The Nexus™ Framework for Scaling Scrum
This page intentionally left blank
The Nexus™
Framework for Scaling Scrum

CONTINUOUSLY DELIVERING AN INTEGRATED PRODUCT WITH MULTIPLE SCRUM TEAMS

Kurt Bittner
Patricia Kong
Dave West
To the members of the community of Professional Scrum Trainers, from whom we learn every day.
This page intentionally left blank
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreword</td>
<td>xi</td>
</tr>
<tr>
<td>Preface</td>
<td>xiii</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>xvii</td>
</tr>
<tr>
<td>About the Authors</td>
<td>xix</td>
</tr>
</tbody>
</table>

**Chapter 1**  
Introduction to Scaling Agile  
- Why Agile?  
- Why Scrum?  
  - What Is a Product?  
  - What Is Scrum?  
- Why Nexus?  
- Simplicity Is the Key to Scaling  
1 
2 
2 
3 
3 
5 
6

**Chapter 2**  
Introducing Nexus  
- What Is Nexus?  
- Nexus Extends Scrum  
- The Nexus Integration Team  
- Nexus Events  
  - Refinement  
  - Nexus Sprint Planning  
  - The Nexus Daily Scrum  
  - The Nexus Sprint Review  
7 
7 
8 
10 
12 
13 
13 
15 
15
### Contents

- Optional Practice: Connecting Product Backlog Items to Value Delivery 51
- Building the Nexus Sprint Backlog and Scrum Team Backlogs 52
- Closing 55

#### Chapter 5 Running a Sprint in Nexus 57
- The Nexus Daily Scrum 58
- Providing Transparency Inside and Outside the Nexus 61
  - Optional Practice: Product Backlog Treemap 62
  - Optional Practice: Visualizing Product Backlog Burndown and Velocity 63
- The Nexus Sprint Review 66
  - Optional Practice: Using the “Exposition” (Expo) Format for Nexus Sprint Reviews 67
  - Optional Practice: Using Offline Review Techniques for Nexus Sprint Reviews 67
- Nexus Sprint Retrospective 68
- Closing 73

#### Chapter 6 Evolving the Nexus 75
- Optional Practice: Organizing Scrum Teams around Features 77
- Optional Practice: Managing Code Like an Open-Source Project 78
- Optional Practice: Organizing Teams around Personas 81
- Expanding the Nexus Integration Team 83
- Updating and Refining the Product Backlog 83
- Nexus Sprint Planning, Revisited 85
- The Nexus Daily Scrum, Take Two 86
- The Nexus Sprint Review, Take Two 88
- The Nexus Sprint Retrospective, Take Two 89
  - Too Much Work, Not Enough Progress 90
  - Growing Technical Debt 91
  - Unavailable Product Owner 91
  - Inadequate Build and Test Automation 92
  - Forming a Plan to Improve 93
  - The Challenges of Scaling Scrum 94
- Closing 96
This book is excellent. It begins with a simple application of Nexus. It then describes its application in increasingly complex situations. The authors lay out the complexities, the problems they cause, and how one can apply Nexus to address them. They thread the ideas together with a case study. This is backed up by The Nexus Guide, the definitive body of knowledge.

But, why does Nexus even exist?

Scrum is a framework within which a team of people can address a complex problem to create an increment of value within a short period of time. Over 27 years, Scrum has proven its value in many applications.

However, Scrum is only designed for a single team. Situations often call for multiple teams with different capabilities to work together to create value. Organizations naturally want to build on the initial Scrum framework.

Over the years, I have worked with hundreds of organizations, adhering to the framework and values of Scrum while scaling its use to tens, hundreds, and even thousands of people working together to create a single outcome.
Many other Scrum practitioners have also done so. To the degree that we applied our prior knowledge, much of the productivity and value of Scrum was retained.

Based on my experiences and those of others that I work with in Scrum.org, I designed a defined framework for using many Scrum teams on a single product or problem. The result is Nexus, an exoskeleton that rests on top of many Scrum teams. Nexus provides information and management information for guiding their working together. As much productivity as possible is retained, methods of increasing productivity are described, and remediation techniques for resolving failures are included.

