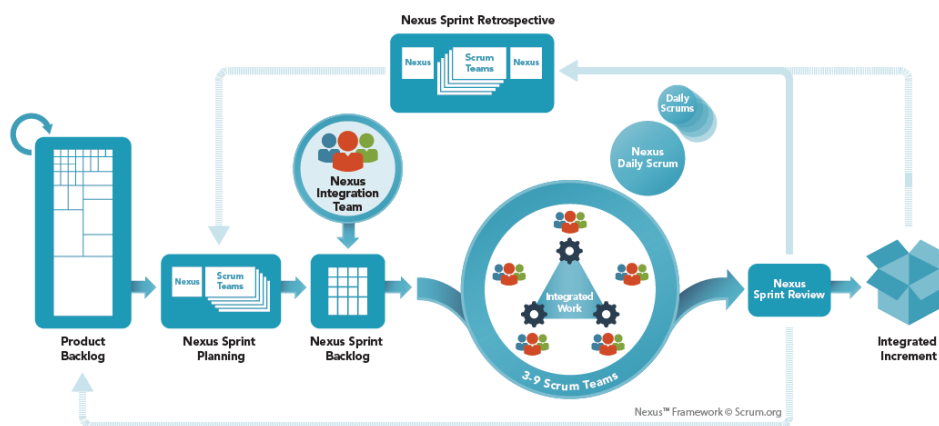# Nexus™ Guide

The Definitive Guide to Nexus:
The exoskeleton of scaled Scrum development

July 2015

Developed and sustained by Ken Schwaber and Scrum.org

Table of Contents

Nexus Overview

Purpose of the Nexus Guide

Nexus is a framework for developing and sustaining scaled product and software development initiatives. It uses Scrum as its building block. This Guide contains the definition of Nexus. This definition consists of Nexus roles, events, artifacts, and the rules that bind them together. Ken Schwaber and Scrum.org developed Nexus. The Nexus Guide is written and provided by them.

Definition of Nexus

Nexus (n): Unit of development (in Scaled Professional Scrum)

Nexus is a framework consisting of roles, events, artifacts, and techniques that bind and weave together the work of approximately three to nine Scrum Teams working on a single Product Backlog to build an Integrated Increment that meets a goal.

Nexus Background

Software development is complex and the integration of that work into working software has many artifacts and activities that must be coordinated to create a done outcome. The work must be organized, sequenced, the dependencies resolved, and the outcomes staged. One person may tie a knot; when several people attempt to cooperatively tie the knot, the outcome becomes uncertain. Software presents additional difficulties since it is not physically present.

Many software developers have used the Scrum framework to work collectively as a team to develop an Increment of working software. However, if more than one Scrum Team is working off the same Product Backlog and in the same code base, difficulties arise. If the developers are not in the same collocated team, how will they communicate when they are doing work that will affect each other? If they work in different teams, how will they integrate their work and test the Integrated Increment? These challenges appear when two teams are integrating, and become significantly more difficult with three or more teams.

There are many dependencies that arise between the work of multiple teams that collaborate to create a complete and "Done" increment at least every Sprint. These dependencies are related to:

1. Requirements: The scope of the requirements may overlap, and the manner in which they are implemented may also affect each other. That knowledge should be taken into account when ordering the Product Backlog and selecting requirements.
2. Domain knowledge: The people on the teams have knowledge of various business and computer systems. That knowledge should be mapped to the Scrum Teams to ensure its adequacy and also to minimize interruptions between teams during a Sprint.
3. Software and test artifacts: The requirements are or will be instantiated in software code and test suites.

To the extent that requirements, team members' knowledge, and code/test artifacts are mapped to the same Scrum Teams, dependency between teams can be reduced.

When software development using Scrum is scaled, these dependencies of requirements, domain knowledge, and software/test artifacts should drive team organization. To the extent that it does, productivity will be optimized.

Nexus Framework

Nexus is an exoskeleton that rests on top of multiple Scrum Teams when they are combined to create an Integrated Increment. Nexus is consistent with Scrum and its parts will be familiar to those who have worked on Scrum projects. The difference is that more attention is paid to dependencies and interoperation between Scrum Teams, generating one integrated, "Done" increment at least every Sprint.

