Scrum Primer

Agile development is a way of creating software that focuses on getting things done quickly and efficiently. Here's what it means in simple terms:

* **Agile** is all about creating working software as soon as possible, rather than spending a lot of time planning and writing long documents before starting.
* It uses **small teams** made up of people with different skills, and these teams can make decisions themselves, instead of relying on many layers of management.
* Agile works in **short cycles**, constantly improving the software based on feedback from customers during the process, instead of waiting until the end.

In summary, Agile is a fast, flexible approach to building software that involves teamwork and frequent updates.

**1. Build working software quickly**

Instead of spending months planning, Agile focuses on **creating something usable fast**. Imagine you're building a mobile app. In Agile, you would first create a simple version of the app that does basic things, like letting users log in and view a home screen. You wouldn’t wait to finish every feature before showing it to people.

**2. Small teams making decisions**

Agile uses **small, flexible teams** with different skills working together. For example, in the app team, you’d have a developer, designer, and tester working closely to solve problems quickly. They wouldn’t need to ask their bosses for approval every time they make a small decision. This helps things move faster compared to traditional teams where decisions are made by different departments (like design, development, and testing working separately).

**3. Rapid iteration with customer input**

In Agile, the team works in **short cycles** (often called "sprints") to improve the product step by step. After each sprint (usually 1–2 weeks), the team would show the app to customers or users to get feedback. For example, after a sprint, users might say the login button is hard to find, and the team would fix that in the next sprint. This way, changes happen quickly based on real feedback.

**Example:**

Let’s say you’re building an online shopping website.

* **Instead of writing a long plan** for every feature of the site, you might start by making a basic version where users can browse a few products.
* A **small team** of developers, designers, and testers would work together closely to get that done quickly.
* After showing the basic website to customers, they might say they want a "search bar" or an easier checkout process. The team would **quickly improve** the site based on that feedback in the next few sprints.

In Agile, it’s all about working fast, getting real input from users, and constantly improving!

One of the most popular Agile methods is called **Scrum**. It helps teams work better and faster, especially in software development. It was created more than 10 years ago by **Ken Schwaber** and **Dr. Jeff Sutherland**. Today, many big and small companies use Scrum, like:

* **Yahoo!**
* **Microsoft**
* **Google**
* **Lockheed Martin**
* **Motorola**
* **SAP**
* **Cisco**
* **GE Medical**
* **CapitalOne**
* **US Federal Reserve**

These companies use Scrum because it helps them manage their work in a flexible way, where they can adapt to changes easily.

**Quiz:**

Here’s a simple quiz about Scrum. Let's look at the important parts:

**1. List of Roles in Scrum:**

* **Product Owner:** Decides what the team should work on.
* **Scrum Master:** Helps the team follow Scrum rules and removes any problems.
* **Development Team:** The people who actually do the work (like building software).

**2. List of Meetings in Scrum:**

* **Sprint Planning:** At the start of the sprint, the team decides what to work on.
* **Daily Scrum (Stand-up):** A short meeting every day to check on progress.
* **Sprint Review:** At the end of the sprint, the team shows what they’ve done.
* **Sprint Retrospective:** The team discusses what went well and what can be improved.

**3. List of Documents in Scrum:**

* **Product Backlog:** A list of all tasks and features the team will work on.
* **Sprint Backlog:** A smaller list of tasks the team will focus on during the current sprint.
* **Burndown Chart:** A chart that shows how much work is left to do during a sprint.

**How to create a Burndown chart? Burndown chart examples..**

These are the basic things you need to know to understand Scrum!