Read and learn more. Scrum on.

—Ken Schwaber
Our goal in writing this book was simple: to expose people already familiar with Scrum to a simple yet powerful way to apply the same Scrum concepts with which they are familiar to Products that demand the efforts of more than one team. More than 12 million people use Scrum every day, and many of those people work on large multi-team efforts. Nexus evolved to meet the needs of these people, and although it is being used by many organizations, no book describing it yet existed. We hope that the readers of this book will be able to apply use Nexus to scale, and perhaps even improve, their Scrum practices. As we like to say, “Scaled Scrum is still Scrum.”

Who Should Read This Book

Anyone who uses Scrum will benefit from reading this book, because at some point you will find that a single Scrum Team is no longer sufficient to deliver your Product. Adding teams sounds easy, but unmanaged inter-team dependencies quickly overwhelm a merely intuitive approach. This book will help every team member understand Nexus better. Beyond the Scrum Teams, stakeholders for Scrum Teams will find this book helpful in understanding the challenges that multi-team efforts face, and it will help them to better support the teams with whom they work.
How This Book Is Organized

This book assumes that you are already familiar with the Scrum Framework and builds on that knowledge by explaining how to scale Scrum to develop a large product using Nexus.

Chapter 1, “Introduction to Scaling Agile,” does just that. It introduces you to the use of Agile in contexts that require more than one Scrum team working on a project.

Chapter 2, “Introducing Nexus,” focuses on the basic principles and concepts behind Nexus, including when you need a Nexus and what you need to get started.

Chapter 3, “Forming a Nexus,” focuses on how to form a Nexus around a product, even if that product is still only an idea without a team. For existing products and teams, we describe how to add teams while creating a Nexus. We also describe how you can organize the Scrum Teams in the Nexus and how to identify (and minimize) Product Backlog dependencies.

Chapter 4, “Planning in Nexus,” focuses on organizing the work of the Nexus: soliciting, refining, and validating a large backlog against business objectives; setting goals; and planning the Sprint.

Chapter 5, “Running a Sprint in Nexus,” focuses on the work of the Nexus during the Sprint: working with the Nexus Sprint Backlog, running the Nexus Daily Scrum, conducting Nexus Sprint Reviews, and conducting the Nexus Sprint Retrospective.

Chapter 6, “Evolving the Nexus,” focuses on managing the Nexus, including reporting progress, improving performance and throughput, and removing bottlenecks.

Chapter 7, “The Nexus in Emergency Mode,” focuses on how Nexus helps organizations overcome typical scaling challenges, including helping distributed teams work better together and responding to challenges that keep teams from working together effectively.
Chapter 8, “Retrospective on the Nexus Journey,” reflects on the typical journey that teams and organizations take when they scale Scrum. It looks at how the elements of Nexus help them on that journey, the typical challenges they face, and how they can overcome those challenges. It also looks ahead at things they can do to continue their journey of improving their ability to deliver complex applications.

Register your copy of *The Nexus™ Framework for Scaling Scrum* on the InformIT site for convenient access to updates and/or corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN (9780134682662) and click Submit. Look on the Registered Products tab for an Access Bonus Content link next to this product, and follow that link to access any available bonus materials. If you would like to be notified of exclusive offers on new editions and updates, please check the box to receive email from us.
This page intentionally left blank
Acknowledgments

We had a lot of help and support in writing this book. First, we have to thank Ken and Christina Schwaber for their support, encouragement, and perspective on how Nexus evolved from Scrum, and to thank Ken Schwaber and Jeff Sutherland for creating Scrum itself, upon which Nexus is based. The Nexus Framework exists because a collaborative team of people came together to translate their experiences into something that could be shared with everyone in the form of The Nexus Guide.

We are also indebted to the Professional Scrum Trainer community, whose members shared their valuable time helping to improve the book through their thoughtful suggestions and painstaking reviews. For their extensive contributions, our deepest gratitude goes to Peter Götz, Jesse Houwing, Richard Hundhausen, Ralph Jocham, Mikkel Toudal Kristiansen, Rob Maher, Jeronimo Palacios, and Steve Porter. Our thanks also extend to Eric Naiburg, whose careful writer’s eye helped us to express ideas more simply and effectively, and to Sabrina Love, who designed our cover.

