moved to the top,
- Preparing the high-priority user stories for the upcoming Sprint Planning Meetings (usually during Backlog Refinement Meetings),
- (Re-)Estimating the user stories in the Scrum Product Backlog (usually during Backlog Refinement Meetings).

The Scrum Product Owner is responsible for making sure that the Scrum Product Backlog is always in good shape. And yet maintaining the Scrum Product Backlog is a collaborative process. A reasonable capacity of the Scrum Team members should be reserved for managing the Scrum Product Backlog for the time they need to spend during Scrum Rituals (Events).

Furthermore, note that, this collaborative maintenance of the Scrum Product Backlog helps to clarify the requirements and creates buy-in of the emerging software product from the Scrum Team members.

THE SPRINT BACKLOG

The Sprint Backlog stores and maintains user stories required to deliver the shippable software increment of a Sprint.

These user stories are the ones the Scrum Team committed during the Sprint Planning Meeting. All user stories in the Sprint Backlog are estimated to enable the monitoring and tracking of the work.

The Sprint Backlog is a living artifact, and during the course of Sprint, the Scrum team members continuously update it.

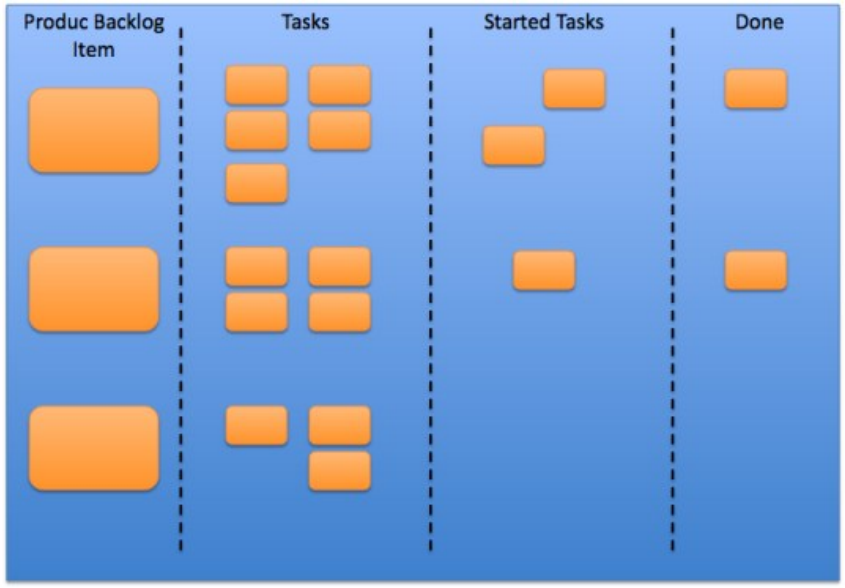
When a Scrum Team member works on a task, his or her name is associated with it. The status of the task is set to "Started Tasks" or "Work In Progress". When the task is completed according to its Definition of Done, its status is set to "Done" or "Completed".

New tasks (not new user stories) can be added to the Sprint Backlog during the Sprint. At the end of every day, the remaining effort to deliver the full Sprint Backlog is calculated. This helps the Scrum Team and the Scrum Product Owner to monitor the remaining amount of work to bring the Sprint goals to success.

The Sprint Backlog can be kept electronically by using a Scrum Project Management Software or spreadsheet software such as Excel or Google Sheet. Alternatively, for smaller and new teams, cards (or post-its) glued on a real physical board can be used too.

The latter has some advantages such as transparency of work and easy access for everyone, including the stakeholders. Its downside is that it's not sustainable if the Scrum Team is distributed over multiple locations.

The figure below shows a sample of how such a Sprint Backlog Board can be built. The structure should be adapted to reflect the needs of the project and the Scrum Team.



A Sample Sprint Backlog Board

It's evident that for a Scrum Team to work productively, they need to understand the difference between a Product Backlog and a Sprint Backlog, and how these two elements interact to execute forward the project.

Remember that there is a complete list of user stories from the Product Backlog. And now, a small subset of this list is being moved to Sprint Backlog to accomplish the goals of the Sprint.

During the Sprint Planning Meeting, all members of the Scrum Team should discuss what tasks must be done and how these tasks will be completed.

It's at this point that each item on the Sprint Backlog is broken down into tasks or steps that will be taken to complete the committed user stories. All of this must be clearly communicated and agreed upon, so the Scrum Team members have an unmistakable consensus about what is coming in the Sprint and what it takes to accomplish its goals.

WHAT IS A SPRINT?

In the scrum framework, all activities to implement the requirements happen during Sprints.

Sprints are cycles of work activities to develop shippable software product or service increments. Sprints are also sometimes called iterations.

You think about sprints as if they're micro-projects. As the term "Sprint" implies, a sprint doesn't take long, and it's maximum for four weeks.

Sprints are always short, usually between 2 to 4 weeks. But depending on the reliably foreseen amount of work or for Exploration Sprints, a one-week long Sprint is also typical. **With the approval of the Scrum Product Owner, a Scrum Team may need an Exploration Sprint to investigate new technology, build a mock-up or a proof of concept.**

Different from a running race, at the end of a Sprint, the Scrum Team shouldn't be out of breath and power. Instead, the Scrum Team should be fresh, and they should energetically start the next Sprint.

The Scrum Software Development and Delivery Framework has nothing to do to create pressure and stress for its team members. It's a well-known fact that people perform best when they can work focussed, relieved, and undistracted. If it's applied correctly this is what Sprints help us accomplish.

Every Sprint starts with a Sprint Planning Meeting. During the Sprint Planning Meeting, the Scrum Team decides what and how many requirements they will implement in this given Sprint. Subsequently, the Scrum Team starts the execution of activities to deliver requirements.

Daily Scrum (Daily Scrum Meeting) happens every day at the same time in the same place. The goal of this meeting is to align all members of the Scrum Team regularly. Scrum Team Members can also ask help from the Scrum

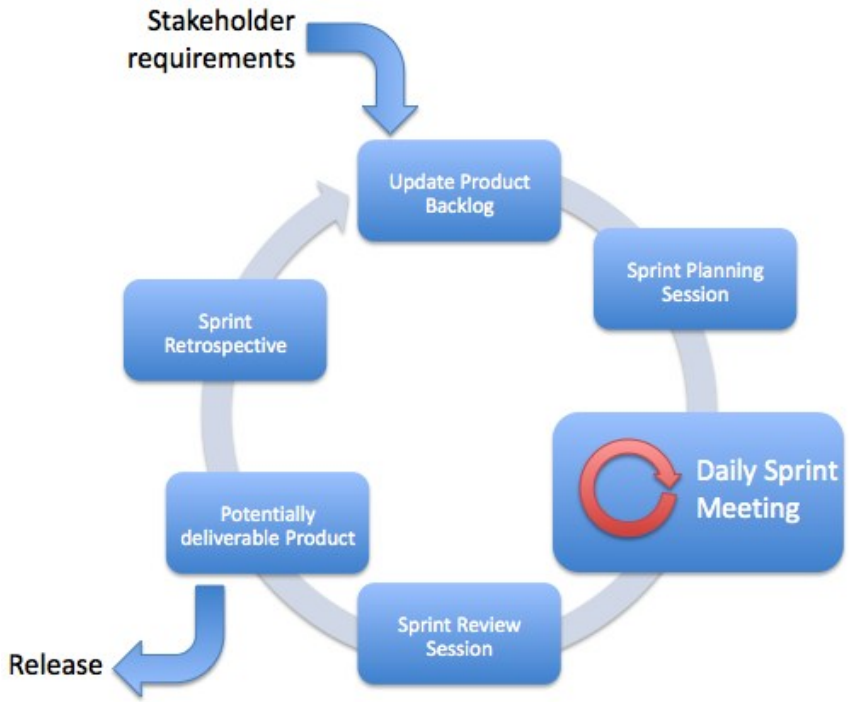
Master to remove any impediments which may potentially slow down or block the progress of a Sprint.

The Scrum Team finalizes a Sprint with Sprint Review and Sprint Retrospective meetings. The Sprint Review Meeting enables the Scrum Product Owner to review and control the work results the Scrum Team has created during the Sprint.

During the Sprint Retrospective Meeting, the Scrum Team reflects the way they work together, how they use the Scrum Framework, and how they can improve.

So the Scrum Team jointly takes these improvement potentials and feedback into account during the next Sprint Planning Meeting. **Sprints enable such feedback loops during short periods with small batch sizes of work.**

That's one of the significant reasons why and how the Scrum framework helps teams and organizations to improve themselves continuously.

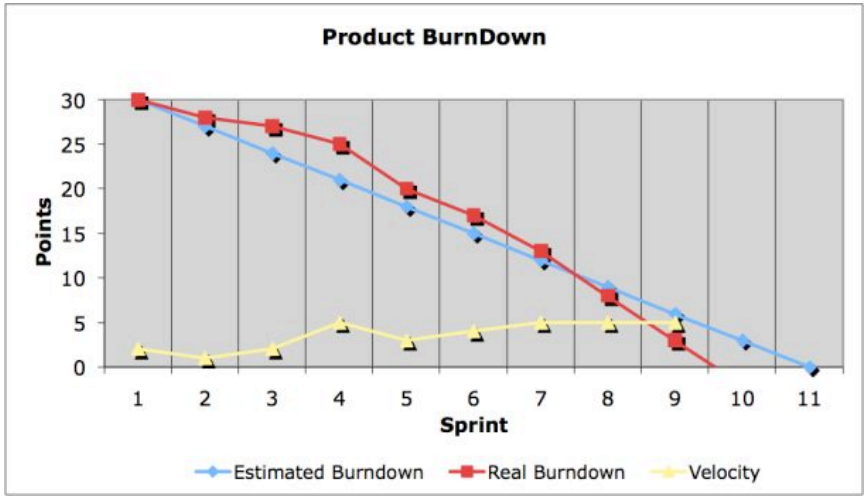


Sprints are cycles of work activities to develop shippable software product or service increments.