Nexus consists of:

1. Roles: A new role, the Nexus Integration Team, exists to coordinate, coach, and supervise the application of Nexus and the operation of Scrum so the best outcomes are derived. The Nexus Integration Team consists of a Product Owner, a Scrum Master, and Nexus Integration Team Members.
2. Artifacts: All Scrum Teams use the same, single Product Backlog. As the Product Backlog items are refined and made ready, indicators of which team will do the work inside a Sprint are made visual. A new artifact, the Nexus Sprint Backlog, exists to assist with transparency during the Sprint. All Scrum Teams maintain their individual Sprint Backlogs.
3. Events: Events are appended to or placed around regular Scrum events to augment them. As modified, they serve both the overall effort of all Scrum Teams in the Nexus, and each individual team.

These are displayed in the following graphic:

Nexus™ Framework, exoskeleton of scaled Scrum
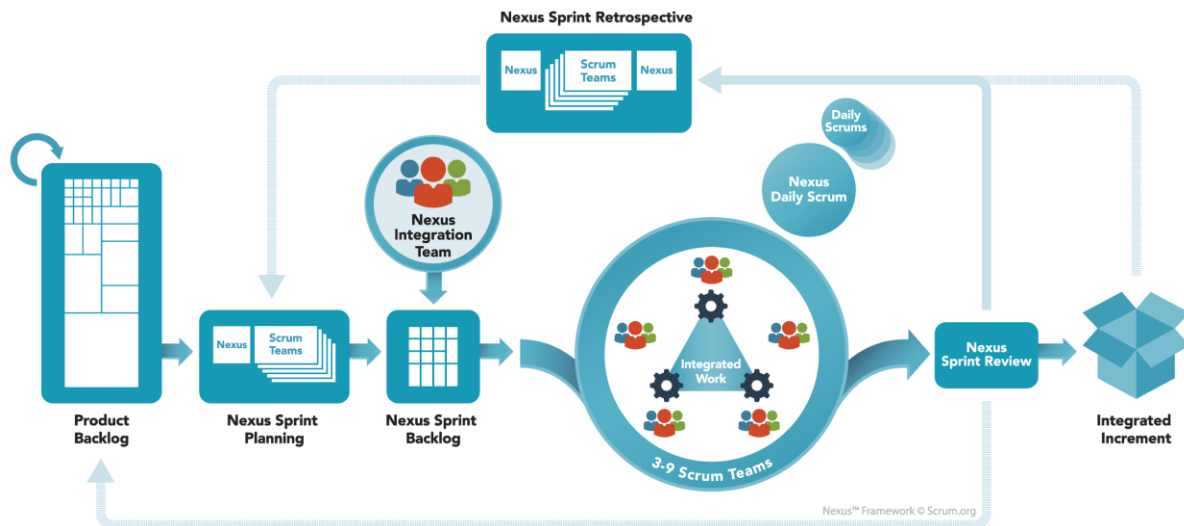
Nexus Process Flow

All work in a Nexus may be done by all members of the Scrum Teams, as cross-functional members of the Nexus. Based on dependencies, the teams may select the most appropriate members to do specific work.

1. Refine the Product Backlog: The Product Backlog needs to be decomposed so that dependencies are identified and removed or minimized. Product Backlog items are refined into thinly sliced pieces of functionality and the team likely to do the work should be identified as early as possible.
2. Nexus Sprint Planning: Appropriate representatives from each Scrum Team meet to discuss and review the refined Product Backlog. They select Product Backlog items for each team. Each Scrum Team then plans its own Sprint, interacting with other teams as appropriate. The outcome is a set of Sprint Goals that align with the overarching Nexus Goal, each Scrum Team's Sprint Backlog and a single Nexus Sprint Backlog. The Nexus Sprint Backlog makes the Scrum Team's selected Product Backlog items and any dependencies transparent.
3. Development work: All teams develop software, frequently integrating their work into a common environment that can be tested to ensure that the integration is done.
4. Nexus Daily Scrum: Appropriate representatives from each Scrum Team meet daily to identify if any integration issues exist. If identified, this information is transferred back to each Scrum Team's Daily Scrum. Scrum Teams then use their Daily Scrum to create a plan for the day, being sure to address the integration issues raised during the Nexus Daily Scrum.
5. Nexus Sprint Review: All teams meet with the Product Owner to review the Integrated Increment. Adjustments may be made to the Product Backlog.