**Scrum** is a way of working that helps teams build things, like software, in small, manageable steps. Let’s break down how it works, with examples:

* **Scrum is Iterative and Incremental:** Scrum is like doing your work in small pieces or steps. Each step is called a **Sprint**, which usually lasts between 1 to 4 weeks. For example, if you're building a new app, instead of doing everything at once, you break it down into smaller tasks, like designing a screen or coding a feature.
* **Fixed Sprint Length:** Each Sprint lasts a fixed amount of time. Let’s say your Sprint is 2 weeks long. No matter what, the Sprint ends in 2 weeks, even if some work isn't finished. You don’t extend the Sprint; you just move on to the next one.
* **Picking Tasks at the Start of a Sprint:** At the beginning of a Sprint, the team looks at a list of things that need to be done (this is called a **Product Backlog**) and picks a few tasks to focus on. For example, they may decide to work on fixing bugs or adding new features to an app. The team promises to finish these tasks by the end of the Sprint.
* **Daily Check-in (Daily Scrum):** Every day, the team has a quick meeting (sometimes called a **Daily Scrum** or **Stand-up**). During this meeting, everyone shares what they did yesterday, what they’ll do today, and if they have any problems. It’s like a daily update to make sure everyone is on track.
* **End of Sprint Review:** At the end of the Sprint, the team shows what they’ve built, like a new feature in an app or a working piece of software. They also get feedback from others, which they can use to make improvements in the next Sprint.
* **Producing a "Done" Product:** Scrum focuses on making sure that at the end of each Sprint, the team has a working product that is really "done." For example, if it’s software, it means the code has been fully tested and is ready to be shipped to customers if needed.

**Scrum Roles:**

There are three main roles in Scrum:

1. **Product Owner:**
   * This person decides what the team should work on and in what order. Think of them as the team’s guide, making sure the most important work gets done first.
2. **Team Members:**
   * These are the people doing the actual work, like writing code, designing, or testing. They’re the ones who complete the tasks in each Sprint.
3. **Scrum Master:**
   * The Scrum Master helps the team follow the rules of Scrum and solves any problems that get in the way of progress. They make sure the team can work smoothly, like removing obstacles or distractions.

In summary, Scrum is a way of working where teams break their work into smaller parts (Sprints), check progress daily, and deliver working products in short cycles!

Scrum is a way of working in teams to build products, like software or websites, in an organized and efficient way. There are three key roles in Scrum: **Product Owner**, **Team Members**, and **Scrum Master**. Let’s explain each one using simple examples.

**Product Owner:**

The **Product Owner** is like the team leader for deciding **what the product should be**. They listen to different people, like customers, team members, and company leaders, to understand what the product needs. Then, they turn these ideas into a clear **vision** for the team to follow.

* **Example:** Imagine the team is building a new mobile app. The Product Owner listens to customer feedback, maybe through surveys or talking to users. They also ask the development team for input. Based on all this, the Product Owner decides what features to add, like a new messaging system in the app.
* In some cases, the Product Owner could also be the **customer** themselves, especially if the product is for a single company. But sometimes, the customer is **millions of people**, like users of a social media app.
* The Product Owner is responsible for managing the **Product Backlog** (a list of tasks to do), **planning releases**, and deciding which tasks are most important based on **business value**. They also **accept or reject** the team’s work.

**Team Members:**

The **Team Members** are the people who actually **build the product**. They might be programmers, designers, testers, marketers, or other experts needed to complete the project.

* **Example:** If the project is building a website, team members would include people who code the website, design the layout, test the functionality, and maybe even promote the site.
* A typical Scrum team has between **5 to 10 people**, but teams can be as small as 3 or as large as 15. If the project is really big, you can have multiple Scrum teams working together on different parts of the product.
* It’s better if team members are **fully dedicated** to one project at a time, so they can focus and be more productive.
* The team is **self-organizing** and **cross-functional**, meaning they work together without being told exactly what to do, and they have all the skills needed to complete their tasks.

**Scrum Master:**

The **Scrum Master** helps the team **stay on track** and makes sure they can work smoothly. They are not the team’s boss but are more like a **coach** or **referee** who helps remove obstacles, organizes meetings, and ensures everyone follows the Scrum process.

