June 2016

Scrum.org | Whitepapers

An Introduction to the Nexus™ Framework

Simon Bourk & Patricia Kong

Preface

In today's fast pace of living, technology moves at light speed. The industry has had to deploy an impressive workforce to respond to the pace. Product development teams are being created with staff ranging from a few software developers building the product to product teams made up of thousands of software developers to just maintain the trend.

Historically, the most common way to manage software development is sequentially with a "waterfall method." In waterfall, the application development lifecycle starts with defining all of the users' needs which would then lead to the design of the entire system that would drive the implementation of the code. The code would be verified at the end during the testing phase before the system would be released into production.

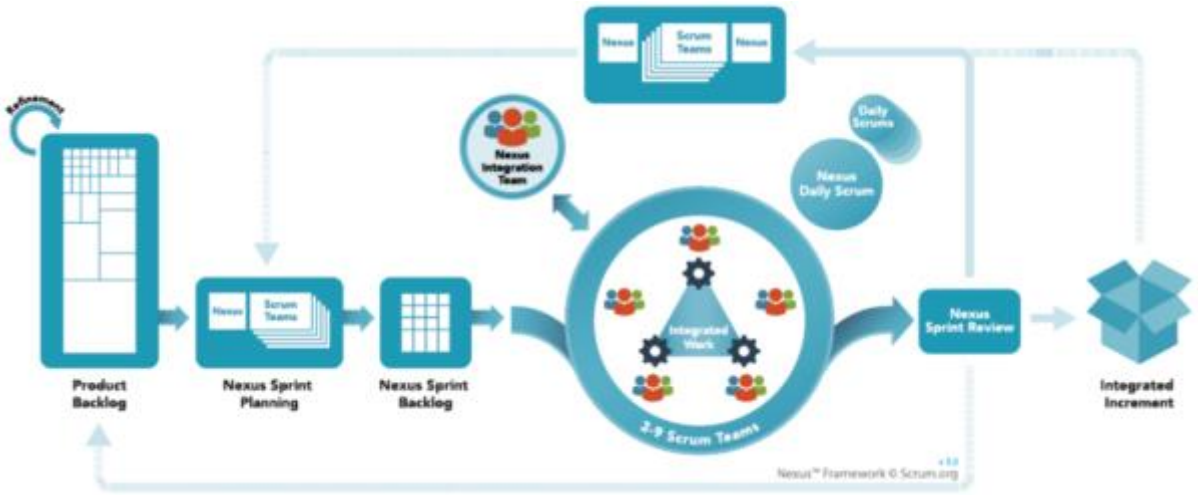
Over the years, the industry has found that following such a waterfall process has led to significantly more failures than successes. It has been difficult to estimate the timeframe and cost of a project. When time went on and on based off a stale set of requirements, the final delivery, if it ever happens, no longer meets the needs of users. At best, they would end up with a buggy product that didn't "age" well, years after the initial request.

Scrum has been proven to be a great framework for developing software in an agile way for complex environments, avoiding many of the issues that come from waterfall. It's based off empirical cycles and is simple for a small number of teams.

Organizations need to build products that add higher value to customers and the business while doing it better, less expensively, and faster in order to be more competitive and lucrative. Development organizations have naturally been addressing this challenge by trying to scale with more developers and managers. However, experience and research shows that the more teams working on the same product, the harder it is to coordinate the workflow and collaboration between those teams.

It is easier for teams with fewer members to communicate and self-organize. Multiplying the number of teams working on the same product will add complexity in communication, collaboration and self-organizing of the whole. Therefore, it is recommended that the original team inspect and perfect their workflow before expanding the work to more teams. This will increase their chances of success scaling up.

Introducing the Nexus Framework



The Nexus framework was created by Ken Schwaber, co-creator of the Scrum framework, and was released by his organization, Scrum.org, along with a body of knowledge, the Nexus Guide[1], in 2015. Nexus is based on the Scrum framework and uses an iterative and incremental approach to scaling software and product development. Nexus augments Scrum minimally with one new role and some expanded events and artifacts. It preserves Scrum. As Ken describes it, "Nexus is the exoskeleton of Scrum."

The Nexus framework is a foundation to plan, launch, scale and manage large product and software development initiatives. Nexus is for organizations to use when multiple Scrum Teams are working on one product as it allows the teams to unify as one larger unit, a Nexus. As an "exoskeleton," a Nexus protects and strengthens the teams by creating connections between them and by maintaining bottom-up intelligence. The foundation of a Nexus is to encourage transparency and keep scaling as uniform as possible.

A Nexus consists of 3-9 Scrum Teams working on a single Product Backlog to build an Integrated Increment that meets a goal. The framework is especially beneficial for organizations who have had positive results using Scrum with one or two teams working on a single product and who want to scale more broadly. On the flip side, it is also extremely useful for those struggling with a scaled initiative as the framework focuses rigorously on two core scaling matters, cross-team dependencies and integration issues.

The Nexus framework introduces one new role, the **Nexus Integration Team (NIT).** The NIT is responsible for ensuring that an Integrated Increment (the combined work completed by a Nexus) is produced at least every Sprint. It is ultimately accountable for the successful integration of all work created by all the Scrum Teams in a Nexus. Common activities may include identifying cross-team issues, raising awareness of dependencies early, and ensuring integration tools and practices are understood and used. In emergency situations, the NIT may actually do the work in order to fulfil their accountability.

[1] https://www.scrum.org/Resources/The-Nexus-Guide

The Nexus Integration Team is a Scrum Team, and membership may be full or part time. NIT membership should take precedence in order to give priority to, and help resolve issues that may affect multiple teams. The Product Owner and a Scrum Master who can serve as the Scrum Master of the Nexus are on this team. There are also Nexus Integration Team Members who are often members of Scrum Teams within a Nexus, but not always exclusively, as it depends on what makes sense for an organization's needs. Composition of the NIT may change over time with exception of the Product Owner.