SCRUM BURNDOWN CHART

The "Scrum Burndown Chart" is a visual measurement tool that shows the completed work per Sprint against the projected rate of completion for the current project release.

Its purpose is to enable the Scrum Product Owner, the Scrum Team, and other stakeholders to control the progress of the project. So the Scrum Team achieves to deliver the requested software solution within the desired timeline.



Simple Scrum Burndown Chart

The speed/rate of progress of a Scrum Team is called "Velocity". It expresses the total number of story points completed that the Scrum Team delivers per Sprint (Iteration).

An essential rule to assess and calculate the Velocity is that; Only entirely completed user stories that precisely fulfill their Definition of Done (DoD) are counted. The velocity calculation shouldn't take partially completed user stories into account. (For instance, coding of a user story is done, but its tests are still missing)

Only a few Sprints after a new Scrum Team is formed, the Velocity of the team can be reliably calculated. That helps the Scrum Product Owner to predict the throughput of the Scrum Team better, and he or she can foresee what user stories the Scrum Team can deliver in a given Sprint. That would enable the Scrum Product Owner to plan software releases more accurately, with less surprises towards business clients and end-users.

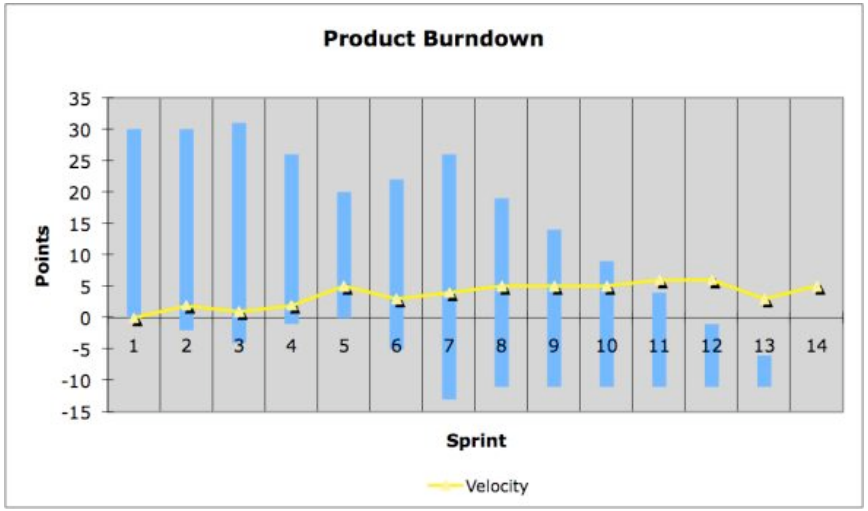
As a simple example: Let's assume the Velocity of your Scrum Team is 50 story points per Sprint.

And the total amount of remaining work has been estimated as 300 story points. Then you can predict that you need 6 Sprints to deliver all of the remaining user stories from the Product Backlog.

However, in reality, the user stories in the Scrum Product Backlog will change over the course of the project. New stories are added, and other stories are modified or even deleted.

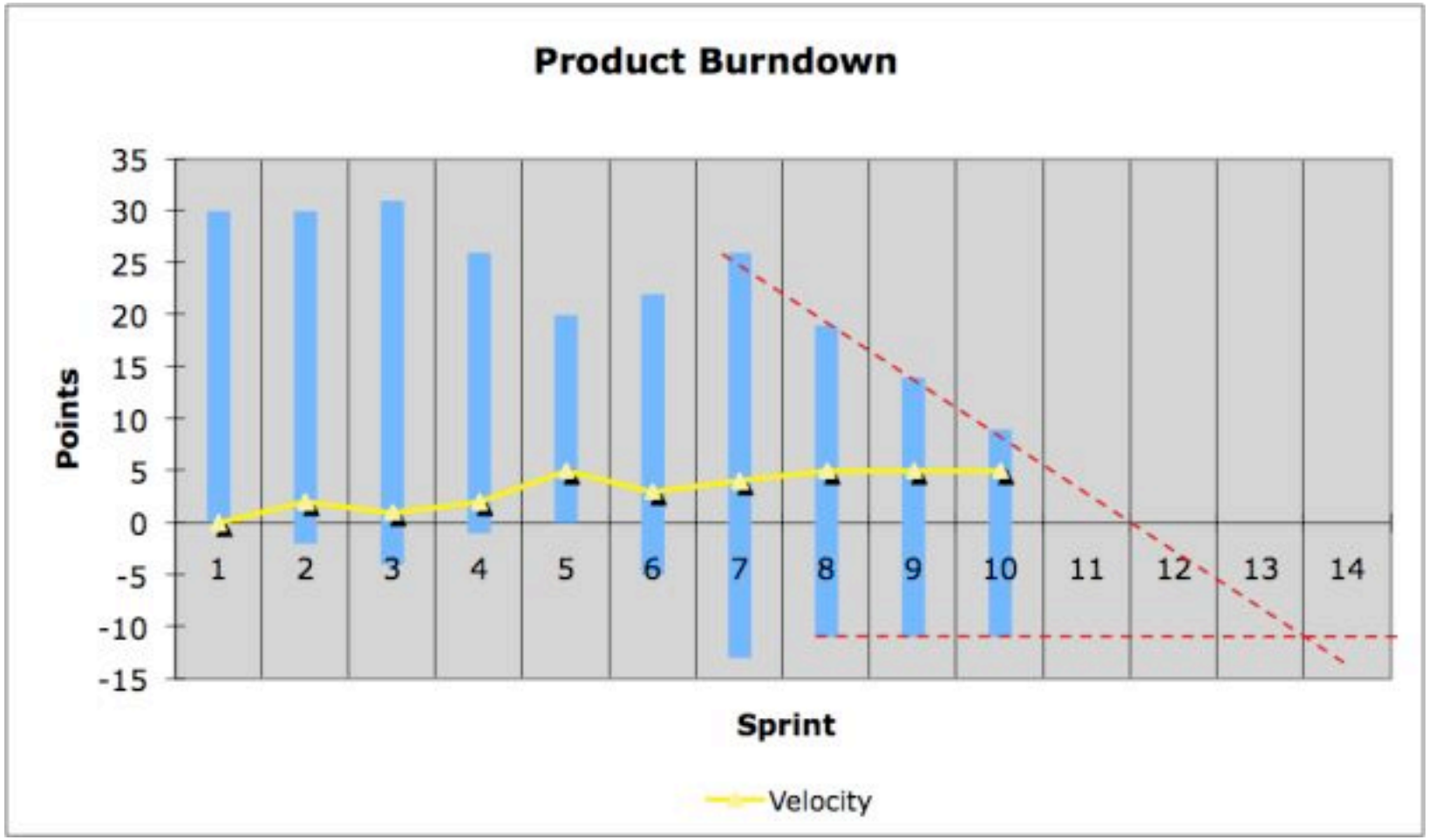
In the Simple Burndown Chart, the Velocity of the Scrum Team and the change of the scope cannot be visualized accurately. **To increase this lost accuracy and visibility, Scrum Teams use another type of diagram, which we call "Extended Burndown Chart".**

Extended Burndown Chart uses a bar chart instead of a line diagram. The size of each bar represents the total number of remaining user stories at the beginning of each sprint. The Velocity of the Scrum Team is subtracted from the top bar, while changes of the Product Backlog are presented at the bottom of the bar.



Extended Burndown Chart
Separating Velocity and Scope Changes

To get even more accurate results with the Burndown Chart, we can also take the rate of changes in total work into account. We call this more precise model "Extended Burndown Chart With Prediction". However, we have to be careful when using this model. The magnitude of changes in the Product Backlog will be relatively higher at the beginning. And yet, the rate of changes will usually drop, and they approach zero towards the end of the project.



Extended Burndown Chart with Prediction

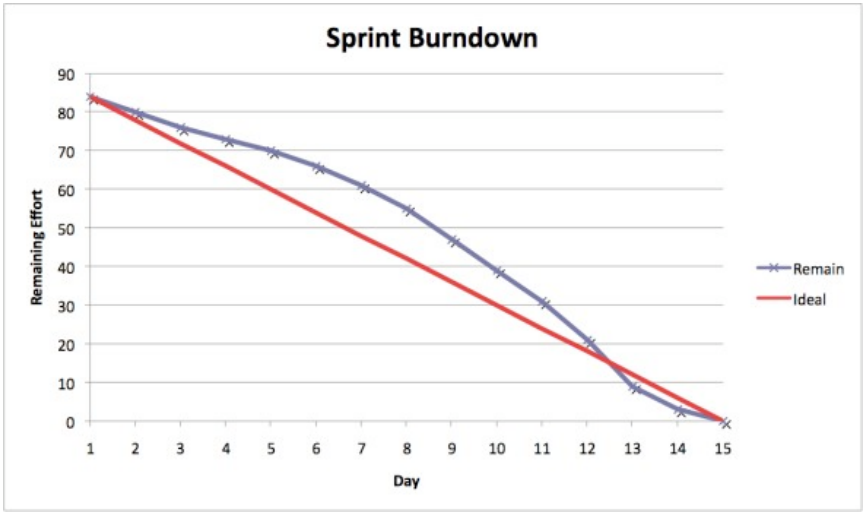
SPRINT BURNDOWN CHART (SPRINT BURNDOWN REPORT)

The Sprint Burndown Chart (Sprint Burndown Report) visualizes the progress within the Sprint towards reaching the Sprint goal.

It enables transparency and actionable progress data about the actual performance (burndown rate) of the Scrum Team. That allows the Scrum Product Owner and the Scrum Team to easily monitor how the Sprint is going. Thanks to this overview delivered by the Sprint Burndown Chart, the team can predict if the Sprint goal can be accomplished until the end of the Sprint on time.

Otherwise, the Scrum Master and the Scrum Team should consider and implement other measures to speed-up the execution of the remaining tasks and user stories in the Sprint Backlog.

In the Sprint Burndown Chart, the initial Sprint Backlog defines the start-point for the remaining efforts. Every day the remaining effort which needs to be completed until the end of the Sprint is summed up, and it's logged on this graph.