Finally, this book would not be possible without the support we received from the team at Pearson/Addison-Wesley, notably our editor, Chris Guzikowski;
Acknowledgments

our development editor, Chris Zahn; our production editor, Julie Nahil; and our copy editor, Stephanie Geels, all of whom helped us to refine and publish the work you are reading.

—Kurt, Patricia, and Dave
Kurt Bittner has more than 35 years of experience helping teams to deliver software in short feedback-driven cycles, as a developer, as a product manager and product owner, as an industry analyst, and as an organizational change agent. He is the author of three other books on software engineering and many blogs and articles, and he is a frequent speaker at conferences.

Patricia Kong is a key contributor to the Nexus Framework and the Evidence-Based Management framework. She led product development, product management, and marketing for several early-stage companies in the U.S. and Europe, and she worked in business development and engagement management for Forrester Research. She is fluent in four languages.

Dave West is the CEO and Product Owner at Scrum.org. He is a frequent keynote speaker at major industry conferences and is a widely read author of books, blogs, articles, and research reports. He has led both product development and consulting organizations for multinational organizations.
This page intentionally left blank
Introducing Nexus

In this chapter, we describe the Nexus Framework in its entirety. As you will see, Nexus is a relatively small and simple extension of Scrum. As we like to say, “Scaled Scrum is still Scrum.” Scrum itself is quite simple, at least to understand. When scaling, this simplicity is a big advantage because complexity is the enemy of scaling. Nexus’ simplicity also makes it highly adaptable, as we will see in subsequent chapters.

What Is Nexus?

Nexus is a framework that enables multiple Scrum Teams to collaboratively work from a single Product Backlog to deliver at least one “Done” Integrated Increment every Sprint. “Multiple” means, typically, three to nine Scrum teams. Why not two? Because two teams can generally coordinate between one another without additional structure. Why nine? Just as Scrum recommends limiting teams to no more than nine members to improve cohesion and reduce complexity, Nexus recommends the same for the number of teams. Just as in Scrum, however, this upper limit is not absolute and slightly larger numbers may still work, depending on the circumstances. With Nexus we have
discovered that collaboration complexity and coordination between teams increases significantly beyond nine teams, and for those cases some different techniques apply.¹

Since Nexus builds on Scrum, its parts will be familiar to those who have used Scrum. The difference is that more attention is paid to dependencies and communication between Scrum Teams (see Figure 2-1).

---

**Figure 2-1** The Nexus Framework for scaling Scrum

---

**Nexus Extends Scrum**

Nexus is Scrum, with some small additions (see Table 2-1).

- It adds one additional Artifact: the Nexus Sprint Backlog. Nexus Sprint Backlog is the Nexus’ plan for the Sprint; it helps the Nexus understand what Scrum Teams are working on and makes any dependencies transparent that may exist between the teams during the Sprint.

---

1. George Miller’s oft-cited paper, “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information,” describes limitations in the way that we process information and form memories that reinforces ad hoc experience in organizing teams: when teams grow beyond about nine people they begin to lose cohesion and their work becomes harder to manage. For more information on Miller’s article, see https://en.wikipedia.org/wiki/The_Magical_Number_Seven,_Plus_or_Minus_Two.
Nexus Extends Scrum

- It adds five additional Events: Refinement, Nexus Sprint Planning, the Nexus Daily Scrum, the Nexus Sprint Review, and the Nexus Sprint Retrospective. These additional events extend Scrum to ensure that work is divided and coordinated across Scrum Teams in the most effective manner possible, and to share experiences across teams in the Nexus.

- It removes the individual Scrum Team Sprint Review, in favor of the Nexus Sprint Review. Since Scrum Teams in a Nexus work together to produce a single Integrated Increment, that Integrated Increment should be reviewed as a whole.

