

Scrum

This documents how to use Scrum to build products. In doing so, it will describe how the framework and its artifacts, time-boxes, roles, and rules work together. Scrum does not include techniques and processes for building products; it is simply a framework within the techniques and processes that you choose can be used. It will, however, point out the efficacy and flaws of these techniques and processes so you can improve them.

Scrum is a framework for developing complex products and systems that is grounded in empirical process control theory¹. Scrum employs an iterative, incremental approach to optimize predictability and control risk. Within each iteration, Scrum employs self-organizing, cross-functional Teams to optimize flexibility and productivity.

The heart of Scrum is a **Sprint**. A Sprint is one iteration of a month or less that is of consistent length throughout a development effort. All Sprints use the same Scrum framework, and all Sprints end with an increment of the end product that is potentially releasable. The increment is a complete slice, or piece, of the finished product or system that is developed by the end of an iteration, or Sprint. One Sprint starts immediately after the other.

Scrum employs time boxes to create regularity. The time boxes within Scrum are the **Sprint Planning Meeting, the Sprint, the Daily Scrum, the Sprint Review, and the Sprint Retrospective**.

Scrum employs self-organizing, cross functional **Scrum Teams** to do the work. Each Scrum Team has three roles where accountability and responsibility lie. The **ScrumMaster** is responsible for the process being understood and followed. The **Product Owner** is responsible for maximizing the value of the work done. The **Team** does the work. The Team consists of developers with all the skills to turn the Product Owner's requests into the potentially shippable increment each Sprint.

The Team is usually seven plus or minus two members.

Scrum employs several artifacts. The **Product Backlog** is a prioritized list of everything that anyone can think of that is needed in the product. The **Sprint Backlog** is a list of tasks to turn the Product Backlog for one Sprint into an increment of potentially shippable product. A **burndown** is a measure of remaining backlog over time. A **Release Burndown** measures remaining Product Backlog across the time of a release plan. A **Sprint Burndown** measures remaining Sprint Backlog across the time of a Sprint.

The Scrum framework consists of these time-boxes, Teams (with roles), and artifacts glued together by **Rules**. One Rule is that only the people committed to turning the Product Backlog into an increment, the Team, talks during a

TIP

If there are fewer people on the team, the team may reach skill constraints during parts of the Sprint. If there are more members, the team may be overwhelmed with too many people to collaborate and to keep informed. In our experience, however, teams do range outside this recommended size of seven plus or minus two.

¹ "Agile Software Development with Scrum," Ken Schwaber, Microsoft Press, 2004

Daily Scrum.

Rules bind the roles, time-boxes, and meetings. These rules are implicit throughout this document as the roles, time-boxes and meetings are described. When there are suggested approaches, or TIPS, on how to proceed, these are in separate *TIP* boxes.

Scrum has been used to develop complex products since the early 1990's. Many best practices have been uncovered for developing complex products within the Scrum framework. In tandem with this guide, the books, "Agile Project Management with Scrum" (Schwaber, Microsoft Press, 2004) and "The Enterprise and Scrum" (Schwaber, Microsoft Press, 2007) contain many tips about managing projects and scaling Scrum.

TIP

When rules are not stated, the users of Scrum are expected to figure out what to do. Don't try to figure out a perfect solution, because the problem usually changes quickly. Instead, try something and see how it works. The inspect and adapt mechanisms of Scrum's empirical nature will guide you.

Scrum Theory

Scrum is a framework for developing complex products and systems that is grounded in empirical process control theory². Empirical process control has three legs underlying all of its implementations: transparency, inspection, and adaptation. Transparency means that those aspects of the process that affect the outcome must be visible to those managing the outcomes. Not only must these aspects be transparent, but also what is being seen must be known. That is, when someone inspecting a process believes that something is done, it must be equivalent to their definition of done.

The second leg is inspection. The various aspects of the process must be inspected frequently enough so that unacceptable variances in the process can be detected. The frequency of inspection has to take into consideration that all processes are changed by the act of inspection. A conundrum occurs when the required frequency of inspection exceeds the tolerance to inspection of the process. Fortunately, this doesn't seem to be true of software development. The other factor is the skill and diligence of the people inspecting the work results.