* **Example:** If the team is having trouble with a tool they need to use, the Scrum Master steps in to solve the problem, so the team can keep working without delays.
* Scrum Masters can come from many different backgrounds, like **Project Management**, **Engineering**, **Design**, or **Testing**. Their job is to **facilitate** meetings, **remove blockers** (things that slow the team down), and **protect** the team from distractions.
* It’s important that the **Scrum Master** and **Product Owner** are different people. Sometimes, the Scrum Master needs to **push back** on the Product Owner, especially if the Product Owner wants to add new tasks in the middle of a Sprint, which could disrupt the team’s focus.
* Unlike a traditional **Project Manager**, the Scrum Master doesn’t tell people what to do or assign tasks. Instead, they help the team manage itself and be successful.

**Key Takeaways:**

* The **Product Owner** decides what to build based on input from different sources and manages the list of tasks.
* **Team Members** build the product and are experts in their fields (coding, design, etc.).
* The **Scrum Master** makes sure the team can work without problems and helps everyone follow the Scrum process.

In Scrum, these roles work together to make sure the product is delivered efficiently and meets the needs of the users or customers!

The **Product Backlog** is a big list of everything that needs to be done to create or improve a product. It is managed by the **Product Owner** and is an important part of the Scrum process. Let’s break it down step by step with examples.

**What is the Product Backlog?**

The **Product Backlog** is the first step in Scrum. The Product Owner creates a **vision** for the product, which means they think about what the product should be and what features it should have. They then turn this vision into a **prioritized list** of tasks that need to be done.

* **Example:** Imagine you're creating an online store to sell books. The Product Backlog would include all the important features, like allowing customers to add books to a shopping cart, creating a secure payment system, and fixing any bugs in the website.

**How is the Product Backlog Organized?**

The items in the Product Backlog are organized by **importance**. The most important tasks that deliver the most value to the customer are at the top of the list. The list is **always changing** based on new ideas, customer feedback, or technical challenges.

* **Example:** In your online bookstore, the Product Backlog might look like this:
  1. **Add books to shopping cart** (high value to customers).
  2. **Fix the checkout button that’s not working** (high priority bug fix).
  3. **Research faster payment options** (exploratory task).
  4. **Make the website handle more users** (technical improvement).

The **Product Owner** regularly updates this list, depending on what’s important at that moment, like customer requests or what competitors are doing.

**What Types of Items Are in the Product Backlog?**

The Product Backlog includes different types of work:

* **Features:** New things the product needs.
  + **Example:** "Let users place books in a shopping cart."
* **Development requirements:** Technical work to improve the product.
  + **Example:** "Make the payment system faster."
* **Exploratory work:** Researching solutions or figuring out problems.
  + **Example:** "Investigate how to speed up credit card validation."
* **Bugs:** Problems that need fixing.
  + **Example:** "Fix the error in the checkout process."

**The Product Backlog Is a Single Source of Truth:**

There is only **one Product Backlog** for the entire product. This means that every task, big or small, is on the same list. It’s the **complete view** of all the work that needs to be done. The Product Owner is responsible for deciding which tasks are the most important.

* **Example:** If both the "Add books to shopping cart" feature and "Fix checkout bug" are on the list, the Product Owner has to decide which one is more urgent and should be done first.

**Backlog Items Can Be Big or Small:**

Some tasks in the Product Backlog are big, and they might need to be **broken down** into smaller parts during planning. Smaller tasks may also be grouped together to save time.

* **Example:** A big task like "Build a user login system" could be split into smaller tasks like "Create the login page," "Connect the login to the database," and "Set up password recovery."

**Detailing the Product Backlog:**

It’s a **myth** that Scrum doesn’t allow detailed planning. The amount of detail for each task is decided by the Product Owner and the team. Some tasks need more details, while others don’t.

* **Example:** If you're adding a new feature like "Let users rate books," you might need a lot of details about how the rating system will work. But for a small bug fix, you might only need a simple description.

**Key Takeaways:**

* The **Product Backlog** is a prioritized list of everything that needs to be done for the product.
* It includes features, technical work, research, and bugs.
* The list is always changing, and the Product Owner decides what’s most important.
* The Product Backlog is the **single source of truth** for all work on the product.

In short, the Product Backlog is the master plan that guides the team in building the product, ensuring they work on the most valuable tasks first!

