AN INTRODUCTION TO SCRUM

Background

Most software development organizations share common goals – increased productivity, reliable predectability, greater transparency, improved agility. How organizations go about achieving these objectives may vary, and most can be achieved using Scrum.

The Scrum development process is typically introduced by contrasting it with more traditional methods of software development methodologies, notably Waterfall. The story basically goes like this – traditional methodologies tend to follow a sequential life cycle. The process begins with a detailed planning phase that meticulously thinks through, designs and documents in great detail the end-product. The next phase involves identifying the specific tasks to be completed and the time needed to complete them estimated to varying degrees of detail. This work is then handed off to the customer for review. Only once a customer's approval is obtained does actual work begin. Teams or individual developers start completing specialized work assembly-line style. This work is tested before being released to the customer. Throughout this process, controls are put in place to assure there are no deviations from what was originally planned.

These traditional approaches have strengths and weaknesses. They're very logical. They require thinking before acting. They require us to write down everything we're supposed to build in great detail. Why not just keep doing things this way?

Experience has shown there are lots of reasons. First, innovation can be stifled under traditional methods. Great ideas don't always manifest themselves at the beginning of the development process, and great ideas late in the release cycle are usually deemed a threat instead of a potential boon. Second, all the effort that's put into creating detailed documentation usually receives little attention. In fact, it's lucky to be read at all. And when it is read, it often leads to greater miscommunication rather than clarification! Third, rigid processes don't allow us to respond to changes in the marketplace during the development cycle. And that's not a good thing.

In the end, traditional development processes products are often condemned to be only as good as the initial idea. Agile development methods, on the other hand (of which Scrum is the most popular), focus on cross-functional teams empowered to make decisions, rapid iteration and continuous customer input. They're fundamentally about getting things done.

The Scrum Work Process Summarized

Scrum structures product development in cycles of work called Sprints. Sprints last no more than one month each, and are scheduled one after the other without delays in between. Sprints are also timeboxed – that is, they end on a specific date whether "scheduled" work is completed or not. They are never extended.

At the beginning of each Sprint, cross-functional teams select "stories" (customer requirements expressed in a functional manner) from a prioritized list. The team commits to completing its chosen stories by the end of the Sprint. Chosen items do not change during the Sprint, and the team gathers each day to for no more than 15 minutes to check on progress and adjust the next steps needed to complete remaining work.

At the end of each Sprint, the team reviews completed work with business stakeholders (demonstrating functionality). Feedback is incorporated into the next Sprint.

It should be apparent that Scrum emphasizes working product at the end of the Sprint that is really "done," meaning code that is integrated, tested and potentially shippable.

It's important to note that the scope of Sprint retrospectives are not limited to a review of the product. Scrum emphasizes short development phases where teams are constantly inspecting both the product and how well the process worked. It further emphasizes the importance of adapting product goals and process practices in a continuous mode. Rinse. Lather. Repeat.

Unlike other project management and team collaboration tools, ScrumDo was built for Scrum from the ground up. Once you know Scrum, you'll find ScrumDo one of the most intuitive & powerful Scrum managment tools in the marketplace.

Scrum Roles

In Scrum, there are three essential roles: Product Owner, ScrumMaster and Team Member.

The Product Owner represents the business side of the house – responsible for maximizing return on investment (ROI). Product Owners identify product features and constantly refine and / or re-prioritize them based on their importance to the business.

In actual day-to-day practice, business "value" remains a fuzzy term. Consequently, prioritization ends up being influenced by a variety of dynamic factors. In some organizations, the Product Owner and the customer are the same person (a very common state for internal business applications). In other organizations, the customer might be represented by a vast marketplace, with a Product Manager or Product Marketing Manager representing their interests. Either way, Scrum's Product Owner is different from a traditional Product Manager because they are called upon to actively and frequently interact with the Development Team as Sprints progress. Product Owners are the ultimate authority of what work gets down and when, as they are solely responsible for the value of that work as it relates to the business.