The third leg of empirical process control is adaptation. If the inspector determines from the inspection that one or more aspects of the process are outside acceptable limits, and that the resulting product will be unacceptable, the inspector must adjust the process or the material being processed. The adjustment must be made as quickly as possible to minimize further deviation.

There are three inspect and adapt points in Scrum. The Sprint Review and Planning meetings are used to inspect progress toward the Release Goal, and to make adaptations that optimize the value of the next Sprint. The Daily Scrum meeting is used to inspect progress toward the Sprint goal, and to make adaptations that optimize the value of the next work day. The Retrospective meeting is used to adapt the process and interactions of the last Sprint, and to make adaptations that make the next Sprint more productive, fulfilling, and enjoyable.

² "Process Dynamics, Modeling, and Control," Babatunde A. Ogunnaike and W. Harmon Ray, Oxford University Press, 1994

Roles

The Scrum Team

The Scrum Team consists of the ScrumMaster, the Product Owner, and the Team (of developers). Scrum team members are called “pigs”. Everyone else is a “chicken.” Chickens cannot tell “pigs” how to do their work. Chickens and pigs come from the story,

“A chicken and a pig are together when the chicken says, “Let’s start a restaurant!”

The pig thinks it over and says, “What would we call this restaurant?”

The chicken says, “Ham n’ Eggs!”

The pig says, “No thanks, I’d be committed, but you’d only be involved!”

The ScrumMaster

The ScrumMaster is responsible for ensuring that Scrum values, practices, and rules are enacted and enforced. The ScrumMaster is the driving force behind all of the Scrum, and helps the Scrum team and the organization adopt Scrum and use it to improve itself. The ScrumMaster teaches the team by coaching, teaching, and leading it to be more productive and produce higher quality product. The ScrumMaster helps the Team understand and use self-management and cross-functionality. However, the ScrumMaster is not the manager of the Team.

TIP

The ScrumMaster may be part of the Team, a developer performing Sprint tasks. However, there may be conflict between removing impediments and performing tasks. The ScrumMaster is never the Product Owner.

The Product Owner

The Product Owner is the one and only person responsible for managing and controlling the Product Backlog. This is the person who is officially responsible for the value of the work done. This person maintains the Product Backlog and ensures that it

TIP

The ScrumMaster works with the customers and management to identify and instantiate a Product Owner. The ScrumMaster teaches the Product Owner how to do his or her job, and to manage to optimize value. Product Owners are expected to know how to manage to optimize value using Scrum. If they don’t, we hold the ScrumMaster accountable.

is visible to everyone. Everyone knows what items have the highest priority, so everyone knows what will be worked on.

The Product Owner is one person, not a committee. Committees may exist that advise or influence this person, but any person or body of people wanting an item's priority changed has to convince the Product Owner. Organizations have many ways of setting priorities and requirements. These practices will be influenced by Scrum across time, particularly through the meeting that reviews product increments (Sprint Review).

For the Product Owner to succeed, everyone in the organization has to respect his or her decisions. No one is allowed to tell the Teams to work from a different set of priorities, and Teams aren't allowed to listen to anyone who says otherwise. The Product Owner's decisions are visible in the content and prioritization of the Product Backlog. This visibility requires the Product Owner to do his or her best, and makes the role of Product Owner both a demanding and a rewarding one.

TIP

For commercial development, the Product Owner may be the product manager. For in-house development efforts, the Product Owner could be the user department manager.

TIP

The Product Owner can be a Team member, also doing development work. This additional responsibility may cut into the Product Owner's ability to work with stakeholders. However, the Product Owner can never be the ScrumMaster.

The Team

Teams are the developers that turn Product Backlog into increments of potentially shippable functionality every Sprint. Teams are responsible for organizing themselves to do the work. Teams are cross-functional, having all the skills needed to create an increments. There are no titles on Teams. Teams self-organize to turn the requirements and technology into product functionality. Scrum avoids people who refuse to code because they are systems architects, or designers. Everyone chips in and does his or her best, doing or learning how to do what is needed. Scrum Team members don't have job descriptions other than doing the best possible. No titles, no exceptions.

