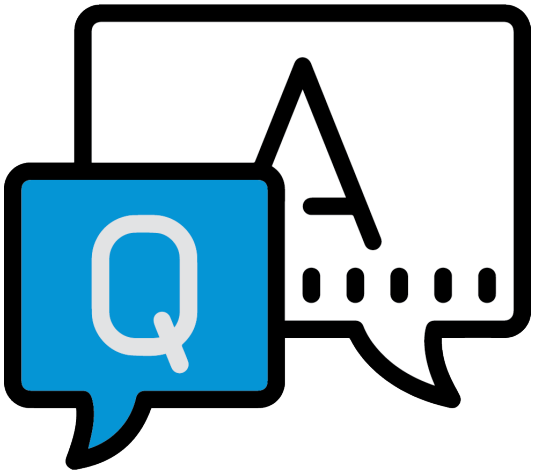
What is Agile? What is Scrum?

The FAQ Guide for everything you need to know



cPRIME RESOURCE CENTER

www.cprime.com | learn@cprime.com | (877) 753-2760

Agile Defined:

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals. Agile development refers to any development process that is aligned with the concepts of the Agile Manifesto. The Manifesto was developed by a group fourteen leading figures in the software industry, and reflects their experience of what approaches do and do not work for software development. Read more about the Agile Manifesto.

Scrum Defined:

Scrum is a subset of Agile. It is a lightweight process framework for agile development, and the most widely-used one.

- A "process framework" is a particular set of practiceSs that must be followed in order for a process to be consistent with the framework. (For example, the Scrum process framework requires the use of development cycles called Sprints, the XP framework requires pair programming, and so forth.)
- "Lightweight" means that the overhead of the process is kept as small as possible, to maximize the amount of productive time available for getting useful work done.

A Scrum process is distinguished from other agile processes by specific concepts and practices, divided into the three categories of Roles, Artifacts, and Time Boxes. These and other terms used in Scrum are defined below. Scrum is most often used to manage complex software and product development, using iterative and incremental practices. Scrum significantly increases productivity and reduces time to benefits relative to classic "waterfall" processes. Scrum processes enable organizations to adjust smoothly to rapidly-changing requirements, and produce a product that meets evolving business goals. An agile Scrum process benefits the organization by helping it to

- Increase the quality of the deliverables
- Cope better with change (and expect the changes)
- Provide better estimates while spending less time creating them
- Be more in control of the project schedule and state

Benefits of Agile

Benefits to Customer

Customers find that the vendor is more responsive to development requests. High-value features are developed and delivered more quickly with short cycles, than with the longer cycles favored by classic "waterfall" processes.

Benefits to Vendors

Vendors reduce wastage by focusing development effort on high-value features, and reduce time-to-market relative to waterfall processes due to decreased overhead and increased efficiency. Improved customer satisfaction translates to better customer retention and more positive customer references.

Benefits to Development Teams

Team members enjoy development work, and like to see their work used and valued. Scrum benefits Team members by reducing non-productive work (e.g., writing specifications or other artifacts that no one uses), and giving them more time to do the work they enjoy. Team members also know their work is valued, because requirements are chosen to maximize value to customers.

Benefits to Product Managers

Product Managers, who typically fill the Product Owner role, are responsible for making customers happy by ensuring that development work is aligned with customer needs. Scrum makes this alignment easier by providing frequent opportunities to re-prioritize work, to ensure maximum delivery of value.

Benefits to Project Managers

Project Managers (and others) who fill the ScrumMaster role find that planning and tracking are easier and more concrete, compared to waterfall processes. The focus on task-level tracking, the use of Burndown Charts to display daily progress, and the Daily Scrum meetings, all together give the Project Manager tremendous awareness about the state of the project at all times. This awareness is key to monitoring the project, and to catching and addressing issues quickly.

Benefits to PMOs and C-Level Executives

Scrum provides high visibility into the state of a development project, on a daily basis. External stakeholders, such as C-Level executives and personnel in the Project Management Office, can use this visibility to plan more affectively, and adjust their strategies based on more hard information and less speculation.

Scrum Requirements

Scrum does not define just what form requirements are to take, but simply says that they are gathered into the Product Backlog, and referred to generically as "Product Backlog Items," or "PBIs" for short. Given the time-boxed nature of a Sprint, we can also infer that each set should require significantly less time to implement than the duration of the Sprint. Most Scrum projects borrow the "XP" (Extreme Programming) practice of describing a feature request as a "User Story," although a minority uses the older concept of a "Use Case." We will go with the majority view here, and describe three reasonably-standard requirements artifacts found in Product Backlogs.

User Story

A User Story describes a desired feature (functional requirement) in narrative form. User Stories are usually written by the Product Owner, and are the Product Owner's responsibility. The format is not standardized, but typically has a name, some descriptive text, references to external documents (such as screen shots), and information about how the implementation will be tested. For example, a Story might resemble the following:

Name: Planner enters new contact into address book, so that one can contact the person later by postal or electronic mail

Description: Planner enters standard contact information (first and last name, two street address lines, city, state, zip / postal code, country, etc.) into contact-entry screen. One clicks "Save" to keep the data, and "Cancel" to discard data and return to previous screen.

Screens and External Documents: http://myserver/screens/contact-entry.html

How to test: Tester enters and saves the data, finds the name in the address book, and clicks on it. One sees a read-only view of the contact-entry screen, with all data previously entered.