6. Nexus Sprint Retrospective: Appropriate representatives from each Scrum Team meet to identify shared challenges. Then, each Scrum Team holds individual retrospectives. Appropriate representatives from each team meet again to discuss any actions needed based on shared challenges.

Software Practices

Many software practices are needed to bind the work of Scrum Teams collaborating together to create an Integrated Increment. Most of these practices require automation. The automation helps manage the volume and complexity of the work and artifacts.

Nexus

Nexus roles, events, and artifacts inherit the purpose and intent attributes of the corresponding Scrum roles, events, and artifacts, as documented in the Scrum Guide.

Nexus Roles

A Nexus consists of a Nexus Integration Team and approximately three to nine Scrum Teams.

Nexus Integration Team

The Nexus Integration Team is accountable for ensuring that an Integrated Increment is produced at least every Sprint. The Scrum Teams are responsible for developing Increments of potentially releasable software, as prescribed in Scrum. All roles for members of the Scrum Teams are prescribed in the Scrum Guide.

The Nexus Integration Team is a Scrum Team that consists of:

- The Product Owner
- A Scrum Master
- One or more Nexus Integration Team Members

People in the Nexus Integration Team may also work on the Scrum Teams in a Nexus, as appropriate and necessary. If this is the case, priority must be given to the work for the Nexus Integration Team. Membership in the Nexus Integration Team takes precedence over individual Scrum Team membership. This preference helps ensure that the work to resolve issues that affect many teams has priority.

Composition of the Nexus Integration Team may change over time to reflect the current needs of a Nexus. Common activities the Nexus Integration Team might perform include coaching, consulting, and highlighting awareness of dependencies and cross-team issues. It might also perform work from the Product Backlog.

The Nexus Integration Team takes ownership of any integration issues. It is accountable for successful integration of all work by all Scrum Teams in a Nexus. Integration includes any

technical and non-technical cross-team constraints that may impede a Nexus' ability to deliver a constantly Integrated Increment. They should use bottom-up intelligence from the Nexus to achieve resolution.

Product Owner in the Nexus Integration Team

A Nexus works off a single Product Backlog, and as described in the Scrum framework, a Product Backlog has a single Product Owner who has the final say on its contents. The Product Owner is responsible for maximizing the value of the product and the work performed and integrated by the Scrum Teams. The Product Owner is in the Nexus Integration Team.

The Product Owner is accountable for ordering and refining the Product Backlog so that maximum value is derived from the Integrated Increment created by a Nexus. How this is done may vary widely across organizations, Nexuses, Scrum Teams, and individuals.

Scrum Master in the Nexus Integration Team

The Scrum Master in the Nexus Integration Team has overall responsibility for ensuring the Nexus framework is understood and enacted within all Scrum Teams. This Scrum Master may also be a Scrum Master for one or more of the individual Scrum Teams.

Nexus Integration Team Members

Scaled development work requires tools and practices that individual Scrum Teams may not frequently use. The Nexus Integration Team consists of software professionals who are skilled in the use of these practices, tools, and the general field of systems engineering. Nexus Integration Team Members ensure that practices and tools are implemented, understood, and used to detect dependencies and frequently reify all artifacts to a definition of "Done."

Nexus Integration Team Members are responsible for coaching and guiding the Scrum Teams in a Nexus to acquire, implement, and learn these practices and tools. Additionally, they coach the Scrum Teams on the necessary development, infrastructural, or architectural standards required by the organization to ensure the development of quality Integrated Increments.

If their primary responsibility is satisfied, Nexus Integration Team Members may also work as Development Team members in one or more Scrum Teams.

Nexus Events

Event Duration

The duration of Nexus events is guided by the length of the corresponding events in the Scrum Guide.