Teams are cross functional. A Scrum Team should include people with all of the skills necessary to meet the Sprint goal. Scrum eschews vertical Teams of analysts, designers, quality control, and coding engineers. A Scrum Team self-organizes so that everyone contributes to the outcome. Each Team member applies his or her expertise to all of the problems. The resultant synergy from a tester helping a designer construct code improves code quality and raises productivity.

TIP

Developers usually have specialized skills, such as programming, quality, analysis, architecture, user interface design, and data base design. However, the shared skills of how to address a requirement and turn it into a usable product tend to be greater than the unique skills.

The size of the Team optimize at seven people, plus or minus two. The Product Owner and ScrumMaster roles are not in this count unless they are also pigs.

Team composition may change at the end of a Sprint. Every time Team membership is changed, the productivity gained from self-organization is diminished. Care should be taken when changing Team composition.

TIP

Teams as small as three can benefit, but the small size limits the amount of interaction that can occur and reduces productivity gains. Teams larger than nine don't work out well. Team productivity decreases and the Scrum's control mechanisms become cumbersome. The Daily Scrum meeting may become too difficult. Most importantly, large Teams generate too much complexity for an empirical process.

The Product Backlog

The requirements for product being developed by the Scrum team(s) are listed in the Product Backlog. The Product Owner is responsible for the Product Backlog, its contents, its availability, and its prioritization. Product Backlog is never complete, and the initial cut at developing it only lays out the initially known and best-understood requirements.

The Product Backlog evolves as the product and the environment in which it will be used evolves. Backlog is dynamic, in that it constantly changes to identify what the product needs to be appropriate, competitive, and useful. As long as a product exists, Product Backlog also exists.

The Product Backlog is the master list of all functionality desired in the product. Product Backlog items have the attributes of a description, priority, and estimate. Priority is driven by risk, value, and necessity (non-functional requirements). There are many techniques for assessing these attributes.

The Product Backlog³⁴ represents everything necessary to develop and launch a successful product. It is a list of all features, functions, technologies, enhancements, and bug fixes that constitute the changes that will be made to the product for future releases.

TIP

Product Backlog items are usually stated as User Stories. Use Cases may be used for very precise needs, such as life or mission critical applications.

Product Backlog is sorted in order of priority. Top priority Product Backlog drives immediate development activities. The higher the priority, the more urgent it is, the more it has been thought about, and the more consensus there is regarding its value. Higher priority backlog is clearer and has more detailed information than lower priority backlog. Better estimates are made based on the greater clarity and increased detail. The lower the priority, the less the detail, until you can barely make out the item.

As a product is used, as its value increases, and as the marketplace provides feedback, the product's backlog emerges into a larger and more exhaustive list. Requirements never stop changing.

Product Backlog is an inventory. Changes in business requirements, marketplace attributes, technology understandings, and staffing cause changes in the Product Backlog. To minimize rework, only the highest priority items need to be detailed, or be fine-grained. The Product Backlog items that will occupy the Scrum Teams for the upcoming several Sprints are fine grained, decomposed so that any one item can be done within the duration of the Sprint.

³ "User Stories Applied: For Agile Software Development," Mike Cohn, Addison-Wesley, 2004

⁴ "Writing Effective Use Cases," Alistair Cockburn, Addison-Wesley, 2000

TIP

Scrum Teams often spend 10% of each Sprint grooming the product backlog to meet the above definition of the Product Backlog. When groomed to this level of granularity, the Product Backlog items at the top of the Product Backlog (highest priority, greatest value) are decomposed so they fit within one Sprint. They have been analyzed and thought through during the grooming process. When the Sprint Planning meeting occurs, these top priority Product Backlog items are well

Multiple Scrum Teams often work together on the same product. One Product Backlog is used to describe the upcoming work on the Product. A Product Backlog attribute that groups items is then employed. Grouping can occur by feature set, technology, or architecture, and is often used as a way to organize work by Scrum Team.

TIP

Acceptance tests are often used as another Product Backlog item attribute. They can often supplant more detailed text descriptions with a testable description of what the Product Backlog item must do when completed.