The elements in this User Story are:

1. **Name:** The Name is a descriptive phrase or sentence. The example uses a basic "Role-Action-Reason" organization. Another common style, popularized by Mike Cohn, follows the template "As a <type of user>, I want <some goal> so that <some reason>." The choice of template is less important than having a workable standard of some kind.
2. **Description:** This is a high-level (low-detail) description of the need to be met. For functional (user-facing) requirements, the description is put in narrative form. For non-functional requirements, the description can be worded in any form that is easy to understand. In both cases, the key is that the level of detail is modest, because the fine details are worked out during the implementation phase, in discussions between team members, product owners, and anyone else who is involved. (This is one of the core concepts of Scrum: Requirements are specified at a level that allows rough estimation of the work required to implement them, not in detail.)

3. **Screens and External Documents:** If the Story requires user-interface changes (especially non-trivial ones), the Story should contain or link to a prototype of the changes. Any external documents required to implement the Story should also be listed.
4. **How to test:** The implementation of a Story is defined to be complete if, and only if, it passes all acceptance tests developed for it. This section provides a brief description of how the story will be tested. As for the feature itself, the description of testing methods is short, with the details to be worked out during implementation, but we need at least a summary to guide the estimation process.

There are two reasons for including the information about how to test the Story. The obvious reason is to guide development of test cases (acceptance tests) for the Story. The less-obvious, but important, reason, is that the Team will need this information in order to estimate how much work is required to implement the story (since test design and execution is part of the total work).

Story

Not all requirements for new development represent user-facing features, but do represent significant work that must be done. These requirements often, but not always, represent work that must be done to support user-facing features. We call these non-functional requirements "Technical Stories." Technical Stories have the same elements as User Stories, but need not be cast into narrative form if there is no benefit in doing so. Technical Stories are usually written by Team members, and are added to the Product Backlog. The Product Owner must be familiar with these Stories, and understand the dependencies between these and User Stories in order to rank (sequence) all Stories for implementation.

Defect

A Defect, or bug report, is a description of a failure of the product to behave in the expected fashion. Defects are stored in a bug-tracking system, which may or may not be physically the same system used to store the Product Backlog. If not, then someone (usually the Product Owner) must enter each Defect into the Product Backlog, for sequencing and scheduling.

Scrum Roles

The three roles defined in Scrum are the ScrumMaster, the Product Owner, and the Team (which consists of Team members). The people who fulfill these roles work together closely, on a daily basis, to ensure the smooth flow of information and the quick resolution of issues.

ScrumMaster

The ScrumMaster (sometimes written "Scrum Master," although the official term has no space after "Scrum") is the keeper of the process. The ScrumMaster is responsible for making the process

run smoothly, for removing obstacles that impact productivity, and for organizing and facilitating the critical meetings. The ScrumMasters responsibilities include

- Removing the barriers between the development Team and the Product Owner so that the Product Owner directly drives development.
- Teach the Product Owner how to maximize return on investment (ROI), and meet his/her objectives through Scrum.
- Improve the lives of the development Team by facilitating creativity and empowerment.
- Improve the productivity of the development Team in any way possible.
- Improve the engineering practices and tools so that each increment of functionality is potentially shippable.
- Keep information about the Team's progress up to date and visible to all parties.

In practical terms, the ScrumMaster needs to understand Scrum well enough to train and mentor the other roles, and educate and assist other stakeholders who are involved in the process. The ScrumMaster should maintain a constant awareness of the status of the project (its progress to date) relative to the expected progress, investigate and facilitate resolution of any roadblocks that hold back progress, and generally be flexible enough to identify and deal with any issues that arise, in any way that is required. The ScrumMaster must protect the Team from disturbance from other people by acting as the interface between the two. The ScrumMaster does not assign tasks to Team members, as task assignment is a Team responsibility. The ScrumMaster's general approach towards the Team is to encourage and facilitate their decision-making and problem-solving capabilities, so that they can work with increasing efficiency and decreasing need for supervision. The goal is to have a team that is not only empowered to make important decisions, but does so well and routinely.

Product Owner

The Product Owner is the keeper of the requirements. The Product Owner provides the "single source of truth" for the Team regarding requirements and their planned order of implementation. In practice, the Product Owner is the interface between the business, the customers, and their product related needs on one side, and the Team on the other. The Product Owner buffers the Team from feature and bug-fix requests that come from many sources, and is the single point of contact for all questions about product requirements. Product Owner works closely with the team to define the user-facing and technical requirements, to document the requirements as needed, and to determine the order of their implementation. Product Owner maintains the Product Backlog (which is the repository for all of this information), keeping it up to date and at the level of detail and quality the Team requires. The Product Owner also sets the schedule for releasing completed work to customers, and makes the final call as to whether implementations have the features and quality required for release.

Team

The Team is a self-organizing and cross-functional group of people who do the hands-on work of developing and testing the product. Since the Team is responsible for producing the product, it must also have the authority to make decisions about how to perform the work. The Team is therefore self-organizing: Team members decide how to break work into tasks, and how to allocate tasks to individuals, throughout the Sprint. The Team size should be kept in the range from five to nine people, if possible. (A larger number make communication difficult, while a smaller number leads to low productivity and fragility.) Note: A very similar term, "Scrum Team," refers to the Team plus the ScrumMaster and Product Owner.

What is your next step with Agile?

Training and Certifications

We are the largest provider of Agile training in the United States. Our public training courses are available across the country, with locations and dates that are convenient for you. We are Scrum Alliance Registered Education providers and SAFe Gold SPCT partners, and we are confident that we have the right course for you. You can also talk to us about corporate training deals and get your entire team (or company) certified!

Get more resources

We have plenty more resources to help you along your Agile journey. Visit www.cprime.com to check out our full resource library. We add blogs weekly and add new live and recorded webinars every month!

Get help from our experts

Sometimes you need a little extra help to bring about lasting change. Our experts have deep expertise and a range of specialties. Talk to us about workshops, private training, assessments, or anything else. We are confident we can help. Email learn@cprime.com.