The **Sprint Planning Meeting** is the first step in every **Sprint** (a short, focused period of work in Scrum). It’s a meeting where the **Product Owner**, the **Scrum Team**, and the **Scrum Master** get together to decide what the team will work on for the next Sprint. Let’s break it down with examples:

**Part 1: Reviewing the Product Backlog**

In the first part of the Sprint Planning Meeting:

* The **Product Owner** shows the team the **Product Backlog**, which is a list of tasks ranked by importance. The Product Owner explains what’s most important and why, so the team understands the goals.
* The **Scrum Master** helps organize the discussion, making sure everyone understands the work ahead.
* **Example:** Imagine you’re building a new feature for an app. The Product Owner tells the team the top priority is to add a "wishlist" feature where users can save items they like. They explain why this is important for customers and how it fits into the overall plan for the app.

**Part 2: Selecting Work for the Sprint**

In the second part of the meeting:

* The **Scrum Team** looks at the Product Backlog, starting with the highest priority tasks, and decides which items they can realistically complete by the end of the Sprint (usually 1-4 weeks).
* This is a key part of Scrum: **the team chooses** how much work they can commit to, instead of the Product Owner assigning tasks to them. This makes the commitment more reliable because the team knows their own limits.
* **Example:** The team looks at the "wishlist" feature and decides that in the next two weeks, they can build the part that lets users add items to the wishlist. They choose this task because it’s the highest priority for the Product Owner.

**How Much Work Can the Team Handle?**

* The Sprint Planning Meeting can take a few hours because the team needs to think carefully about how much work they can handle. The team considers how much time each person has for **Sprint-related work**. This means subtracting time spent on other things like fixing urgent bugs, attending meetings, answering emails, and taking breaks.
* Usually, each team member has about **4-6 hours per day** to work on Sprint tasks.
* **Example:** If a team member is busy fixing bugs for 2 hours each day and attending meetings for another hour, they have around 5 hours each day for Sprint work. Over a 2-week Sprint, that’s about 50 hours of time they can commit.

**Creating the Sprint Backlog**

* After deciding how much time they have, the team starts with the top item on the **Product Backlog** (the Product Owner's highest priority) and breaks it down into smaller tasks. These tasks go into a document called the **Sprint Backlog**, which is the list of all the work the team has agreed to complete during the Sprint.
* **Example:** The team takes the "wishlist" feature and breaks it down into smaller tasks, like:
  1. Design the wishlist page.
  2. Code the part where users can add items to the wishlist.
  3. Test if the wishlist works.

Each team member chooses which task they’ll work on and estimates how long it will take. For example, a designer might sign up to design the wishlist page and estimate it will take them 6 hours.

**End of the Sprint Planning Meeting**

* By the end of the meeting, the team will have a **Sprint Backlog**, which is a list of tasks for the Sprint, along with the time estimates for each task and who is responsible for each one.
* **Example:** After the meeting, the Sprint Backlog might look like this:
  1. **Design wishlist page** – Assigned to Sarah – 6 hours.
  2. **Code wishlist feature** – Assigned to John – 10 hours.
  3. **Test wishlist functionality** – Assigned to Alex – 4 hours.

This list helps keep everyone organized and on track for the rest of the Sprint.

**Key Takeaways:**

* The **Sprint Planning Meeting** is when the team decides what to work on for the upcoming Sprint.
* The **Product Owner** explains the most important tasks from the Product Backlog.
* The **Scrum Team** chooses how much work they can handle and creates a detailed plan, called the **Sprint Backlog**, with tasks, time estimates, and responsibilities.

This meeting ensures that everyone is on the same page and ready to start the Sprint with clear goals!

Scrum is designed to keep the team focused on their goals for a short period of time called a **Sprint** (usually 1-4 weeks). Let’s break down two important parts of Scrum: **Handling Change** and the **Daily Stand-Up Meeting**, using simple language and examples.

**Handling Change in Scrum**

In Scrum, once a **Sprint** starts, the **Product Owner** (the person in charge of deciding what needs to be done) **cannot add new tasks** until the next Sprint begins. The idea is to help the team stay focused on finishing what they’ve already committed to.