Release Planning

The purpose of release planning is to establish a plan and goals that the Scrum Teams and the rest of the organizations can mutually understand and communicate around. Release planning answers the question, how can we turn the vision into a winning product in the best possible way, and meet or exceed the desired customer satisfaction and Return on Investment. The release plan establishes the goal of the release, the highest priority Product Backlog, the major risks, and the overall features and functionality that the release will contain. It also establishes a probable delivery date and cost if nothing changes. The organization can then inspect progress and make adaptations on a Sprint by Sprint basis.

Products are built iteratively using Scrum, wherein each Sprint creates an increment of the product, starting with the most valuable and riskiest. More and more Sprints create additional increments of the product. Each increment is a potentially shippable slice of the entire product. When enough increments have been created for the Product to be of value, of use to its investors, the product is released.

Most organizations already have a release planning process. Most planning is done at the beginning of the release. Then the plan is followed. In Scrum release planning, an overall goal and probable outcomes are depicted. This release planning usually requires no more than 15-20% of the time an organization consumed to build a traditional release plan. However, a Scrum release performs just-in-time planning every Sprint Review and Sprint Planning meeting, as well as daily just-in-time planning at every Daily Scrum meeting. Overall, Scrum release efforts probably consume slightly more effort than traditional release planning efforts.

Release planning requires estimating and prioritizing the Product Backlog for the Release. There are many techniques for doing so that lie outside the purview of Scrum and should be investigated and used appropriately⁵

The Sprint

A Sprint is one iteration. Sprints are time-boxed. Sprints are protected by the ScrumMaster from any changes that would affect the Sprint Goal, The Team composition is constant throughout the Sprint. The quality of the increment remains constant throughout the Sprint.

A Sprint is similar to a project, consisting of planning, work, and a deliverable. A Planning Horizon is the period covered by a particular plan. In general, its length is dictated by the degree of uncertainty in the external environment: higher the uncertainty, shorter the planning horizon. In complex product development, the planning horizon is relatively short. The length of the Sprint is determined by the Planning Horizon. However, no Sprint is longer than one month, and all Sprints used to develop a product are of the same length. The consistent length of all Sprints creates the heartbeat of the overall work on the product.

Sprints contain and consist of the Sprint Planning meeting, the development work, the Sprint Review, and the Sprint Retrospective. Sprints occur one after another, with no time in between Sprints

A project is used to accomplish something; in software development, it is used to build a product, or system. Every project consists of a definition of what is to be built, a plan to build it, the work done according to the plan, and the resultant product.

Every project has a horizon, that is the time frame for which the plan is good. If the horizon is too long, the definition may have changed, too many variables may have entered in, the risk may be too great, etc.

Scrum is a framework for a project whose horizon is no more than one month long, where there is enough complexity that a longer horizon is too risky. The predictability of the project has to be controlled at least each month, and the risk that the project may go out of control or become unpredictable is contained at least each month.

TIP

If the Team senses that it has overcommitted, it meets with the Product Owner to remove or reduce the scope of Product Backlog selected for the Sprint. If the Team senses that it may have extra time, it can work with the Product Owner to select additional Product Backlog.

TIP

When a team begins Scrum, two-week Sprints allow it to learn without wallowing in uncertainty. Sprints of this length can be synchronized with other teams by adding two increments together.

⁵ "Agile Estimating and Planning," Mike Cohn, Prentice Hall, 2005

Sprint Planning Meeting

The Sprint Planning meeting is when the iteration is planned. It is time boxed to eight hours for a one month Sprint. For shorter Sprints, allocate approximately 5% of the total Sprint length to this meeting. and consists of two parts. The first part, a four-hour time box, is when what will be done in the Sprint is decided upon. The second part, another four-hour time box, is when the Team figures out how it is going to build this functionality into a product increment during the Sprint.

Sprint Planning Meeting Part 1

Sprint Planning Meeting (What) – In the first part, the Product Owner presents the top priority Product Backlog to the Team. They mutually figure what functionality to be developed during the next Sprint. Input to this meeting is the Product Backlog, the latest increment of product, the capacity of the Team, and past performance of the Team. The amount of backlog the Team selects is solely up to the Team. Only the Team can assess what it can accomplish over the upcoming Sprint.