Sprint Burndown Report/Chart

It’s worthwhile to remember that; While a new Scrum team is forming, the performance is often not as good as how the ideal burndown rate envisioned it. That usually happens due to wrong estimates or unforeseen impediments that have to be removed to bring the Scrum Team on full speed.

SPRINT PLANNING MEETING

During the Sprint Planning Meeting, the Scrum Team builds a viable Sprint Backlog, which determines the user stories and tasks the team is going to implement until the end of this Sprint (Product Increment).

A Sprint Planning Meeting constitutes two parts, namely the WHAT-Meeting, and the HOW-Meeting. As those names imply, the WHAT-Meeting focuses on the scope of a Sprint, whereas the HOW-Meeting focuses on how this scope will be implemented.

During Sprint Planning Meetings, the presence of the Scrum Product Owner is mandatory. So she can answer questions from the Scrum Team and bring clarifications for the requirements and their acceptance criteria.

Stakeholders such as end-users or line managers can join Sprint Planning Meetings as view-only audiences. They're not allowed to influence the flow of the Sprint Planning Meeting.

The Sprint Goal

The Scrum Product Owner defines the Sprint Goal. It is a concise and realistic description of what the Sprint will attempt to achieve for business, end-users, and IT systems.

The Team Capacity

The total capacity of the Scrum Team might change from Sprint to Sprint. **To make realistic commitments, the Scrum Team needs to know its capacity for the upcoming Sprint.**

The team needs to take vacations, public holidays, time spent on Scrum Rituals, and a small contingency for unforeseen issues into account to propose a reasonable capacity.

The WHAT-Meeting

A Sprint Planning Meeting starts with a WHAT-Meeting. It requires some preparation:

- The Scrum Product Owner defines the Sprint Goal.
- Based on this goal, the Scrum Product Owner chooses the candidate user stories for the Sprint from the Scrum Product Backlog.
- These user stories are edited and broken into smaller user stories when necessary so that they can be processed within one Sprint.
- The user stories are estimated and prioritized.
- The Scrum Team plans its capacity for the upcoming Sprint.

The duties of various Scrum roles during the course of the WHAT-Meeting part of Sprint Planning Meetings are the following:

- The Product Owner introduces the Sprint goal and her preselection of requirements, which should go into the product increment.
- The Scrum Team identifies the required tasks and their estimations. It's the role of the Scrum Team to decide which and how many requirements preselected by the product owner they can confidently deliver in this Sprint.
- The Scrum Master moderates the meeting. She ensures that the self-organization and autonomous decision-making capability of the Scrum Team remain intact.

The HOW-Meeting

The goal of HOW-Meeting is to fill the Sprint Backlog by identifying the concrete tasks needed to implement committed user stories for the Sprint. These tasks usually (but not limited to) include activities such as analysis, design, development, testing, software build packaging, and documentation.

The HOW-Meeting can be done in a separate session after the WHAT-Meeting, or it may follow the WHAT-Meeting without a pause.

After identifying the tasks to be completed, the Scrum Team estimates these tasks. The unit for this estimation can be person-hours. Based on these estimates, the team has now its baseline about how long each user story will take them to deliver.

DAILY SCRUM MEETING / DAILY STAND-UP MEETING

The Daily Scrum Meeting is a maximum of 15 minutes meeting. These meetings take place on every working day at the same time in the same place.

It's best to conduct Daily Scrum Meetings with direct access to the Sprint Backlog and Sprint Burndown Chart. **So the Scrum Team can direct the Daily Scrum Meeting based on the facts and progress which are visible to everyone in the team.**

Daily Scrum Meeting aims to support the self-organization of the Scrum Team and identify impediments systematically.

All members of the Scrum Team, the Scrum Master and the Scrum Product Owner need to join Daily Scrums. Other stakeholders can also join these meetings, but only as a view-only audience.

Daily Scrum Meetings are structured in the following way. Every member of the Scrum Team answers three questions.

- **Daily Scrum Meeting - Question #1:**
  What activities have I performed since the last Daily Scrum Meeting?

- **Daily Scrum Meeting - Question #2:**
  What activities am I planning to perform until the next Daily Scrum Meeting? What is my action plan?

- **Daily Scrum Meeting - Question #3:**
  Did I encounter or am I expecting any impediment which may slow down or block the progress of my work?

The Scrum Master has to moderate Daily Scrum Meetings.

She needs to ensure disciplined and fast-pacing progress so that all team members can answer these three questions in at most 15 minutes.

The goal of Daily Scrum Meetings does not include to give time-consuming decisions or solve problems.

On the other hand, no issues or concerns from any Scrum Team member should be ignored or undermined due to the time constraint of the Daily Scrum Meeting. Concerns associated with specific user stories must be clearly articulated, discussed, and resolved after the meeting with the Scrum Team members related to these user stories.

To align on decisions and solve problems, the Scrum team can organize separate on-demand basis meetings. These separate meetings to focus on decisions or issues take usually place subsequently after Daily Scrum Meetings.

The Scrum Master documents the identified impediments and its dates in a separate log, flip chart or report which is accessible to the team. We find it very beneficial to quickly update the status of all known impediments at the very end of Daily Scrum Meetings.

SPRINT REVIEW MEETING

The Sprint Review Meeting happens at the end of the Sprint. Regardless of the duration of the Sprint, it usually takes one to two hours. Depending on the type of software and available infrastructure, the Sprint Review Meetings take place in a meeting room, in a lab or in the room of the Scrum team.

The goal of the Sprint Review Meeting is to review the shippable product increment built during the Sprint and bring transparency to the product development process.

The Scrum Team members do not need to prepare a Powerpoint or Keynote presentation to show in the Sprint Review Meetings. **They should instead spend their energy on the successful completion and demonstration of the Sprint outcome, which we call the shippable product increment.**

The Scrum team demonstrates its work results. The Product Owner controls whether the Scrum team delivered the requirements they had committed during the Sprint Planning Meeting accurately or not.

The Sprint Review Meeting enables the Product Owner to inspect the current status of the product and adapt the goals of the subsequent Sprint. **Therefore, Sprint Review Meetings are another way of formal and practical application of "inspect and adapt".**

Participants of the Sprint Review Meeting are the Scrum Team, the Scrum Product Owner, the Scrum Master. Optionally all other stakeholders such as end-users, clients, business representatives can join Sprint Review Meetings.

In Sprint Review Meetings, everyone is allowed to deliver their feedback about the demonstrated product increment.

In this way, the Scrum team has the opportunity to get unfiltered and direct input from stakeholders for whom they're building the software.

SPRINT RETROSPECTIVE MEETING

The goal of the Sprint Retrospective Meetings is to improve the teamwork of Scrum Team members and the application of the Scrum process.

The Sprint Retrospective Meeting happens directly after the Sprint Review Meeting, and it closes the Sprint.

Therefore, Sprint Retrospective Meetings lead to an increase in productivity, the performance of Scrum Team members, and the quality of engineered software.

The Sprint Retrospective is an integral part of the "inspect and adapt" process. Without this meeting, the Scrum Team will never be able to improve its overall throughput, and they cannot focus on the improvement of team performance.

Remember this: What you focus on is what becomes better!

A Sprint Retrospective Meeting is virtually a mini-post-mortem to define improvement potentials and remove process-related obstacles. Some examples of such improvement potentials that we frequently see in our projects are:

- Adoption of agile software development practices such as **extreme programming (XP)** or **test-driven development (TDD)**,
- Identification of new team norms and principles, or
- Increased availability of the Scrum Product Owner.

Without exception, the Scrum team conducts Sprint Retrospective Meetings at the end of every Sprint.

Only in this way, these meetings enable the Scrum Team to systematically adapt and improve their use of the Scrum process to the specifics of their project.

SCRUM GROOMING (BACKLOG REFINEMENT) MEETING

Scrum Backlog Refinement (Grooming) Meetings aim to keep the Product Backlog up to date. So the Product Backlog reflects the best know-how and understanding of the Scrum team about the ongoing Scrum project.

Scrum Backlog Refinement meetings can happen on-demand or scheduled basis up to two times a week, 30 minutes each session. The Scrum Team, the Scrum Product Owner, and the Scrum Master participate in these meetings.

During Scrum Backlog Refinement (Grooming) meetings, the participants fine-tune the Product Backlog with the following actions:

- Add new user stories based on newly discovered requirements.
- Remove user stories which are no longer required for the product.
- Fine-tune estimates of user stories.
- Update priorities of user stories.
- Split giant user stories into digestible smaller user stories, and prioritize them accordingly.

Here are our other suggestions to improve the outcomes of Backlog Refinement Meetings:

- **Ensure Cross Team Participation**, which means you identify the dependencies as early as it is possible,
- **Size User Stories Correctly**, which means all user stories can result in a shippable product increment within one single Sprint,
- **Prioritize User Stories**, which means you deliver immediate value to end-users, enable quick wins, and make them happy,
- **Estimate Like a Pro**, which means you obtain actionable inputs for reliable release planning,
- **Identify Dependencies**, which means you pull required teams and resources on board to make your project a success,
- **Uncover Risks**, which means you avoid tedious surprises during the later stages of your project.

SCALED SCRUM FRAMEWORK (DISTRIBUTED & LARGE SCRUM PROJECTS)

The Scrum Framework - as described so far - works best for a single Scrum Team in one location. However, in reality, a singular Scrum Team often cannot implement a project entirely, or the team members have to spread over multiple locations.

As a consequence, the number of teams has to increase with various distributed teams. **In many instances, we have also been observing that those teams are distributed in geographically distant locations or continents.**

There are numerous reasons which motivate organizations to distribute their teams across different locations:

- **Technical Reasons:** Some knowhow to build separate components of the software are not locally available in the headquarters,
- **Expertise Reasons:** Some capabilities related to the execution of different software engineering activities are not locally available. For instance, test automation, user interface design, or integration of in-house software to the software of other vendors can require experts outside the headquarters,
- **Size-related Reasons:** The project takes more people on board to deliver it to its clients in a predefined timeline. If this is the case, then the project organization will need more members than it can conceivably fit one single Scrum Team. So the Scrum Team has to be distributed,
- **Business-related Reasons:** Use of human resources from lower-cost locations or enabling the continuity of work by using engineers from different time-zones could build a good business case.