- It adds a new Role: the Nexus Integration Team. The Nexus Integration Team (NIT) exists to promote and provide transparent accountability for integration in a Nexus. It coaches and guides the application of Nexus with the Scrum Teams as well as within the organization. The NIT consists of the Product Owner of the product, a Scrum Master, and NIT members who are usually members of the Scrum Teams in the Nexus, but may come from other functional areas in the organization such as Operations, Security, Architecture, or other specialist areas that may help the Nexus deliver an Integrated Increment. These “outside” members may be temporary members who join the NIT for as long as is necessary.

Table 2-1 Nexus Roles, Events, and Artifacts

<table>
<thead>
<tr>
<th>Roles</th>
<th>Events</th>
<th>Artifacts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Teams</td>
<td>The Sprint</td>
<td>Product Backlog</td>
</tr>
<tr>
<td>Product Owner</td>
<td>Nexus Sprint Planning *</td>
<td>Nexus Sprint Backlog *</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>Sprint Planning</td>
<td>Sprint Backlog</td>
</tr>
<tr>
<td>Nexus Integration Team *</td>
<td>Nexus Daily Scrum *</td>
<td>Integrated Increment</td>
</tr>
<tr>
<td></td>
<td>Daily Scrum</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Nexus Sprint Review *</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Nexus Sprint Retrospective *</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sprint Retrospective</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Refinement *</td>
<td></td>
</tr>
</tbody>
</table>

* Nexus specific
The Nexus Integration Team

The NIT ensures that an Integrated Increment is produced at least every Sprint for the Nexus. The Scrum Teams do the work. Ultimately, the NIT is accountable for maximizing the value of the integrated Product (see Figure 2-2). Their activities may include developing tools and practices that will help with integration or serving as coaches and consultants to help with coordination.

NIT members need to have a teaching mind-set to help Scrum Teams resolve their issues whenever possible. Their role is to help to highlight issues that need to be solved and to help the Scrum Teams solve the issues. Only in emergencies does the NIT jump in and solve problems directly.

The NIT consists of:

- **The Product Owner**, the owner of the Product, and ultimately accountable for its success. In the context of the NIT, the Product Owner is accountable for ensuring that maximum value is delivered by the Nexus during each Sprint. The Product Owner’s role does not change from Scrum; the scope of the work is simply more complex.
- **A Scrum Master**, who has overall responsibility for ensuring the Nexus framework is enacted and understood. This Scrum Master is often a Scrum Master in one or more of the other Scrum Teams in the Nexus.
- **A Development Team**, whose members are usually members of Scrum Teams in the Nexus.

Contrary to what its name may suggest, the NIT doesn’t integrate the work of all Scrum Teams as it is delivered. Instead, it is accountable for ensuring that the teams are able to achieve integration themselves.

Members coach Scrum Teams and help remove dependencies. If something is preventing the Scrum Teams in the Nexus from producing an integrated Product, the NIT is accountable for making sure that those issues get resolved.
The Nexus Integration Team

- **Product Owner**
  Accountable for maximizing the value of the Product

- **Development Team**
  Accountable for creating Integrated Increments that are “Done”

- **Scrum Master**
  Accountable for the Scrum Teams doing Scrum and Nexus correctly and maximizing the value delivered by the Development Team

*Figure 2-2* The NIT is accountable for maximizing the value of the integrated Product

Members of the NIT may also work on Scrum Teams in the Nexus, but when they do they must put their work on the NIT first for the greater benefit the whole Nexus (see Figure 2-3).

*Figure 2-3* Members of the NIT are usually drawn from Scrum Teams
NIT members may come from outside the Scrum Teams; that is, from other parts of the organization. When they do, it is to provide unique expertise that the Scrum Teams lack, in areas such as Enterprise Architecture or Continuous Delivery, or in some area of specialized domain knowledge. NIT members may simply obtain their help without them actually becoming full-fledged members of the Nexus, but in some cases, when extensive support is needed, it may make sense for them to actually join the Nexus. When they do, they become accountable for Product delivery, just as are other members of the NIT.

The membership of the NIT can change over time as its needs evolve. Early in the life cycle of a Nexus, it may focus on coaching the Scrum Teams in the Nexus on scaling practices, or it may be more involved in stabilizing the shared build and test automation framework. Later in the life cycle, when the Nexus is running smoothly, the NIT may shift to raising awareness of issues from cross-team dependencies. Chapter 3, “Forming a Nexus,” describes the formation, composition, and evolution of the NIT in more detail.