Team Members are the individuals responsible for actually building the product defined by the Product Owner. Typically consisting of 5-9 people, its members are cross-functional in nature, meaning all of the expertise required to build and deliver the potentially shippable product is resident on the team each Sprint. So for a software product, the Team might include individuals with skills in analysis, development, testing, UI design, database design,etc. Teams are intended to be "self-organizing" (self-managing), carrying a high degree of autonomy and accountability.

The Team decides on what stories it will commit to delivering each Sprint, plus how best to accomplish that commitment. They tend to be most productive when all members work exclusively on one product during the Sprint and don't multi-task across multiple products or projects. Also, teams with infrequent membership changes are associated with higher productivity. Application groups with many people are often organized into multiple teams focused on different product features (with a close coordination of effort).

Scrum Masters hold limited but essential roles in the Scrum process. They are not team or project managers. Unlike a project manager, the Scrum Master does not tell individual team members what to do or assign tasks. Their role is to simply facilitate the process, supporting the Team as it organizes and manages itself.

Scrum Masters should also educate and guide Product Owners, Team members, and others within the business on how to use the Scrum process effectively. One of the Scrum Master's most important roles is to protect the team from outside interference.

As Scrum exposes many risks and impediments to a successful Sprint, an effective Scrum Master will help resolve those issues. Although it's ideal to have a full-time Scrum Master, smaller teams will often have one of their members serve in this role (and carrying a lighter work load as a result).

It's important that the Scrum Master and the Product Owner aren't the same person. For one reason, the Scrum Master is sometimes called upon to push back on the Product Owner (for example, if they ask the team to work on a new deliverable in the middle of a Sprint). Another lies in the fact that they may not have a sufficient connection to the business issue sto prioritize and refine work definitions effectively.

If a Scrum Master were previously in a management position, they will need to significantly change their mindset and style of interaction to be successful. Sometimes an (ex-)project manager can step into the role, , but this has met with mixed success.

Starting with Scrum

In order to begin developing software using Scrum, the first thing to do (after forming your teams and assigning roles) is for the Product Owner to define the product and ultimately create a prioritized list of features. This is called the "Product Backlog." At any point, the Product Backlog is the single, definitive view of all work that could ever be completed by the Team (in order of priority). It should evolve over the lifetime of the product.

A well-defined Product Backlog will incorporate a variety of items. Most will be new features, but others can include improvement efforts, research, and known defects.

A common myth concerning Scrum is that it is devoid of detailed specifications. In practice, however, the Product Owner and Team members ultimately dictate how much detail is included for each item. This can and often will vary from one item to the next. Low priority items (and those far from being implemented) typically possess fewer details, whereas higher priority items (and those soon to be worked on) tend to have more detail.

The portion of the Product Backlog intended for the current release is known as the Release Backlog, and is usually the primary focus of the Product Owner. In addition to continuously refining and re-prioritizing the backlog based on new ideas, customer feedback, competitive pressures, and similar factors, he or she should assign an estimated business value estimate to each item. This will often be an unfamiliar practice to new Product Owners, and is something a good Scrum Master will offer guidance.

The Team is responsible for providing estimates of the effort required for each item on the Product Backlog. The Product Owner should use the estimates of effort and value (along with risk estimates and other factors) to inform his or her continual refinement of the backlog. In practice, estimates are often refreshed each Sprint as the team learns.

Over time, a Team should be able to track how much work it can do each Sprint. This information can then be used to project a release date to complete all features, or how many features can be completed by a certain date (assuming no changed circumstances). In Sprint we call this the "velocity" of the team. Velocity is typically expressed in the units utilized to create Backlog item size estimates. New users of Scrum should note the value of various estimating techniques devised over the years has come into some question.

Sprint Planning

A Sprint Planning Meeting should be help prior to the beginning of each Sprint. Effective sessions usually consist of two stages. The first stage focuses on understanding what the Product Owner wants. The Product Owner and Team will review the highest priority items in the Backlog, discuss the goals and context of these items, and review the "Definition of Done" each must meet (though these standards should have been previously established). According to established rules of Scrum, the Product Owner may leave the session at the conclusion of this review, but should be available to answer questions and provide clarification on any issues that arise during the next stage.