As communication is an integral part of the Scrum Framework, all Scrum Team members should pay attention to overcome the challenges to deal with working within a distributed project environment. **Furthermore, all team members should have access to communication tools,**

including audio/video conferencing and screen sharing tools.

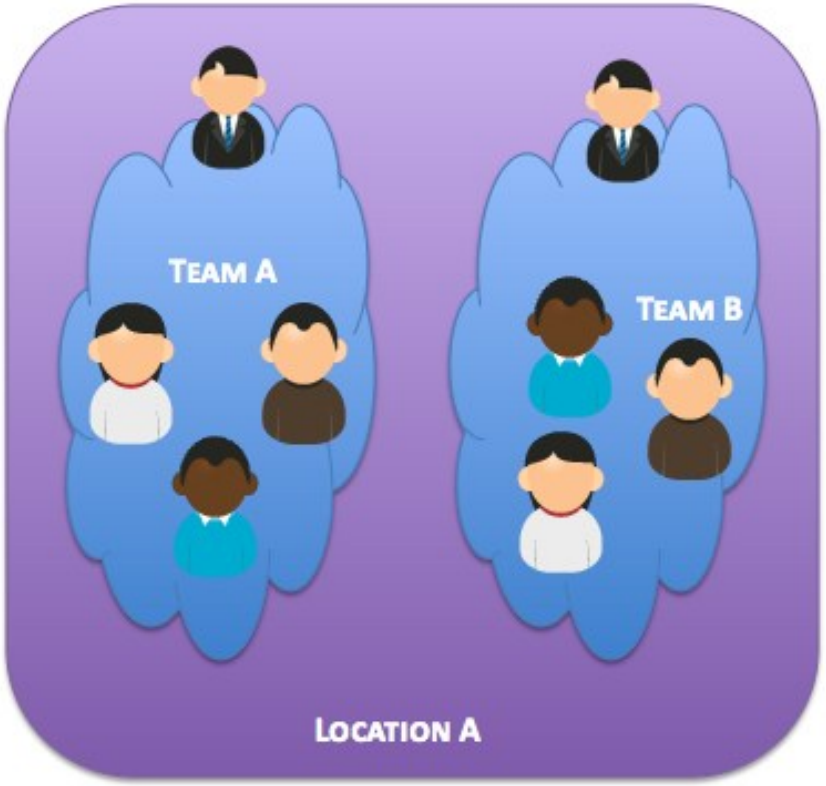
These commonly used project management tools support teams to enable healthy and continuous communication. Those can include product backlogs, sprint backlogs, incident lists, knowledge/news sharing tools, and so on.

Project Organization: Multiple Teams

The simplest way of expanding the Scrum Framework while working in a larger-scale project setup is to increase the number of teams in the same location.

If multiple teams need to work together to implement a project, it is best to grow the number of teams progressively.

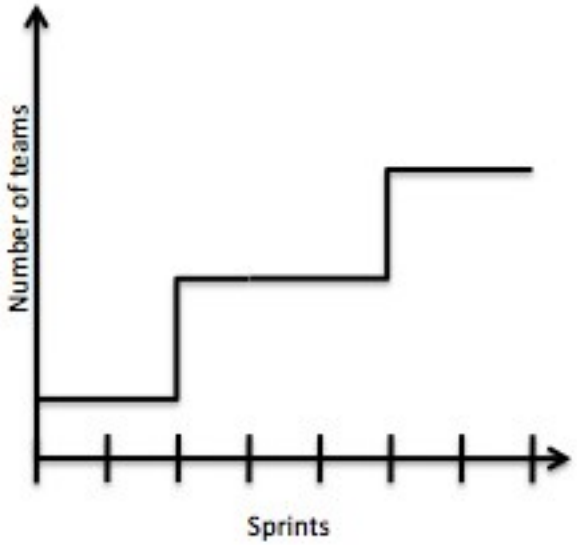
What does this mean to you?



Multiple Teams in a Single Location

In most organizations, progressive growth is more manageable than launching ten different new teams in one go. The best practice is to start with a single Scrum Team. After a few successful Sprints, one or two additional Scrum Teams can join the project. Once you ensure that these

multiple Scrum Teams work together well, you can keep on adding further Scrum Teams to your distributed project organization.



Increasing the Number of Teams

There are two typical ways of creating new Scrum Teams:

- You split an existing Scrum Team into multiple teams and add new Scrum Team members where and when necessary,
- You construct a new Scrum Team from completely different engineers who haven't involved the project so far.

Splitting an existing Scrum team has the advantage of leveraging the Scrum Team members who are already knowledgeable and who have already experienced with the ongoing project.

Therefore, those new teams are usually at least at some degree productive as soon as they're formed. The major drawback of this scenario is that the existing and fully functional Scrum Team has now been split into two teams. That could always cause some issues with the motivation of Scrum Team members. **Especially if those changes are happening without an in advance announcement and justification from senior leadership, and when the team members are mentally and technically unprepared.**

When adding completely new teams, these already existing teams can continue with their Sprints without any interruption and extra integration effort. However, it will take longer to

build up the necessary knowhow and momentum to ramp-up the entirely newly formed Scrum teams.

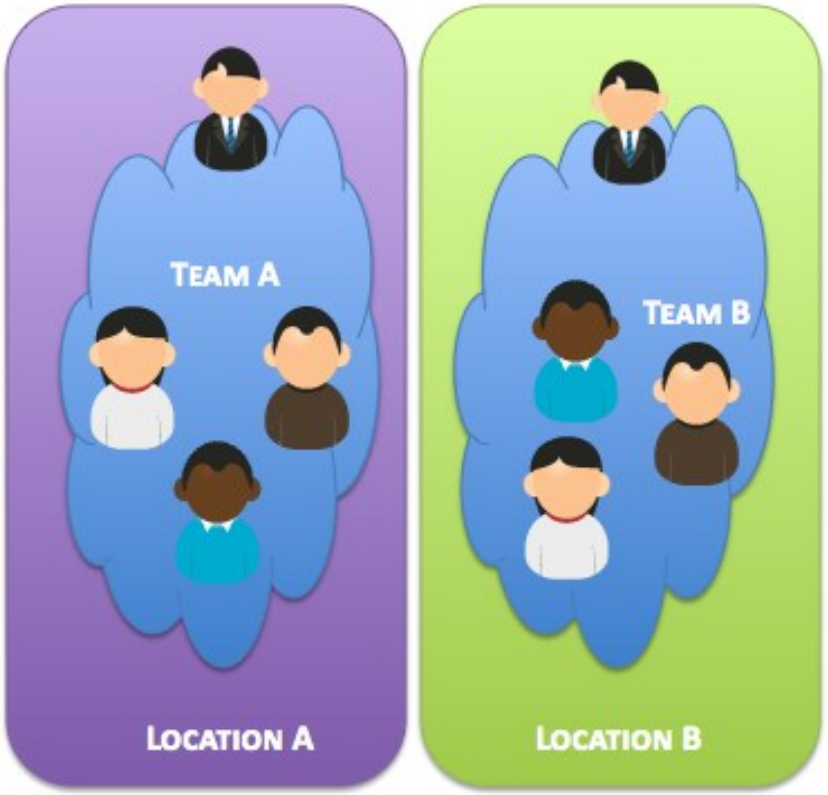
Independent from the decision on how you add new Scrum Teams to your organization bear in mind the following principles:

- Start with a small number of Scrum teams,
- Increase the number of teams gradually,
- Ensure the continuity of work and smooth delivery of software and business value during the times of change and growth,
- If there're significant problems that hinder productivity and continuity of work, first focus on fixing them rather than the expansion with new Scrum Teams.

Project Organization: Distributed Teams

The major complexity of multiple teams manifests itself when the new Scrum Teams have to be distributed over various locations.

Communication barriers between people, coordination difficulties of work, and misunderstandings of joint project norms across teams are only a few of many when it comes to mentioning this complexity.



Multiple Teams in Multiple Locations

The consequences of not addressing these challenges are severe.

Companies have to count billions of dollars of wasted IT budget because of the lack of their skills in **Organizational Leadership** and **Scaled Scrum Expertise**.

There are four critical suggestions for you to cope with these challenges:

1. You ensure that new Scrum Team members are trained in the Scrum Framework as a **Scaled Scrum Expert**,

2. You ensure that new Scrum Team members are introduced to the project adequately, so they have a proper understanding of what they're serving for. Not only technically but also from a professional business value point of view, so they can make decisions in their work to increase the value of their contribution,

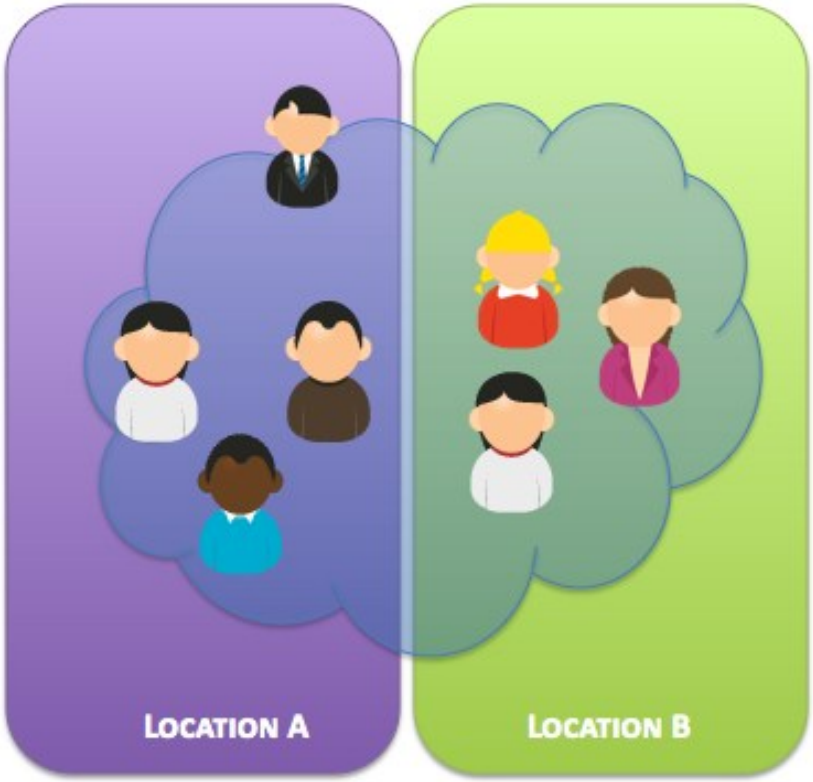
3. You ensure that the project norms are established. Similar to a single Scrum Team, which has its norms of how to communicate, how to plan, how to get the work done, a multiple project team organization should have its higher-level norms too. So these teams can communicate, plan, operate, solve problems, and deliver client and business value together.