* **Example:** Imagine the team is working on adding a "wishlist" feature to an online store. Halfway through the Sprint, the Product Owner realizes they want to add a "customer reviews" feature. Even though it’s a good idea, they have to wait until the next Sprint to add it because the team is already focused on finishing the "wishlist" feature.

**What if something big changes?**

If something **really important** happens and changes the priorities (like a sudden business need), the Product Owner can **cancel the Sprint**. This means the team stops what they’re working on, and they start a new Sprint from scratch.

* **Example:** Let’s say the company finds out a competitor just launched a game-changing feature that must be copied immediately. The Product Owner can stop the current Sprint, and the team can start a new one focused on building that feature instead.

**The Daily Stand-Up Meeting**

Once a Sprint starts, the **Daily Stand-Up Meeting** happens every workday. This meeting is **short** (about 15 minutes) and takes place at the **same time each day**. Everyone on the **Scrum Team** attends, and they literally stand up during the meeting to keep it brief.

**What happens in the meeting?**

Each team member answers three quick questions:

1. **What did you do since the last meeting?**
2. **What will you do by the next meeting?**
3. **Are there any problems blocking your work?**

There’s **no discussion** during this meeting. If people need to talk about a problem, they wait until the meeting is over to discuss it further.

* **Example:**
  + **Sarah (a designer):** "Yesterday, I finished designing the wishlist page. Today, I’ll start designing the user profile page. I’m stuck because I need feedback from the Product Owner."
  + **John (a developer):** "Yesterday, I coded half of the wishlist feature. Today, I’ll finish the rest. No blockers."

**Role of the Scrum Master**

* The **Scrum Master** listens for any **problems** mentioned during the meeting (like Sarah needing feedback) and works to solve them after the meeting.
* The **Product Owner** or other managers can attend the meeting, but they shouldn’t ask questions or interfere with the team’s reports. The meeting is for the team to talk to **each other**, not to the boss.
* **Example:** If the Product Owner joins the meeting, they listen quietly while the team reports. After the meeting, they might talk to Sarah to give her the feedback she needs.

**What Happens After the Stand-Up Meeting?**

After the meeting:

* Each team member updates the time remaining to complete their tasks in the **Sprint Backlog**. This isn’t about how much time they’ve already spent but about how much work is still **left to do**.
* **Example:** If John finished half of the wishlist feature and thinks the second half will take another 5 hours, he updates the Sprint Backlog to show 5 hours remaining for that task.

**Tracking Progress**

The team uses these updates to see if they’re on track to finish the Sprint. If they realize they’re behind schedule, they either need to work faster or make the tasks **simpler** to complete them on time.

* **Example:** If the team sees that they’re running out of time to finish the wishlist feature, they might decide to cut out less important parts and focus on just getting the basics done.

**Key Takeaways:**

1. **No changes allowed** during the Sprint, unless the Product Owner cancels it due to a big change in priorities.
2. The **Daily Stand-Up Meeting** is a short daily check-in where team members quickly share progress and problems.
3. The **Scrum Master** helps remove problems, and the team focuses on completing the Sprint by tracking how much work is left, not how much time they’ve spent.

These practices help the team stay on track and focused, ensuring they work efficiently and address any roadblocks quickly!

Scrum has a process where the team works in short, focused periods of time called **Sprints** (usually 1-4 weeks). Let’s go through what happens during and after a Sprint.