**Nexus Events**

Nexus adds four events to Scrum, and replaces one Scrum event, to help Scrum Teams divide and coordinate work across teams in the most effective manner. The events defined by Nexus are

- **Refinement** is a formal event for the Nexus to collaborate on the details of the Product Backlog Items (PBIs) and see that they are adequately independent, so that the teams can select and work on without excessive conflict. In the process of working out the dependencies, teams also work out which backlog items they will likely work on. The Nexus continually refines the Product Backlog, as needed, and there is no specific time box for refinement.

---

2. Nexus events are guided by the time boxes for their related Scrum events, meaning that they generally take a similar amount of time. As a practical matter, a Nexus event takes as much time as the Nexus needs and is over when it’s over. If, after that, the Nexus thinks that it took too long, there is a good opportunity for inspection and adaptation to improve for the next time.
Nexus Events

- **Nexus Sprint Planning** helps the teams in the Nexus to collectively agree on the Nexus Goal and how each team will contribute to it.

- **The Nexus Daily Scrum** helps the Nexus to make integration issues transparent so that the Scrum Teams can know who is responsible for fixing them. It is a daily opportunity for the teams in the Nexus to sync with one another.

- **The Nexus Sprint Review** enables the Nexus to gather feedback on the Integrated Increment. It replaces individual Scrum Team Sprint Reviews.

- **The Nexus Sprint Retrospective** helps the teams share experiences and coordinate their resolution of shared challenges.

Refinement

In Scrum, Product Backlog refinement is not a mandatory event, but it is a strongly recommended practice. In Nexus, refinement is essential; it helps Scrum Teams work together to determine which team will deliver specific PBIs and to identify cross-dependencies across teams. Refinement is a cross-team event, with as many Scrum Team members present as is necessary to understand and decompose the PBIs.

Refinement results in a Product Backlog that is granular enough for Scrum Teams to pull work without creating unmanageable dependencies. During Refinement, the Scrum Teams should focus on these questions.

- What work will each team pull?
- In what order does that work need to be done to deliver the greatest business value earliest, while minimizing risk and complexity?

Nexus Sprint Planning

The Nexus takes the refined Product Backlog as input for the Nexus Sprint Planning event (see Figure 2-4). Nexus Sprint Planning helps to synchronize the activities of the Scrum Teams in a Nexus for a single Sprint.
Nexus Sprint Planning consists of:

- **Validating the Product Backlog.** The Scrum Teams review the PBIs and make any necessary adjustments needed to the work from the Refinement event. All of the Scrum Teams should participate and contribute to minimize communication issues; however, only the appropriate representatives (those who feel that they can make a contribution to refining the PBIs) from each of the Scrum Teams need to attend.

- **Formulating the Nexus Goal.** The Nexus Goal is a Sprint objective that is met through the implementation of PBIs by multiple teams.

- **Scrum Team Sprint Planning.** Once the Nexus Goal for the Sprint is understood, each Scrum Team will conduct its individual Sprint Planning events in which the members create their own Sprint Backlogs. As they identify dependencies with other teams, they work with those teams to minimize or eliminate the dependencies.

In some cases, this will mean that the sequence of work across teams may have to be adjusted to let one team finish its work before another starts.
This could be accomplished by breaking apart dependent work so that some parts can be worked independently, or by one team choosing non-dependent PBIs to work on, to avoid waste resulting from unresolved cross-team dependencies. Teams may also work together to shift work from one team to another to better balance the work. The NIT will help to make sure that dependencies are communicated and visualized on the Nexus Sprint Backlog.

Nexus Sprint Planning is complete when each Scrum Team in the Nexus has finished its individual Sprint Planning events.

**The Nexus Daily Scrum**

The Nexus Daily Scrum brings together the 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. Topics typically discussed include the following.

- Was the previous day’s work successfully integrated, and if not, why?
- Have any new dependencies been identified?
- What information needs to be shared across teams in the Nexus?