Nexus Sprint Planning

The purpose of Nexus Sprint Planning is to coordinate the activities of all Scrum Teams in a Nexus for a single Sprint. The Product Owner provides domain knowledge and guides selection and priority decisions.

To begin Nexus Sprint Planning, appropriate representatives from each Scrum Team validate and make adjustments to the ordering of the work as created during Refinement events. All members of the Scrum Teams should participate to minimize communication issues if possible.

Once the overall work for the Nexus is understood, each Scrum Team will perform their own separate Sprint Planning. If the meeting is conducted in one collocated space, the teams can continue to share newly found dependencies. Nexus Sprint Planning is complete when each Scrum Team has finished their individual Sprint Planning events.

New dependencies may emerge during Nexus Sprint Planning. They should be visualized and minimized. The sequence of work across teams may also be adjusted. An adequately refined Product Backlog will minimize the emergence of new dependencies during Nexus Sprint Planning. All Product Backlog items selected for the Sprint and their dependencies should be visualized on the Nexus Sprint Backlog.

The Product Backlog should be adequately refined with dependencies identified and removed prior to the Nexus Sprint Planning. If this work has not been performed, it must be done in this meeting before the development work of the Sprint can begin.

Nexus Daily Scrum

The Nexus Daily Scrum is an event for appropriate representatives from individual Scrum Teams to inspect the current state of the Integrated Increment and to identify integration issues or newly discovered cross-team dependencies.

During the Nexus Daily Scrum, appropriate representatives should focus on each team's impact on the Integrated Increment and discuss:

- Was the previous day's work successfully integrated?
  - *If not, why not?*
- What new dependencies have been identified?
- What information needs to be shared across teams in the Nexus?

During the Nexus Daily Scrum, the Nexus Sprint Backlog should be used to visualize and manage current dependencies.

Work that is identified during the Nexus Daily Scrum is then taken back to individual Scrum Teams for planning inside their Daily Scrum events.

Nexus Sprint Review

The Nexus Sprint Review is used to provide feedback on the Integrated Increment that a Nexus has built over the Sprint.

A Nexus Sprint Review replaces the individual Scrum Team Sprint Review, because the entire Integrated Increment is the focus for capturing feedback from stakeholders. It may not be possible to show all completed work in detail. Techniques may be necessary to maximize stakeholder feedback.

Nexus Sprint Retrospective

The Nexus Sprint Retrospective is a formal opportunity for a Nexus to focus on inspection and adaptation. It consists of three parts:

1. The first part is an opportunity for appropriate representatives from across a Nexus to meet and identify issues that have impacted more than a single team. The purpose is to make shared issues transparent to all Scrum Teams.
2. The second part consists of each Scrum Team holding their own Sprint Retrospective. They can use the issues raised from the first part of the Nexus Retrospective as input to their team discussions. The individual Scrum Teams should form actions to address these issues during their individual Scrum Team Retrospectives.
3. The final, third part is an opportunity for appropriate representatives from the Scrum Teams to meet again and agree on how to visualize and track the identified actions. This allows the Nexus as a whole to adapt.

Because they are common scaling dysfunctions, the following questions should be addressed at every Retrospective:
- Was any work left undone? Did the Nexus generate technical debt?
- Were all artifacts, particularly code, frequently (as often as every day) successfully integrated?
- Was the software successfully built, tested, and deployed often enough to prevent the overwhelming accumulation of unresolved dependencies?

For the questions above, address:
- Why did this happen?
- How can it be undone?
- How can the recurrence be prevented?

Refinement

At the scale of Nexus there are many levels of refinement. Only when the Product Backlog items are adequately independent can they be selected and worked on without excessive conflict between Scrum Teams in a Nexus.

The number, frequency, duration and attendance of Refinement meetings are based on the dependencies inherent in the Product Backlog. The greater the complexity and dependencies, the more the Product Backlog items must be refined to remove dependencies. Product Backlog items pass through different levels of decomposition from very large requests to actionable work that a single team could deliver inside a Sprint.

Product Backlog refinement at scale serves a dual purpose. It forecasts which team will deliver which Product Backlog items, and it identifies dependencies across those teams. Visualization allows the teams to minimize dependencies.