4. You ensure that the new team members do at least temporarily work together with the experienced project members. That could require remote site visits and on-the-job-training. That's totally fine and even desired. Thanks to this approach, the knowhow can be smoothly transferred, and the two-ways and personal dialog between people in different teams and locations can be established.

Virtual Teams

Another option of a distributed Scrum Team is having its members spread over multiple locations. Such a team is called a "Virtual Team".

The main challenge here is to ensure flawless communication among the team members. Scrum Team members must still need to conduct all Scrum Rituals (Scrum Events) to coordinate their work, but now they have to do this while not all of them are present in the same room.



Virtual Teams

Scrum Team members colocated in the same location should still work together in the same room. And yet, they now have to rely more on the use of collaboration and communication tools. They can join the Scrum Events from the same meeting room to connect to the other half of the virtual team via video conferencing technologies.

Scrum Product Owner Team

As we have covered many times in this material so far, regular communication between the Scrum Product Owner and the Scrum Team is crucial for the successful delivery of a project.

We need to ensure that the Scrum Product Owner is always available to Scrum Teams located in different locations. **Therefore, it is often necessary to have multiple Scrum Product Owners working together. Ideally, there is one dedicated Scrum Product Owner for each team.**

The Scrum Product Owners should then build a dedicated "Scrum Product Owner Team" to work

together effectively. One of the Scrum Product Owners should be assigned to the role of the "Chief Scrum Product Owner".

He or she is responsible for ensuring that:

- The correct product is built to satisfy the demands of its client,
- All Scrum Product Owners collaborate efficiently, and they enable their teams to build the business and technical value for their clients.

Since the Scrum Product Owner Team is responsible for the complete requirement engineering, it is beneficial to have other competencies and stakeholders in this team. Those can include the representatives of the business case, relevant stakeholders, enterprise architects, and technology architects.

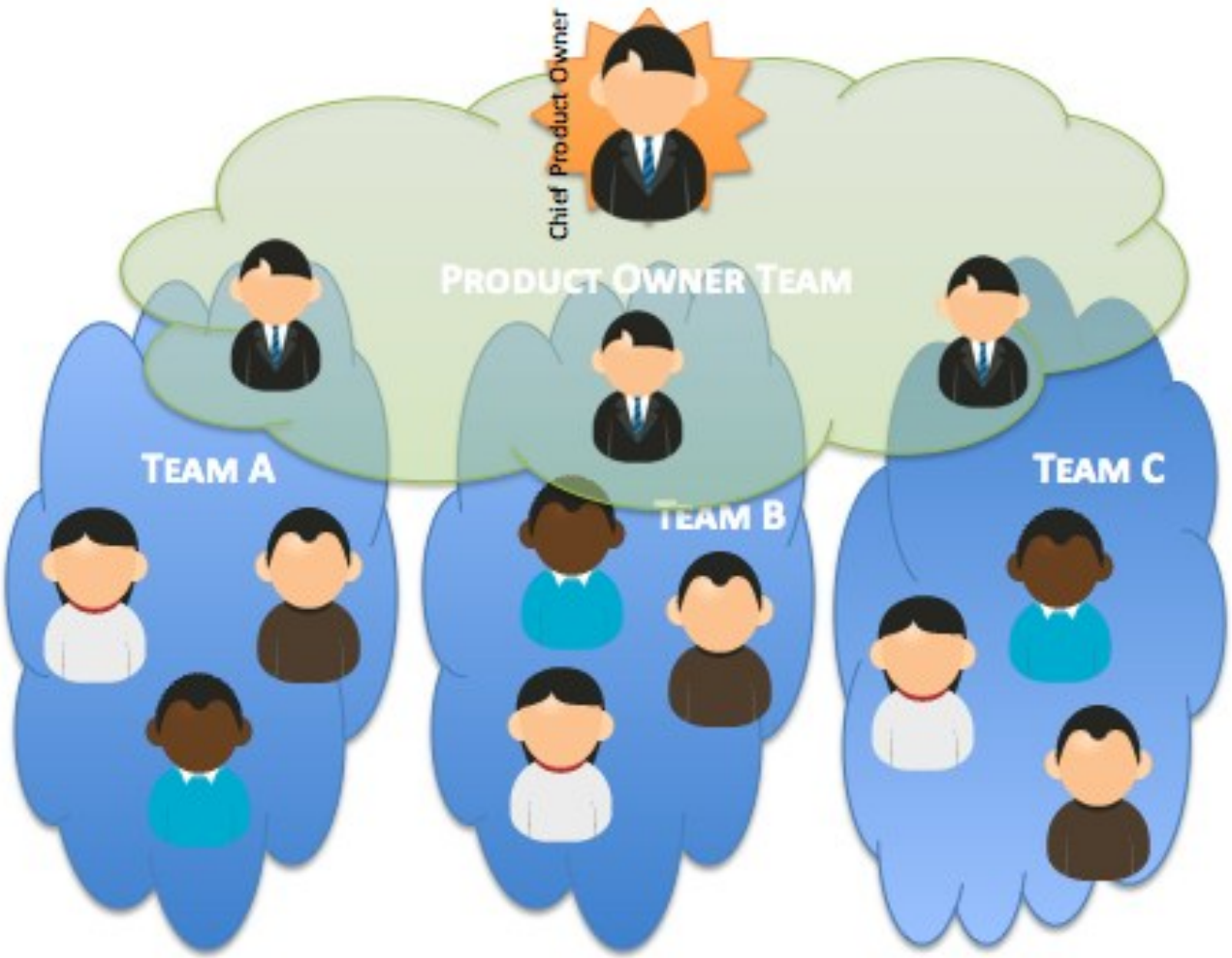
All Scrum Product Owners should work within a single large Scrum Product Backlog containing all stories relevant for the project. Each Scrum Team is responsible for delivering some of these user stories. And yet there will be still instances of specific user stories that require the attention and deliverables from multiple Scrum teams.

Component vs Feature Teams

When distributing work among different teams, we can make the teams accountable for specific software components or features. That is why we call them "Component Team" or "Feature Team."

Component Teams

When using Component Teams, each team is only responsible for the implementation of dedicated components from the overall system. To finish a user story, it is usually necessary to split the user stories into smaller pieces to implement them within a single component. The dependencies between the components of these Component Teams make continuous integration an inevitable part of successful deliveries.



Scrum Product Owner Team

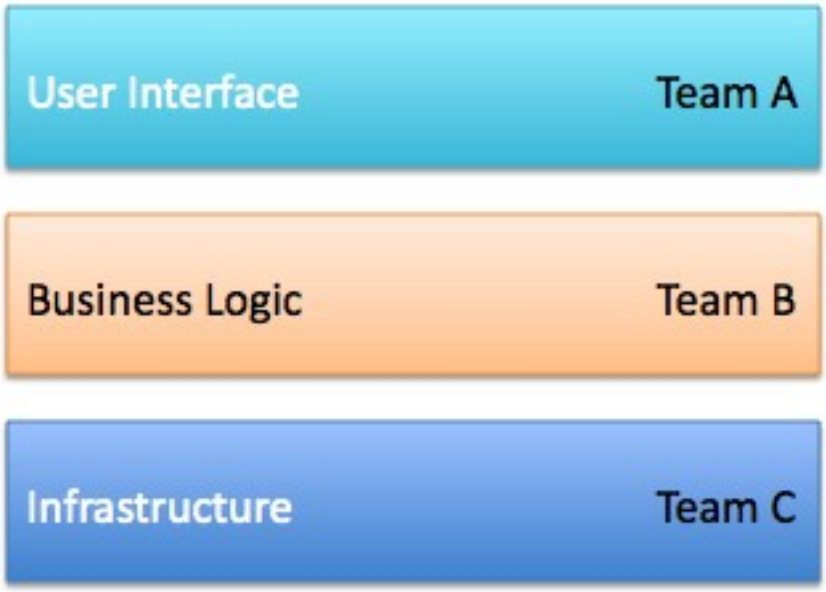
Thus, a feature cannot be usually delivered within a Sprint because its implementation depends on the deliverables from user stories of other teams.