During the Nexus Daily Scrum and throughout the day, the Nexus Sprint Backlog may be updated by the Scrum Teams to visualize and manage current inter-team dependencies. It is not simply an aggregation of the individual teams’ Sprint Backlogs, since each team will have work for itself as well as the Product Backlog work that it needs to do. Work that is identified during the Nexus Daily Scrum is then taken back to individual Scrum Teams for planning inside their Daily Scrum events.

**The Nexus Sprint Review**

The Nexus Sprint Review replaces individual Scrum Team Sprint Reviews and is held at the end of the Sprint. Its purpose is to capture feedback from stakeholders on the entire Integrated Increment of the Nexus. The Nexus Sprint Review replaces the individual Scrum Team Sprint Reviews because individual Scrum Teams might not produce a meaningful Integrated Increment on their own when Nexus is used.
There are several benefits to having a single Sprint Review for the Nexus, including the following.

- Teams are logically each other’s stakeholders, so they can provide one another with feedback that helps the Nexus improve.

- If individual Scrum Team Sprint Reviews were held, stakeholders may not be able to attend all of them, and even if they did they would not see the integrated Product.

- Some issues only become evident when the integrated Product is reviewed as a whole, especially when each team is developing one or more components. Each component may work in isolation, but they may not work together to produce an integrated Product.

- Reviewing the Integrated Increment as a whole brings all the teams in the Nexus together and reminds them that their goal is a single integrated solution.

Even when some teams may actually deliver logically separated subproducts that may be independently reviewed, shipped, and used, there is value in reviewing them in the context of the Nexus’ integrated Product Increment.

All members of the Nexus participate in the Nexus Sprint Review.

**The Nexus Sprint Retrospective**

The Nexus Sprint Retrospective provides the means by which the Nexus enables inspection and adaptation. To conduct the Nexus Retrospective:

1. Representatives from across the Nexus meet and identify issues that have impeded more than a single team to make shared issues transparent to all Scrum Teams. The representatives consist of the NIT members, as well as anyone with an interest in sharing their perspectives on inter-team issues.

2. Each Scrum Team holds its own Sprint Retrospective, just as they would do in Scrum, but the team also considers issues raised from the first part of the Nexus Retrospective as input to its team discussions while the members determine actions to address these issues.
3. Representatives from the Scrum Teams meet once again to discuss common issues identified in the Scrum Team Retrospectives. They agree on how to visualize and track the identified actions that will enable the Nexus to learn and adapt as a whole.

**Questions to Ask in Every Nexus Sprint Retrospective**

Nearly every Nexus encounters common scaling challenges. Questions that help teams to identify challenges include the following.

- 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?

When challenges are identified, ask the following:

- Why did this happen?
- How can technical debt be undone?
- How can the recurrence be prevented?

Nexus Events are described in more detail in Chapter 5, “Nexus in Action.”

**Nexus Artifacts**

Artifacts capture the results of work performed. They also provide transparency and opportunities for inspection and adaptation.

**Product Backlog**

There is a single Product Backlog for the entire Nexus and all of its Scrum Teams. Since a Nexus is organized a single product, it only has a single Product Owner, and that single Product Owner maintains a single Product Backlog. All teams pull work from this single artifact.
**Nexus Goal**

During the Nexus Sprint Planning meeting, the Product Owner discusses a goal for the Sprint. This is called the Nexus Goal. It is the sum of all the work and Sprint Goals of the individual Scrum Teams within the Nexus. The Nexus should demonstrate the functionality that it developed to achieve the Nexus Goal at the Nexus Sprint Review.

**Nexus Sprint Backlog**

The Nexus Sprint Backlog contains the PBIs that have cross-team dependencies or potential integration issues. It does not contain PBIs that have no dependencies, nor does it contain tasks from the individual Scrum Team Sprint Backlogs. 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 is the integrated aggregation of all work completed by all the Scrum Teams in a Nexus. The Integrated Increment must be usable and potentially releasable, which means it must meet the definition of “Done” agreed to by the Development Team. The Product Owner is a key stakeholder for it and defines the quality criteria that the Product Increment must meet. The Integrated Increment is inspected at the Nexus Sprint Review.