The first part of cross-team Refinement should be spent decomposing Product Backlog items into enough detail in order to understand which teams might deliver them, and in what sequence over upcoming Sprints.

The second part of Refinement should be spent focused on dependencies. Dependencies should be identified and visualized across teams and time. Teams will need this information to re-order the sequencing and allocation of their work in order to minimize the number of cross-team dependencies.

Enough Refinement meetings have been performed during a Sprint if the Product Backlog items are ready and selectable with minimum dependencies during the Sprint Planning meeting.

Nexus Artifacts

Product Backlog

There is a single Product Backlog for the entire Nexus and all of its Scrum Teams. The Product Owner is responsible for the Product Backlog, including its content, availability, and ordering.

At scale, the Product Backlog must be understood at a level where dependencies can be detected and minimized. To support resolution, Product Backlog items are often resolved to a granularity called "thin sliced" functionality. Metadata should be added to track and manage dependencies between the Product Backlog items. For example:

PBI	Initiative	Capability	Feature Set	Code Set	Team
Add QR codes to product list	Hands off	QR	Cart	Seg 4	Charlie
Twitter feed on home page	24/7	Twitter	Dashboard	Seg 7, 8	SocialQ

Product Backlog items are deemed "ready" for the Nexus Sprint Planning meeting when they can be selected to be done by Scrum Teams with no or minimal dependencies with other Scrum Teams.

Nexus Goal

A Nexus Goal is used to unite the Scrum Teams within a Nexus toward a shared purpose. A Nexus Goal is an overarching Sprint objective used to provide focus to a Nexus. Implementation of the Sprint Increment fulfills the goal.

Nexus Sprint Backlog

A Nexus Sprint Backlog is the composite of all the Sprint Backlogs of the individual Scrum Teams. It is used to highlight dependencies and the flow of work during the Sprint. It is updated at least daily, often as part of the Nexus Daily Scrum.

Integrated Increment

The Integrated Increment represents the sum of all integrated work completed by a Nexus. The Integrated Increment must be usable and potentially releasable which means it must meet the definition of "Done". The Integrated Increment is inspected at the Nexus Sprint Review.

Artifact Transparency

Just like its building block, Scrum, Nexus is based on transparency. A Nexus Integration Team works with the Scrum Teams within a Nexus and the organization to ensure that transparency is apparent across all artifacts and that the integrated state of the Increment is widely understood.

Decisions made based on the state of Nexus artifacts are only as effective as the level of artifact transparency. Incomplete or partial information will lead to incorrect or flawed decisions. The impact of those decisions can be magnified at the scale of Nexus. A lack of complete transparency will make it impossible to guide a Nexus effectively in order to minimize risk and maximize value.

Software must be developed so that dependencies are detected and resolved before technical debt becomes unacceptable. The test of unacceptable technical debt is when integration occurs, and it remains unclear that all dependencies are resolved. In these cases, the unresolved dependencies remain tucked in the code and test base, lowering the overall value of the software.

Definition of "Done"

The Nexus Integration Team is responsible for a definition of "Done" that can be applied to the Integrated Increment developed each Sprint. All Scrum Teams of a Nexus adhere to this definition of "Done". The Increment is "Done" only when usable and potentially releasable by the Product Owner.

A Product Backlog item can be considered "Done" when that functionality has been successfully added to the product and integrated into the Increment. All Scrum Teams are responsible for developing and integrating their work into an Increment that satisfies these attributes.

Individual Scrum Teams may choose to apply a more stringent definition of "Done" within their own teams, but cannot apply less rigorous criteria than agreed for the Increment.

End Note

Nexus is free and offered in this Guide. As with the Scrum framework, Nexus' roles, artifacts, events, and rules are immutable. Although implementing only parts of Nexus is possible, the result is not Nexus.

Acknowledgement

Nexus and Scaled Professional Scrum were collaboratively developed by Ken Schwaber, David Dame, Richard Hundhausen, Patricia Kong, Rob Maher, Steve Porter, Christina Schwaber, and Gunther Verheyen.