That results in increasing batch sizes and lead times of ongoing, not yet integrated work. **That doesn't sound so good, because Scrum Teams should target delivering shippable software increment in smaller batch sizes and shorter lead times.**

The advantage of component teams is that they make it easier to focus on and build expertise about architectural and design details of particular components. **That could be massively beneficial for components that require discovery and innovation.**

On the other hand, the members of component teams do only specialize in individual components of the whole system. **They could lose their bird-eye view and business necessity of features.**

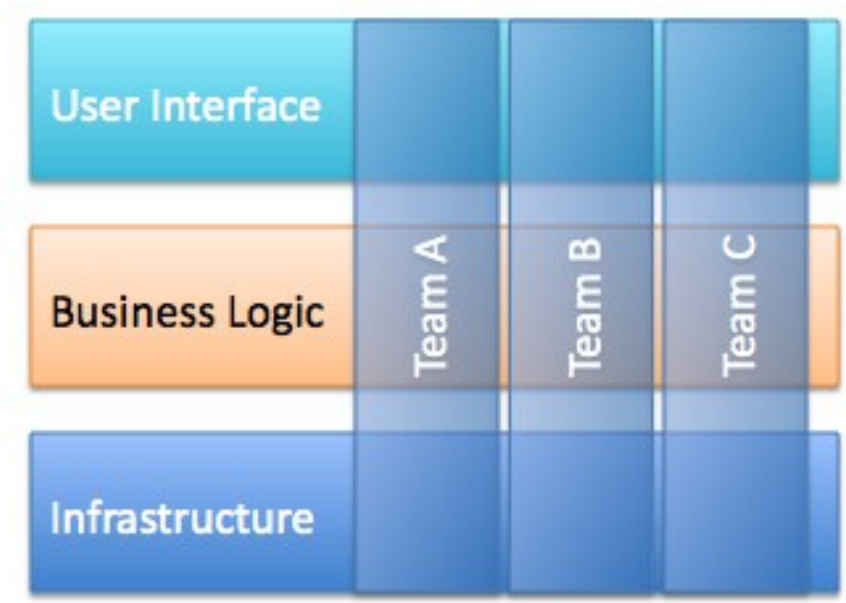
Keep in mind that our clients do not compensate us to deliver components, but features with which they will execute their businesses. Without this relentless focus on features, overall optimization, and integration of software might take extra time. Since decisions of component teams tend to optimize single components, those decisions can construct invisible bottlenecks for the success and performance of the overall solution.



Component Teams

Feature Teams

Feature teams are fully responsible for the implementation of user stories as they're specified within the Product Backlog. The teams do no longer need to be divided for various components. Each Feature Team is responsible for delivering a fully-functional feature and a business value associated with this feature.



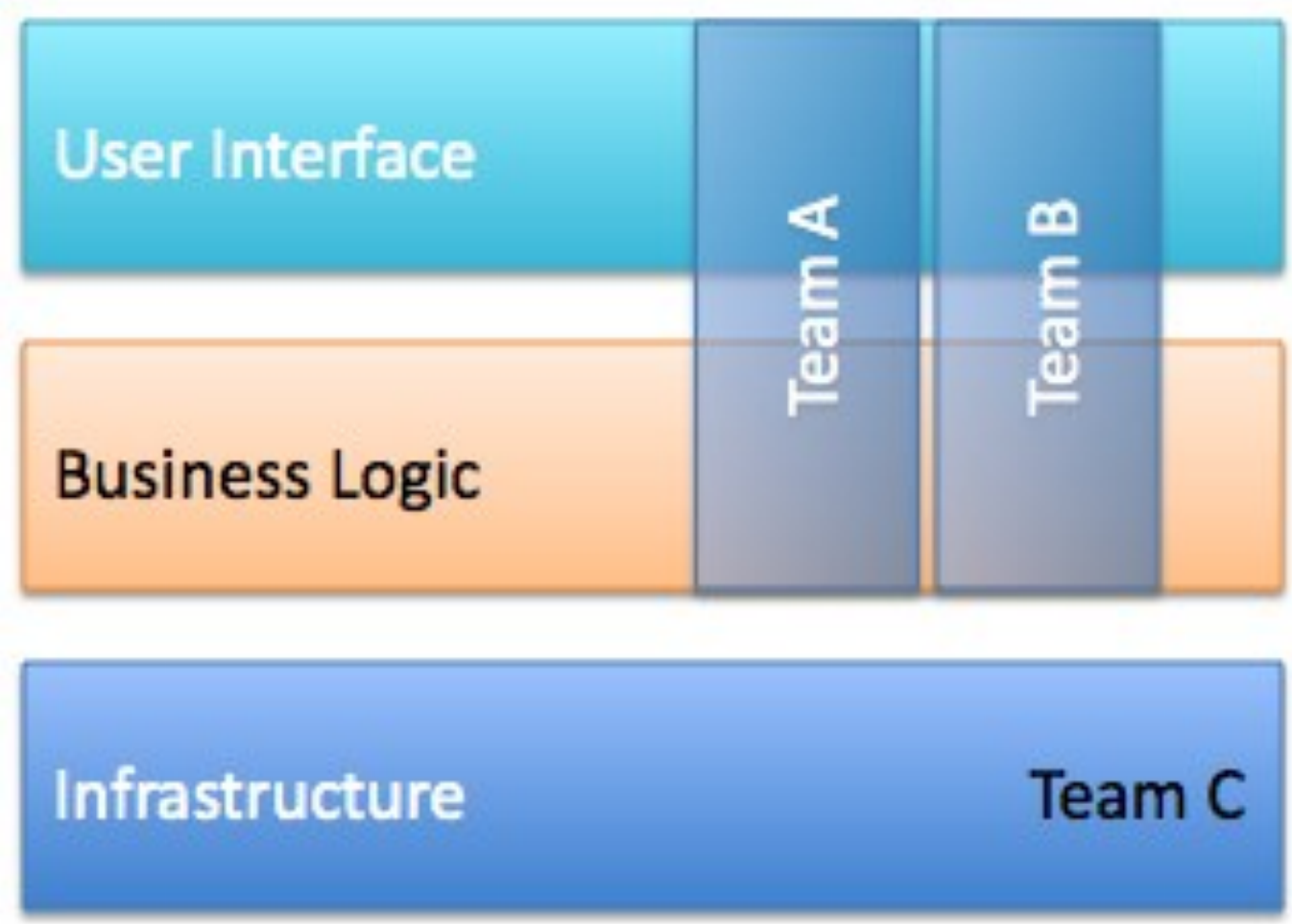
Feature Teams

Members of feature teams possess cross-functional skills. They act as autonomous as it is possible to deliver fast. **The advantage of feature teams is that the team maintains the system-knowledge, and this makes it easier for them to integrate their features with the rest of the system.**

However, for feature teams, it may become more challenging to build sufficient knowhow about components. Furthermore, bringing up an autonomous feature team that can deliver fast and independently takes time as building an interdisciplinary functional team is not that easy. And yet, these are the high-performer teams which get the job done in most organizations, probably including yours.

How Do We Choose Component Teams vs Feature Teams?

In practice, most of the large organizations use both dedicated Component Teams and Feature teams too.



Component and Feature Teams

Team C, on the chart, is a Component Team. It provides planned and on-demand infrastructure services to other teams that function as Feature Teams. **Team C does not directly implement end-to-end user stories per se. They deliver the requirements of the user stories committed by the Feature Teams.**

That allows the minimization of the number of qualified people in Feature Teams with the knowhow of those components.

The Scrum Master In The Distributed Project Environment

In a distributed project environment, the role of the Scrum Master is even more essential. In those project configurations:

- There will be extra effort required to align the teams on the values of the Scrum Framework,
- It will take longer to establish individual team and project norms (standards) which influences numerous teams,
- Last but not least, there will be many impediments due to the increased number of dependencies between teams and their deliverables.

One important rule to bear in mind is that the Scrum Master should physically locate where his or her team is. Otherwise, it will be almost impossible for the Scrum Master:

- To remove the impediments for his team,
- To Establish their norms, and
- To help them to improve their use of the Scrum Framework.

The best practice is to have a Lead (Primary) Scrum Master to guide the overall use of the Scrum Framework across multiple teams.

In other unit Scrum teams, which form the larger Scrum organization, someone should be acting as a local Scrum Master too.

SCALED SCRUM FRAMEWORK (MULTI-TEAM COORDINATION & PLANNING)

Scrum of Scrum

After having seen the last chapter of this course, the next logical question in your mind could be how you do coordinate those different Scrum Teams. So they do work together efficiently.

That's a fair question, and we have attempted to cover this answer too.

Scrum of Scrum Meetings

Scrum of Scrum Meetings resemble Daily Scrum Meetings. **And yet, here during Scrum of Scrum Meetings, the focus is not the work of individual Scrum Team members, but the Scrum Teams themselves.**

Scrum of Scrum Meetings do take place every day, and they are limited (timeboxed) to 15 minutes too. And yet depending on the complexity of the project, especially during its early stages while Scrum Teams are just forming, these meetings can take 30 to 60 minutes. That's totally fine as well.

Each team sends out one of its Scrum Team members (usually its local Scrum Master) to participate in Scrum of Scrum Meetings. And yet, the teams can also choose to rotate their representative daily or weekly basis depending on their discretion. Each participant from a team answers the following three questions:

- What did the team do yesterday?
- What is the team planning to do today?
- Are there any impediments to hinder or slow down the progress of the team?

These answers should obviously cover the user stories and interdependencies, which impact other teams too.

The Chief Scrum Product Owner and the Lead Scrum Master can jointly moderate Scrum of Scrum Master meetings. Alternatively, one of them can take over the moderation duty of these meetings, or they can choose to rotate this duty among themselves as well.

Common Sprint Review Meetings

Common Sprint Review Meetings with the participation of all Scrum Teams are not mandatory, but they could be very beneficial. Note that Common Sprint Review Meetings do not replace Sprint Review Meetings the Scrum Team conduct locally.

The participants of Common Sprint Review Meetings are the delegates from Scrum Teams and/or their respective Scrum Product Owners.

The Scrum Teams can also rotate their delegates based on their preferences. **The Lead (Primary) Scrum Master carries the responsibility of moderating a Common Sprint Review Meeting.**

Common Sprint Review Meetings enable all Scrum Teams to demonstrate their Shippable Product Increments to the Chief Scrum Product Owner and all other Scrum Product Owners.

In this way, the Common Sprint Review Meetings fulfill two purposes:

1. All Scrum Teams are now aligned about the current status of the overall project.
2. All Scrum Teams collect feedback for their work, and they have the chance now to take this feedback into account, while they do their upcoming Sprint Planning Meetings.

Common Sprint Retrospectives

Similar to Common Sprint Review Meetings, Common Sprint Retrospective Meetings are not mandatory, but they could be very beneficial.

Note that Common Sprint Retrospective Meetings do not replace Sprint Respective Meetings the Scrum Team conduct locally.