The second half of the Planning process focuses on detailed task planning. The team will often begin by estimating how much time each member has for Sprint-related work – in other words, their average availability to perform work less time spent attending meetings, managing email, etc. This typically entails 4-6 hours per day for most team members, and represents the team's capacity for the upcoming Sprint.

The Team then selects those items from the Product Backlog they believe they can complete by the end of the Sprint (starting at the top of the Product Backlog and working down the list in order). Having the team select how much work it will commit to complete rather than having it assigned to them by the Product Owner makes for a more reliable commitment because the Team is making it based on its own analysis and planning.

Although the Product Owner does not have control over how many items the Team commits to complete, he or she knows the work to which the team commits is at least drawn from the top of the Product Backlog (the items he or she has rated as most important). The Team can ask to complete lower priority items when there's agreement that it makes sense to do so.

Many Teams will begin making use of a visual task-tracking tool (in the form of a wall-sized task board) where the progress of work will be tracked across columns (often labeled "To Do", "Doing" and "Done").

Once the Team makes its Sprint commitment, any additions or changes must be deferred until the next Sprint. If an external circumstance arises to significantly change priorities, the Product Owner or Team has the option of terminating the Sprint. This dictates a new Sprint Planning meeting to initiate a new Sprint. The attending disruption of doing this is significant, and serves as a huge disincentive to make such a decision except in the rarest of circumstances.

ScrumDo has many features that make release planning, sprint planning, backlog grooming and other common management tasks a snap! And it doesn't matter whether team members sit across the room, across town or across the globe!

Daily Scrum

Once the Sprint has started, the Team will conduct a daily Scrum – a short, 10-15 minute meeting that takes place every workday at an appointed time. The team should synchronize their work and report to each other on any impediments.

Each member of the Team should report on only three) things:

(1) what they were able to complete since their last meeting;
(2) what they're planning to finish by the next meeting; and
(3) any obstacles in their way.

The Daily Scrum should ***not*** be a status meeting for management. There should be no discussion during the Daily Scrum – only reporting answers to the three questions. If discussion are required, they should occur immediately after the Scrum in among necessary parties.

It is generally recommended that managers or others in positions of authority ***not*** attend the Daily Scrum. Doing otherwise can cause the Team to feel "monitored" and / or under pressure to report major progress every day (an unrealistic expectation). More importantly, it can inhibit team embers from reporting problems.

Ending the Sprint

Sprints are never extended. They end on their assigned date regardless of whether the Team has completed the work to which it committed. New teams tend to over-commit during their first several Sprints (thus failing to achieve their commitments). Teams experiencing this should be careful to avoid over-compensating by under-committing in later Sprints. Most teams should be capable of establishing a reliable rhythm by their third or fourth Sprint.

Sprint Review and Retrospective

The Team and Product Owner should review the completed work at the end of each Sprint. It's an opportunity for the Product Owner to learn about what's going on with the product and for the Team to learn what's going on with the Product Owner and the market.

One trap for the unwary is for the ScrumMaster to fail to assure everyone is aware of the "Definition of Done" as defined for the current product or release. He or she should prevent the team from demonstrating or discussing Product Backlog

Items that are not done according to their definition. Items that aren't done should go back to the Product Backlog where they will be re-prioritized by the Product Owner.

There is a distinction between the Sprint Review and the Sprint Retrospective. The first is all about the product. The second is all about the process. It's an opportunity for the Team to discuss what's working and what's not working, and agree on changes to try. Unfortunately, this practice is often skipped by many teams, which deprives them of any opportunity to identify areas of potential improvement and transform them into results.

Doing it all again

There is no down time between Sprints. Following the Sprint Review and Retrospective, the Product Owner will update the Product Backlog based on new insights (if warranted), and the team moves right into its next Planning Session. It's important for both teams and the business to acknowledge a core agile principle here, and that's the notion of "sustainable pace." Only by working regular hours at reasonable levels will Teams be able to continue this cycle indefinitely.

Thanks for Downloading our Guide!

Once you're ready to start using Scrum, we hope you'll visit us again and consider using ScrumDo as your management and collaboration tool.

Take advantage of our free trial, and you'll soon understand why thousands of Scrum teams in companies from Apple to Zylun use ScrumDo to help manage their work!

http://scrumdo.com/