**Sprint Status**

1. **Fixed End Date**: A key rule in Scrum is that the **Sprint end date never changes**. Even if the team hasn’t finished all the work they planned, the Sprint **ends on the scheduled date**. No extensions.
   * **Example:** If the team committed to finishing a new login page by the end of the Sprint, but didn’t finish, the Sprint still ends as planned.
2. **Admitting When Work Isn’t Done**: If the team didn’t finish their work, they openly **acknowledge this** at the end of the Sprint. This helps the team improve their ability to estimate and plan future work better.
   * **Example:** "We didn’t finish the login page because it took longer than expected to fix some bugs."
3. **Learning from Over- and Under-Commitment**: In the first few Sprints, teams often **over-commit** (take on too much work) and don’t finish everything. Then, they may **under-commit** (take on too little work) and finish early. After a few Sprints, teams usually get better at predicting how much work they can finish.
   * **Example:** After a few Sprints, the team figures out that they can finish 8 small tasks in a 2-week Sprint, and they plan accordingly.
4. **Consistent Sprint Length**: Teams should keep the **same Sprint length** (like 2 weeks) each time. This helps the team establish a work rhythm and better understand how much they can get done in that time.
   * **Example:** If a team always has 2-week Sprints, they can become good at estimating how much work can be done in two weeks.

**Sprint Review**

After the Sprint ends, the team holds a **Sprint Review** to show what they’ve built. This is like a demo where they showcase their work to the **Product Owner**, **Scrum Master**, **customers**, **stakeholders**, and anyone else interested.

1. **No Fancy Presentations**: It’s a simple **demo**—no PowerPoints or flashy presentations, just a demonstration of what was built. The focus is on showing the real progress.
   * **Example:** If the team worked on the login page, they show how the login now works on the website.
2. **Getting Feedback**: Everyone in the meeting can ask questions and give feedback. This helps the team improve their work in the next Sprint.
   * **Example:** A stakeholder might say, “The login button should be more visible,” and the team takes note for the next Sprint.

**Sprint Retrospective**

After the Sprint Review, the team has a **Sprint Retrospective**. This is a meeting where the team talks about **what went well** and **what didn’t go well** during the Sprint, and how they can improve in the next Sprint.

1. **What Worked and What Didn’t**: The team writes down things that went well and things that could be improved.
   * **Example:** On one piece of paper, someone might write “Good collaboration,” and on another, someone might write “Too many distractions.”
2. **Look for Common Issues**: The team marks common problems (like distractions) and discusses how to fix them in the next Sprint.
   * **Example:** If several people mention distractions, the team could decide to have fewer unnecessary meetings during the next Sprint.
3. **Everyone Participates**: The **Scrum Team**, **Product Owner**, and **Scrum Master** all attend. Sometimes, an outside person helps run the meeting to make sure it’s fair and helpful.

**Starting the Next Sprint**

After the Sprint Review and Retrospective, the team is ready to start planning the next Sprint.

1. **Updating the Product Backlog**: The **Product Owner** takes all the feedback and new ideas that came up during the Sprint and updates the **Product Backlog**. Some new items are added, some are reordered, and some may be removed.
   * **Example:** If someone suggested adding a “Forgot Password” feature during the demo, the Product Owner might add this to the backlog and prioritize it for the next Sprint.
2. **Prioritization Meeting**: Many teams find it useful to have a quick **Prioritization Meeting** before the next Sprint. This is when the team and Product Owner review the backlog and suggest items to focus on in the upcoming Sprint.
   * **Example:** The team might suggest that before starting new features, they need to fix technical bugs in the current system.
3. **No Break Between Sprints**: Teams usually move from one Sprint to the next without downtime. They might finish a Sprint Review in the afternoon and start the next **Sprint Planning** meeting the next morning.
   * **Example:** After the demo, the team goes home, and the next morning they gather to plan the work for the next Sprint.
4. **Sustainable Pace**: Scrum emphasizes working at a **sustainable pace**—regular hours, no extreme overtime—so the team can keep this rhythm going Sprint after Sprint without burning out.
   * **Example:** The team works regular 8-hour days so they can continue working effectively over the long term.

**Key Takeaways:**

1. **The Sprint never gets extended**, even if the team doesn’t finish all their tasks.
2. The **Sprint Review** is a simple demo of what the team built, and they get feedback from anyone interested.
3. The **Sprint Retrospective** helps the team reflect on how to improve and avoid repeating mistakes.
4. After every Sprint, the team updates their **Product Backlog** and jumps right into the next Sprint without taking breaks, working at a pace they can sustain.