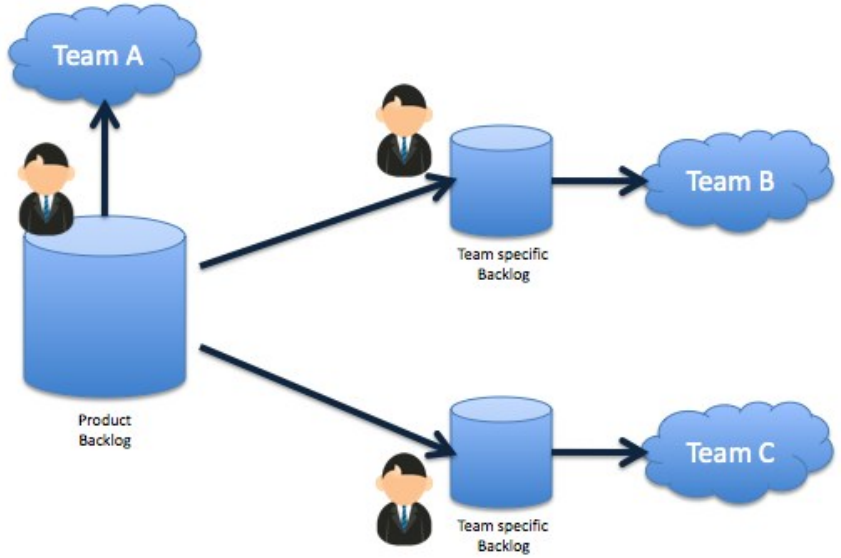
The participants of Common Sprint Retrospective Meetings are the delegates from Scrum Teams. The Scrum Teams can choose to rotate their delegates based on their discretion.

Common Sprint Retrospective Meetings are led by the Lead (Primary) Scrum Master. These meetings aim to find out and act on improvement potentials about how the larger Scrum project organization uses the Scrum Framework.

All issues which require the attention and collaboration of multiple Scrum Teams to resolve should be highlighted in these meetings. Their paths towards resolution need to be planned, scheduled, and followed-up.

Multi-Team Planning: The Global Scrum Product Backlog

When working with multiple teams, it is essential to manage a Global Scrum Product Backlog, which contains the user stories of all Scrum Teams. The Chief Scrum Product Owner could govern the Global Scrum Product Backlog. Yet, its contents are maintained by all Scrum Product Owners.



Team-Specific Backlogs

When necessary, the user stories from the Global Scrum Product Backlog can be broken down into more team-specific user stories.

These more detailed user stories are maintained in a Local Scrum Product Backlog. References

from Local Scrum Product Backlog to Global Scrum Product Backlog should be present. **These references will help the Scrum Teams to see what roles their user stories play in the bigger picture of their project, and what kind of client value they're delivering.**

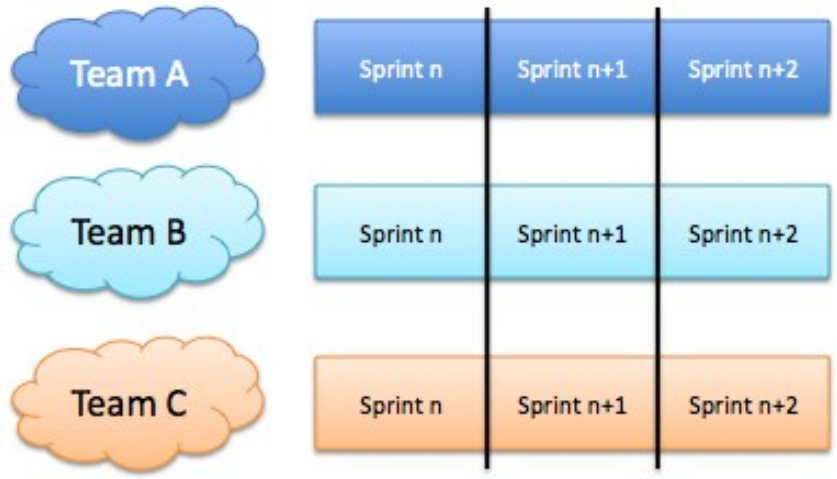
Sprint Scheduling

In a distributed Scrum project environment, there are two options for how you can choose to synchronize the work of different Scrum Teams.

- Synchronous Sprints
- Asynchronous Sprints

The first option is to use Synchronous Sprints. With Synchronous Sprints, all teams start and end their Sprints on the same day.

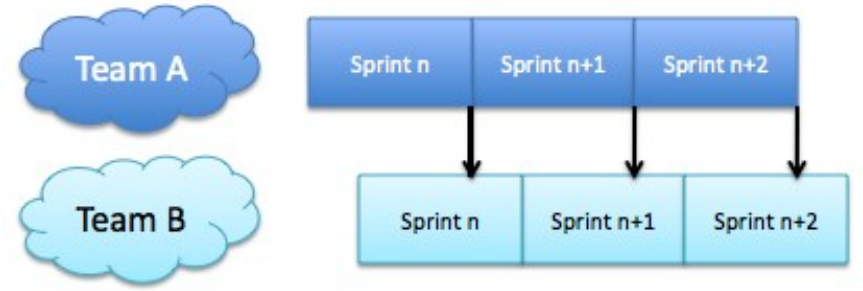
Synchronous Sprints are usually the preferred approach since they make communication and coordination of the Scrum Team relatively easier.



Synchronous Sprints

Another option is to use Asynchronous Sprints. With this option, the Sprints do not start and end on the same day. Using Asynchronous Sprints has the advantage that not all Scrum Rituals of individual Scrum Teams must take place on the same day. So it makes for the Chief Scrum Product Owner and other Scrum Product Owners possible **to participate Sprint Planning, Sprint Review, and Sprint Retrospective Events of other Scrum Teams and support them when they're asked to do so.**

When one team provides services to other teams, asynchronous Sprints bring an additional advantage.



Asynchronous Sprints

Here is a great scenario to clarify this, which was depicted on the above sketch: The work of Team-A (Supplier Team) needs to be integrated into the deliverables of Team-B (Master Team). With the help of Asynchronous Sprints, Team-A can close its Sprint before the Team-B does. So, Team-B (Master Team) can pick the deliverables from Team-A (Supplier Team) and integrate them into their work before they close their own Sprint.

Effort Estimations

All Scrum Teams within the distributed Scrum Project Environment need to use the same unit (Fibonacci Numbers or Shirt Sizes, etc.) to conduct their estimates.

Similarly, the Global Scrum Product Backlog should adhere to this agreed unit of effort estimations too.

Special attention needs to be paid for the estimates of Component Teams. Components Teams do usually provide services for the user stories of Feature Teams. Therefore, they should be getting the necessary support and clarifications during their own Sprint Planning Meetings and estimations.

SCRUM RELEASE PLANNING

The goal of a release plan is to visualize high-level planning for multiple Sprints, usually between three to twelve Sprints, or so-called Product Increments.

A release plan becomes the guideline that reflects expectations from a Scrum Team about:

- Which features will be implemented,
- In what order and when these features will be implemented.

The release plan also serves as a benchmark to monitor and control the progress of a project. A release plan serves as a target for actual deployments of software in IT production systems in two ways:

- Deployment of "Milestone Deliveries" to create business value for the client before the project is complete. These Milestone Deliveries cover a subset of client requirements agreed by the Scrum Product Owner, client, and business stakeholders,
- Deployment of Final Delivery, which includes all known demands and feature requests from the client and business stakeholders.

Before a release plan is created, the following artifacts and information need to be taken into account:

- **A prioritized and estimated Scrum Product Backlog,**
- **The measured velocity** of the Scrum Team (The velocity is estimated, or its value should be extrapolated from the past similar projects if the Scrum Team is just forming),
- **Success criteria imposed by clients** such as schedule, scope, provided human resources allowed by the project budget).

Since a Release Plan is heavily associated with the Product Backlog, the Scrum Product Owner governs and maintains the Release Plans.

Depending on the demands and priorities of the clients, a release plan is created to satisfy one of these three goals:

- Feature-Based Release Planning
- Date-Based Release Planning
- Feature-Based and Date-Based Release Planning **(The Most Typical)**

Feature-Based Release Planning

What we know: Velocity of the Scrum Team, Features we want to deliver.
What we don't know: How long do we need to deliver these features?

Release Goals:
(1) First Release after user able to login & logout.
(2) Second Release after user can manage items
(3) Final Release when all stories are done

First Release
Stories: 7, 1 10 → Estimation = 5 Points
Estimation / Velocity = 5/3 → 2 Sprints

Second Release
Stories: 9,2,4,3,5 → Estimation = 15 Points
Estimation / Velocity = 15/3 → 5 Sprints

Final Release
All Stories → Estimation = 30 Points
Estimation / Velocity =30/3 → 10 Sprints

ToDo List

ID	Story	Estimation	Priority
7	As an unauthorized User I want to create a new account	3	1
1	As an unauthorized User I want to login	1	2
10	As an authorized User I want to logout	1	3
9	Create script to purge database	1	4
2	As an authorized User I want to see the list of items so that I can select one	2	5
4	As an authorized User I want to add a new item so that it appears in the list	5	6
3	As an authorized User I want to delete the selected item	2	7
5	As an authorized User I want to edit the selected item	5	8
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9
8	As an administrator I want to see the list of accounts on login	2	10
Total		**30**	

Velocity: 3 Points / Sprint

Releases after Sprint 2, 7 and 10

Release Plan For A Feature-Driven Project

For Feature-Based Release Plans, the sum of user story points of requested features within a release is divided by the team velocity. That is going to reveal the number of Sprints required to complete a Milestone Delivery or Final Delivery of the product. And we make the release plan accordingly.

Date-Based Release Planning

What we know: Velocity of the Scrum Team, The Date we want to deliver.
What we don't know: What features can we deliver until the deadline?