TIP

Scrum teams often merge Part 1 and Part 2 together. However, the time-box must be adhered to. Otherwise planning activities can consume the Sprint.

Sprint Goal

Having selected the Product Backlog, a Sprint Goal is crafted. The Sprint Goal is an objective that will be met through the implementation of the Product Backlog. This is a statement that provides guidance to the team on why it is building the increment. The Sprint Goal is a subset of the release goal.

The reason for having a Sprint Goal is to give the Team some wiggle room regarding the functionality. For example, the goal for the above Sprint could also be: “Automate the client account modification functionality through a secure, recoverable transaction middleware capability.” As the Team works, it keeps this goal in mind. In order to satisfy the goal, it implements the functionality and technology. If the work turns out to be harder than the Team had expected, then the Team collaborates with the Product Owner and only partially implement the functionality.

Sprint Planning Meeting Part 2

Sprint Planning Meeting (How) - During the second four-hours of the Sprint Planning Meeting, the Team figures out how it will turn the Product Backlog selected during Sprint Planning Meeting (What) into a done increment. The team usually starts

by designing the work. While designing, the Team identifies tasks. These tasks are the detailed pieces of work needed to convert the Product Backlog into working software. Tasks should have decomposed so they can be done in less than one day. This task list is called the Sprint Backlog. The Team self-organizes to assign and undertake the work in the Sprint Backlog, either during the Sprint Planning meeting or just-in-time during the Sprint.

The Product Owner is present during this meeting to clarify the Product Backlog and to help make trade-offs. If the team determines that it has too much or too little work, it may renegotiate the Product Backlog with the Product Owner. The Team may also invite other people to attend in order to provide technical or domain advice. A new Team often first realizes that it will either sink or swim as a Team, not individually, in this meeting. The Team realizes that it must rely on itself. As it realizes this, it starts to self-organize to take on the characteristics and behavior of a real Team.

TIP

Usually, only 60-70% of the total Sprint Backlog will be devised in the Sprint Planning meeting. The rest are stubbed out for later detailing, or given large estimates that will be decomposed later in the Sprint.

Increment of Potentially Shippable Product Functionality

Scrum requires Teams to build an increment of product functionality every Sprint. This increment must be potentially shippable, for Product Owner may choose to immediately implement the functionality. To do so, the increment must be a complete slice of the product. It must be “done.” Each increment should be additive to all prior increments and thoroughly tested, ensuring that all increments work together.

Done

In product development, asserting that functionality is done might lead someone to assume that it is at least cleanly coded, refactored, unit tested, built, and acceptance tested. Someone else might assume only that the code has been built. If everyone doesn't know what the definition of “done” is, the other two legs of empirical process control don't work. When someone describes something as “done”, everyone must understand what “done” means.

Done defines what the Team means when it commits to “doing” a Product Backlog item in a Sprint. Some products do not contain documentation, so the definition of “done” does not include documentation. A completely “done” increment includes all of

the analysis, design, refactoring, programming, documentation and testing for the increment and all Product Backlog items in the increment. Testing includes unit, system, user, and regression testing, as well as non-functional tests such as performance, stability, security, and integration. Done includes any internationalization. Some Teams aren't yet able to include everything required for implementation in their definition of done. This must be clear to the Product Owner. This remaining work will have to be done before the product can be implemented and used.

TIP

“Undone” work is often accumulated in a Product Backlog item called “Undone Work” or “Implementation Work.” As this work accumulates, the Product Backlog burndown remains more accurate than if it weren't accumulated.

TIP

Some organizations are incapable of building a complete increment within one Sprint. They may not yet have the automated testing infrastructure to complete all of the testing. In this case, two categories are created for each increment: the “done” work and the “undone” work. The “undone” work is the portion of each increment that will have to be completed at a later time. The Product Owner knows exactly what he or she is inspecting at the end of the Sprint because the increment meets the definition of “done” and the Product Owner understands the definition. “Undone” work is added to a Product Backlog item named “undone work” so it accumulates and correctly reflects on the Release Burndown graph. This technique creates transparency in progress toward a release. The inspect and adapt in the Sprint Review is as accurate as this transparency.