The NIT works off the Product Backlog and takes ownership of any integration issues. Integration includes resolving any technical and non-technical cross-team concerns. The NIT should use bottom-up intelligence from the Nexus to achieve resolution.

The events in Nexus are the same as in Scrum with the formalization of Refinement. Product Backlog **Refinement** is a formal event that happens during the Sprint in the Nexus framework because of the added complexity of multiple teams working together. The number, frequency, and duration of Refinement meetings will vary based on the dependencies in the Product Backlog. The goal of Refinement is (1) to decompose the Product Backlog items enough so that the Nexus can figure out which teams might deliver them and in what sequence and (2) to identify, visualize, and minimize cross-team dependencies. With those goals in mind, it is important that the appropriate members of the individual Scrum Teams attend and participate in these meetings.

Refinement happens as much as needed during the Sprint. The Product Backlog should be refined enough so that dependencies are identified, removed or minimized enough before Nexus Sprint Planning, because dependencies get bigger and more entwined at scale. To minimize dependencies, it would be important to look at team structure and architectural structure, visualize cross-team dependencies, and focus on modern technical practices.

Nexus Sprint Planning is when the Nexus creates a plan for the next Sprint. It is also when the **Nexus Sprint Goal** is formulated. The Nexus Sprint Goal is an objective that all the Scrum Teams in the Nexus work to achieve during the Sprint. The first part of Nexus Sprint Planning is when the appropriate representatives (the right people) come together to identify, minimize and resolve in-Sprint dependencies. Each Scrum Team needs to be represented, and they will reorder the work as needed. These representatives then join their individual Scrum Teams to do their individual Scrum Team Sprint Planning. This event is finished when the last Scrum Team's Sprint Planning is done.

An artifact resulting from Nexus Sprint Planning is a **Nexus Sprint Backlog.** It is a visualization of any dependencies that will exist during the Sprint, and a way to manage the flow of work in a Nexus. The Nexus Integration Team creates and manages this backlog. It should be reviewed and updated at least daily, especially if the backlog is used to manage/visualize the flow of work across teams in a Nexus.

An opportune time to look at the Nexus Sprint Backlog would be at the **Nexus Daily Scrum.** This is a brief meeting that happens before each individual Scrum Teams' Daily Scrums. Again, not everyone needs to attend, but appropriate representatives from the Scrum Teams must. The purpose of this event is to make integration issues transparent at least once a day. Inspecting the state of the integrated work will allow the teams to plan correctly. This is why the Nexus Daily Scrum happens before the individual Daily Scrums. The output of the Nexus Daily Scrum becomes the input into each individual team's Daily Scrum so that they can plan their next day's work.

The **Nexus Sprint Review** replaces the individual Scrum Team Sprint Reviews. The Scrum Teams of a Nexus deliver this review together so that they can present the fully **Integrated Increment** of work for feedback, and not separate parts. This one event will not replace the need for the Product Owner to see Product Backlog Items as they are being developed, but it will need different practices to ensure that the right information is being shown to the audience.

Like Scrum, there is a formal event for the Nexus to inspect and adapt during the **Nexus Sprint Retrospective.** Nexus Sprint Retrospective occurs after the Nexus Sprint Review and before Nexus Sprint Planning. It is a 3 part event. To begin, representatives from each Scrum Team meet to identify shared challenges. These are typically obvious. Output from this meeting will become input for the second part of the Nexus Sprint Retrospective, which is within the individual teams. Along with coming up with solutions for the common issues within the Nexus, the Scrum Teams should have their individual retrospectives during the second part of the Nexus Sprint Retrospective. The last part of the event is when the representatives rejoin to discuss any actions on those shared challenges. Given the size of a Nexus, visualization techniques are helpful to track items and progress.

A Nexus works within the boundaries of a **Sprint** (30 days or less), as in Scrum. If absolutely necessary, teams can work on different Sprint lengths, but there must be a common denominator. Teams must deliver into the increment at least at the end of each of their Sprints.

Like Scrum, it is easy to get started with Nexus; however, the Nexus Framework does first and foremost presume Scrum experience. Organizations that are already familiar with Scrum will be able to benefit from their existing Scrum knowledge. As with Scrum, Nexus alone isn't enough for success, but it helps drive to the heart of the scaling issue – continually identifying and removing dependencies created by increased complexity. In addition to using Nexus, organizations should establish, promote, and steward technical excellence as a foundation for growth. To start a Nexus, organizations should first:

- Identify the teams in their Nexus
- Form an initial Nexus Integration Team
- Have a single Product Backlog
- Have a definition of "Done"
- Identify a Sprint cadence

Nexus+

For larger initiatives you can form the Nexus+™ (using 10+ teams). Nexus+ is more than one Nexus interoperating to build a large product, usually integrated through a product family framework, api, or similar integrating technology.

At this scale, organizations will have implemented their own personal flavors of the sets of practices and guidelines in pursuing shorter feedback loops. Everything you've learned implementing the Nexus applies to implementation of the Nexus+.

You certainly want to avoid having dependencies across Nexuses. Those cross-dependencies will add additional complexities in coordination, collaboration and communication. Also, take advantage of the technologies or architectural patterns existing today to avoid these cross-dependencies.

Conclusion

Software development can be complex and hard to do. In trying to stay current with the technology and the keeping on top of the demand, larger organizations are facing a breaking point that can hinder their ability to keep the pace.

Organizations can increase their speed to delivering valuable software by using an empirical process. It is clear Scrum has brought that advantage to the community and scaling Scrum with the Nexus framework and forming a Nexus can get us to the next level.