Release Goals:
Release every 6 weeks

ToDo List

ID	Story	Estimation	Priority	
7	As an unauthorized User I want to create a new account	3	1	
1	As an unauthorized User I want to login	1	2	
10	As an authorized User I want to logout	1	3	
9	Create script to purge database	1	4	
2	As an authorized User I want to see the list of items so that I can select one	2	5	*Release*
4	As an authorized User I want to add a new item so that it appears in the list	5	6	
3	As an authorized User I want to delete the selected item	2	7	*Release*
5	As an authorized User I want to edit the selected item	5	8	*Release*
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9	*Release*
8	As an administrator I want to see the list of accounts on login	2	10	
Total		**30**		

Velocity: 3 Points / Sprint **Sprint length 2 weeks**

Release Plan For A Date-Driven Project

For Date-Based Release Plans, we multiply the team velocity by the number of Sprints we have until the release date. That is going to reveal the estimated total number of user story points the Scrum Team can deliver until the release date. And we make the release plan accordingly.

Feature-Based and Date-Based Release Planning

What we know: Velocity of the Scrum Team, Features we want to deliver, The Date we want to deliver these features.
What we don't know: Can the Scrum Team deliver the requested features until the given deadline?

We multiply the team velocity by the number of Sprints we have until the release date. That is going to reveal the estimated total number of user story points the Scrum Team can deliver until the release date. If this number is larger than the sum of user story points of features within a release, then we're safe.

Otherwise, the velocity of the Scrum Team needs to be extended by adding extra human resources to the team. That may not be a viable option as the Scrum team could already possess 9 people, which is the upper limit of an ideal size of a Scrum Team. Then some user stories of the project need to be delivered by another Scrum Team, which is going to work with the original Scrum Team in parallel.

Similar to a Scrum Product Backlog, a Release Plan is not a static plan. It will change during the whole project while we know more about the project. New, removed, modified user stories, and the respective changes of their estimates will influence the release plans as well. Therefore, the release plan should be revisited and refreshed at regular intervals.

NEXT STEPS

How to Guarantee Your Position As A Successful Scrum Professional

I feel that it's now my job to inspire you to actually implement and execute what you have learned from this material.

Let's face it: The big, vast IT industry is not going to accommodate you with more opportunities and more business without you taking some serious initial steps. The IT industry most likely doesn't even know you exist; up until now, you only operated as a small part of it, or you're just getting started. The government is not going to bail you out on your difficult days, and they certainly are not going to help you to advance and conquer on your career journey.

Taking the time to pick up this book and read it suggests that you truly do want to do something different. For this, I acknowledge and congratulate you. Well done to you on getting this book. I applaud you for reading it and even more for finishing it. Now, if you want the world to give you a standing ovation, put lessons in it to work.

Interestingly, one of the most effective ways of perfecting these disciplines is to help others attain success and implement these actions themselves. When people with common goals and motivations come together, they tend to learn faster and become a support system for one another. So gather a group of like-minded and highly driven people who refuse to live by the norms of the mediocre. Assemble a study group to read this book and brainstorm it with you. Ask your co-workers, employees and bosses to read this book as a team. Then help one another apply and commit to using the actions, hold one another accountable to these commitments.

Follow now International Scrum Institute™ LinkedIn® Company Page to get connected to other like-minded professionals who can empower and inspire you in your career.

Something tells me that you didn't pick up this book because you are comfortable or satisfied with where you're in your career. Chances are you want to change or improve your current position. Otherwise, you wouldn't have finished this book. Therefore, I will be happy to support you on your career journey!

Why Should You Be Getting Your Scrum Certifications Today?

A Scrum certification is the testimony of your competence in the Scrum software development and delivery process.

Scrum certification acknowledges your demonstrated knowledge and outstanding expertise in the Scrum framework after a formal multiple-choice test evaluation.

Scrum software development process has been offering immense benefits to millions of professionals until today. **Therefore, there is no reason that you won't join these accomplished men and women who upgraded their careers and skills with the help of the Scrum framework.**

If you still wonder, I want to assure you that you can no longer imagine a growing career without possessing a Scrum certification. It's regardless of your role, title, and experience in Information Technology (IT) ecosystem. You even don't have to be an IT professional anymore to understand what Scrum is, how Scrum works, and get a Scrum certification.

Your are a Five-Star Professional!
You Must Be Acknowledged and Compensated Accordingly!

Whatever you do for a living, regardless you're part of an IT department or not, there is an essential and indisputable fact. **Your tasks and professional business value you've been serving for your organization are dependent on and interrelated to IT, software, and agile Scrum process and principles.**

Moreover, thanks to the shift of traditional business models into software as a service (SaaS) driven businesses or so-called digitalization movement, it's no longer a voluntary decision for any professional in or outside the IT department to get certified as a Scrum professional. However, it's a must today to get a Scrum certification.

You may be just starting your career, or you may be a seasoned IT professional. That doesn't play a role. You need to get your Scrum certification.

Your role may or may not include people and functional management activities. It doesn't matter too; you still need to have a Scrum certification.

You should learn Scrum software development framework and become an accomplished Scrum Professional today by getting your Scrum certification.

The Pros Of Being An Accomplished Scrum Professional

It's now the time to recap. If you didn't get a chance to read every word in this book, let me break down my thoughts. Here are my thoughts about the pros and cons of getting certified as a Scrum professional.

1. The Pros for Employees, Freelancers, Coaches and Trainers

- A Scrum certification will be your recognition of competence and up-to-date knowhow in the Scrum domain.
- A Scrum certification will help you outcompete your peer group who do not develop themselves anymore. And remember, they're a lot. It

will help you get hired for your dream job as a certified and accomplished Scrum professional.
- A Scrum certification will broaden your perspective, and it will further open up your mind for continuous learning. It will help you get more responsibilities and fantastic career opportunities.
- A Scrum certification will provide a brand new toolset with which you can deliver great products and services that your clients and employers would love.

2. The Pros for Organizations and Employers

- Scrum certifications will reduce costs by improving the efficiency of your teams, activities, and processes.
- Scrum certifications will help you win projects with your trained and skilled employees that you couldn't win otherwise.
- Scrum certifications will improve employee satisfaction and commitment by encouraging them to get trained and develop skills.
- Scrum certifications will improve the quality of your deliverables, customer satisfaction, and ultimately, the success and profitability of your organization.

Things To Remember After You Become An Accomplished Scrum Professional

- A Scrum certification shouldn't stop your learning. Don't forget that getting certified as a scrum professional is just the first step. In the spirit of "inspect and adapt" which you learned from the Scrum framework, it's still your duty and obligation to experiment, observe, and learn continuously.

- There is no one size fits all solution for all organizations around the world. The Scrum software development and delivery framework is no exception to this rule. What we observed

is that: Most organizations that we're unable to get the best performance out of the Scrum framework have a common characteristic. These are the organizations that failed to adapt Scrum to their own business and IT-ecosystems. Therefore, again in the spirit of "inspect and adapt", don't see the Scrum framework as a 100% guaranteed recipe for success. Please don't underestimate your cognitive ability to adapt it to the own dynamics of your business and IT. In fact, as a paid professional, this is what you're supposed to do to get the best throughput and business results by using the Scrum framework.

- Scrum didn't solve all the problems we have in our IT departments. Don't stop developing yourself with **Newly emerging software development and delivery processes such as DevOps**. To better understand the known flaws of the Scrum framework and how DevOps handles them, have a look at this top article at a later moment: **What Are TOP 6 Differences Between DevOps and Scrum? (DevOps vs Scrum Comparison)**

In conclusion, a Scrum certification is an excellent way to get started with agile software development and delivery practices.

According to a Gartner study **"Becoming a Better Scrum Master"** published in 2019, until 2023, 92% of companies worldwide, and 96% of companies in the United States will be adopting agile scrum practices.

Therefore, there will be no better time other than now for you to

- Start learning the Scrum framework and,
- **Get yourself certified as a Scrum Professional** with very affordable fees of International Scrum Institute™.

The only remaining question is, when are you going to get started?

Register Your Scrum Certification >>>

★★★★★ **Written by Johan Swart on November 5th, 2019**

Excellent company to further your studies. They are the number one in professionalism and that personal touch. All info is always fast, accurate and to the point with no hidden costs. I truly enjoyed the experience and will enroll for a few other courses as well. My thanks to the team.

Johan Swart, Programmes Manager

★★★★★ **Written by Diana Karen Muñoz Sierra on September 23rd, 2019**

Scrum Institute's Certification helped me get a good overview of what the Agile methodologies are and how the Scrum methodology in organizations are being implemented today. Now I'm the Scrum Master for two important projects.

Diana Karen Muñoz Sierra, Mobile Developer

★★★★★ **Written by Simone Davi on September 17th, 2019**

If you hold a firm understanding of the Scrum framework and Agile principles, but you haven't taken the step to the Scrum certifications yet, the International Scrum Institute™ has the perfect value for money for you. Let me add that the cool thing is that the certification test is for testing the real life experience and less on dry theory. I did it and I do recommend it. Thanks International Scrum Institute™

Simone Davi, Self-Employed

★★★★★ **Written by Prafulla Karale on September 9th, 2019**

This certification from Scrum Institute definitely added weight to my resume. I literally started getting calls from recruiters from Linkedin.
Thanks for International Scrum Institute™ for this amazing initiative. I will recommend this others and my friends to go for this certification and boost their career.

Prafulla Karale, Project Manager (Atos-Syntel)

See More Reviews >>>

THANK YOU

I would like to thank you again for taking the time to read The Scrum Framework. We hope that you enjoyed reading this book as much as we had enjoyed while we were writing it. It is our biggest pleasure if we by any means managed to help you build a strong Scrum foundation for yourself.

We know that it's a very complex, overwhelming and overcrowded world with all Scrum programs out there in the market.

And yet we managed to build our Scrum training and certification programs more concrete, attractive, helpful, useful and simpler than our competition did. This is why we believe our valuable students choose International Scrum Institute™ over bureaucratic, complex, expensive and half-baked solutions of our competitors.

To register and get started with your Scrum Certifications, click **Register Programs Today >>**

Yeliz Obergfell,
International Scrum Institute™