For instance, if a Team is not able to do performance, regression, stability, security, and integration testing for each Product Backlog item. the proportion of this work to the work that can be done (analysis, design, refactoring, programming, documentation, unit and user testing) is calculated. Let's say that this proportion is 6 pieces of “done” and 4 pieces on “undone.” If the Team finishes a Product Backlog item of 6 units of work (the Team is estimating based on what it knows how to “do”), 4 is added to the “undone work” Product Backlog item when they are finished.

Sprint by Sprint, the “undone” work of each increment is accumulated and must be addressed prior to releasing the product. This work is accumulated linearly although it actually has some sort of exponential accumulation that is dependent on each organization's characteristics. Release Sprints are added to the end of any release to complete this “undone” work. The number of Sprints is unpredictable to the degree that the accumulation of “undone” work is not linear.

Sprint Backlog and Sprint Burndown

The Sprint Backlog consists of the tasks the Team performs to turn Product Backlog items into a “done” increment. Many are developed during the Sprint Planning Meeting (How). It is all of the work that the Team identifies as necessary to meet the Sprint goal. Sprint Backlog items must be decomposed. The decomposition is enough so changes in progress can be understood in the Daily Scrum.

The Team modifies Sprint Backlog throughout the Sprint, as well as Sprint Backlog emerging during the Sprint. As it gets into individual tasks, it may find out that more or fewer tasks are needed, or that a given task will take more or less time than had been expected. As new work is required, the Team adds it to the Sprint Backlog. As tasks are worked on or completed, the hours of estimated remaining work for each task is updated. When tasks are deemed unnecessary, they are removed. Only the Team can change its Sprint Backlog during a Sprint. Only the Team can change the contents or the estimates. The Sprint Backlog is a highly visible, real time picture of the work that the Team plans to accomplish during the Sprint, and it belongs solely to the Team.

Sprint Backlog Burndown is a graph of the amount of Sprint Backlog work remaining in a Sprint across time in the Sprint. To create this graph, determine how much work remains by summing the backlog estimates every day of the Sprint. The amount of work remaining for a Sprint is the sum of the work remaining for all of Sprint Backlog. Keep track of these sums by day and use them to create a graph that shows the work remaining over time. By drawing a line through the points on the graph, the Team can manage its progress in completing a Sprint’s work. Duration is not considered in Scrum. Work remaining and date are the only variables of interest.

TIP

Whenever possible, hand draw the burndown chart on a big sheet of paper displayed in the team's work area. Teams are more likely to see a big, visible chart than they are to look at Sprint burndown chart in Excel or a tool.

Daily Scrum

Each Scrum Team meets daily for a 15-minute status meeting called the Daily Scrum. The Daily Scrum is at the same time and same place throughout the Sprints. During the meeting, each Team member explains:

1. What he or she has accomplished since the last meeting;
2. What he or she is going to do before the next meeting; and,
3. What obstacles are in his or her way.

Daily Scrums improve communications, eliminate other meetings, identify and remove impediments to development, highlight and promote quick decision-making, and improve everyone's level of project knowledge.

The ScrumMaster ensures the Team has the meeting. The Team is responsible for conducting the Daily Scrum. The ScrumMaster teaches the team to keep the Daily Scrum short by enforcing the rules and making sure that people speak briefly. The

ScrumMaster also enforces the rule that chickens are not allowed to talk or in anyway interfere with the Daily Scrum.

The Daily Scrum is not a status meeting. It is not for anyone but the people transforming the Product Backlog items into an increment (the Team). The Team has committed to a Sprint Goal, and to these Product Backlog items. The Daily Scrum is an inspection of the progress toward that Sprint Goal (the three questions). Follow-on meetings usually occur to make adaptations to the upcoming work in the Sprint. The intent is to optimize the probability that the Team will meet its Goal. This is a key inspect and adapt meeting in the Scrum empirical process.

Abnormal termination of Sprints

Sprints can be cancelled before the Sprint time box is over. Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Team or the ScrumMaster.

Under what kind of circumstances might a Sprint need to be cancelled? Management may need to cancel a Sprint if the Sprint Goal becomes obsolete. A company as a whole may change direction. Market conditions or technological requirements might change. Management can simply change its mind. In general, a Sprint should be cancelled if it no longer makes sense given the circumstances. However, because of the short duration of Sprints, it rarely makes sense to do so.

When a Sprint is cancelled, any completed and “done” Product Backlog items are reviewed. They are accepted if they represent a potentially shippable increment. All other Product Backlog items are put back on the Product Backlog with their initial estimates. Any work done on them is assumed to be lost.

Sprint terminations consume resources, since everyone has to regroup in another Sprint planning meeting to start another Sprint. Usually, the first question that is asked when a Sprint is terminated is: “Who is responsible for this meeting occurring early?” Because people don’t want to be named as the answer to this question, very few Sprints end up being terminated.

TIP

When a Sprint is terminated, much of the work a Team has done will be lost. This is usually traumatic to the Team.

Sprint Review

At the end of the Sprint, a Sprint Review meeting is held. This is a four-hour time boxed meeting for one-month Sprints. For Sprints of lesser duration, this meeting must not consume more than 5% of the total Sprint.

During the Sprint Review, the Scrum team and stakeholders collaborate about what was just done. Based on that and changes to the Product Backlog during the Sprint, they collaborate about what are the next things that could be done. This is an informal meeting, with the presentation of the functionality intended to foster collaboration about what to do next.

The meeting includes at least the following elements. The Product Owner identifies what has been done and what hasn't been done. The Team discusses what went well during the Sprint and what problems it ran into, and how it solved these problems. The Team then demonstrates the work that is done and answers questions. The Product Owner then discusses the Product Backlog as it stands. He or she projects likely completion dates with various velocity assumptions. The entire group then collaborates about what it has seen and what this means regarding what to do next. The Sprint Review provides valuable input to subsequent Sprint Planning meeting.

Release Burndown

The Release burndown graph records the sum of remaining Product Backlog estimated effort across time. The estimated effort is in whatever unit of work the Scrum team and organization have decided upon. The units of time are usually Sprints.

Product Backlog item estimates are calculated initially during Release Planning, and thereafter as they are created. During Product Backlog grooming they are reviewed and revised. However, they can be updated at any time. The Team is responsible for all estimates. The Product Owner may influence

TIP

The trend line may be unreliable for the first two to three Sprints of a release unless the Scrum teams have worked together before, know the product well, and understand the underlying technology.

development process to make it more effective and enjoyable for the next Sprint.

There are many techniques for Retrospectives documented in books⁶.

The purpose of the Retrospective is to inspect how the last Sprint went in regards to people, relationships, process and tools. The inspection should identify and prioritize the major items that went well, and those items that - if done differently - could make

TIP

In some organizations, more work is added than is done. This may create a trend line that is flat or even slopes upwards. To compensate for this and retain transparency, a new floor may be created when work is added or subtracted. The floor should add or remove only significant changes and should be well documented.

the team by helping understand and select trade-off's, but the final estimate is made by the Team.

The Product Owner keep an updated Product Backlog list and Release Burndown posted at all times. A trend line can be drawn based on the change in remaining work.

Sprint Retrospective

After the Sprint Review and prior to the next Sprint Planning meeting, the Scrum team has a Sprint Retrospective meeting. At this three hour, time-boxed meeting the ScrumMaster encourages the team to revise, within the Scrum process framework and practices, their

⁶ Some helpful techniques for conducting a Scrum Retrospective are contained in "Agile Retrospectives: Making Good Teams Great," Esther Derby and Diana Larsen, Pragmatic Bookshelf, 2006.

things even better. These include team composition, meeting arrangements, tools, definition of “done,” methods of communication, and processes for turning Product Backlog items into something “done.” By the end of the Sprint Retrospective, the Scrum team should have identified actionable improvement measures that it implements in the next Sprint. These changes become the adaptation to the empirical inspection.

Multiple Scrum Teams, Scaling Scrum

Scrum does not change when many teams use it. Techniques for using Scrum in a scaled, multi-team environment can be found in recent books⁷.

⁷ “The Enterprise and Scrum,” Ken Schwaber, Microsoft Press, 2007.