**Artifact Transparency**

Just like Scrum, on which it builds, Nexus is based on transparency. The NIT works with the Scrum Teams in the Nexus, and the broader organization, to ensure that all Scrum and Nexus artifacts are visible and that the state of the Integrated Increment can be easily 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, making it difficult or impossible to guide the Nexus effectively to minimize risk and maximize value.
What Do You Need to Get Started with Nexus?

The greatest challenge a Nexus faces is detecting and resolving dependencies before technical debt accumulates to an unacceptable level. The test of unacceptable technical debt is when the Nexus tries to integrate the work from its Scrum Teams. When that integration fails, the unresolved dependencies remain hidden in the code and test base, lowering or negating the value of the software.

**Definition of “Done” in Nexus**

The NIT 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 it has been determined to be usable and potentially releasable by the Product Owner. A PBI 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 they cannot apply less rigorous criteria than agreed for the Increment.

What Do You Need to Get Started with Nexus?

Nexus is based on the Scrum Framework and adds minimal events, roles, and artifacts to increase transparency, communication, and collaboration among teams. The new Nexus events, roles, and artifacts help ensure a successfully Integrated Increment is developed. Just like with Scrum, you don’t need much to get started with Nexus. And just like with Scrum, Nexus is simple to learn but hard to master. Here are the minimal but required prerequisites you will need to implement Nexus.
You should have

- Scrum experience.
- A single Product Backlog, and a single Product Owner, for a single Product.
- Identified teams that will be in a Nexus. They should have an overview of the Nexus Framework.
- Identified members who will make up the NIT for the Nexus.
- A definition of “Done.”
- Identified Sprint cadence.

Closing

Nexus is simple to understand, but mastering it takes practice and feedback. Like Scrum, its basic concepts are simple. Like Scrum, it is also not prescriptive; it says that you need to engage the Nexus in planning the Sprint, but it does not tell you how to do that because there are many techniques that you might find useful to help you to plan. In the following chapters, we will explore applying Nexus in a case study. In doing so, we will often illustrate how Nexus works by using specific complementary practices. These are not specifically part of Nexus, but they will help you understand better how Nexus works. With that, let’s get started with what it takes to form a Nexus in Chapter 3.
A
acceptance test-driven development, 94
accountability
  NIT (Nexus Integration Team), 34, 111
  Product Owners, 123
  versus responsibility, 34
actors, 37
adaptation, 3–4
advantages
  of Agile, 2
  of Nexus Framework, 5, 7–8
Adzic, Gojko, 35
Agile, advantages of, 2
ambassadors, 29
Andon cord, 115–116
Andreesen, Marc, 2
APIs (application program interfaces), 25
  automated API-based testing, 26
  versioned API management, 26
architectural dependencies, 43
Arduino, 104
arrows on refinement boards, 46
artifacts, 17
  Integrated Increment, 18
  Nexus Goals
    definition of, 18
    establishing, 49
    formulating, 14
  Nexus Sprint Backlog, 8
    building, 52–54
    definition of, 18
  Product Backlog
    consolidating, 35–37
    definition of, 17
  PBIs (Product Backlog Items), 36–37, 49–51
  Product Backlog Burndown charts, 63–64
  Product Backlog Treemaps, 62–63
  refining, 38–47, 83–85, 98–102
upating, 83–85
validating, 14, 35–37
transparency, 18–19
authority of NIT (Nexus Integration Team), 123
automation
automated API-based testing, 26
build and test automation, 92–93
availability of Product Owners, 91–92
B
backlogs
enterprise improvement backlog, creating, 132–133
Nexus Sprint Backlog, 8
building, 52–54
definition of, 18
explained, 18
Product Backlog
consolidating, 35–37
definition of, 17
PBIs (Product Backlog Items), 36–37, 49–52
Product Backlog Burndown charts, 63–64
Product Backlog Treemaps, 62–63
refining, 38–47, 83–85, 98–102
updating, 83–85
validating, 14, 35–37
Scrum Team Backlogs, building, 52–54
benefits
of Agile, 2
of Nexus Framework, 5, 7–8
of Scrum Framework, 2–3
Brooks, Fred, 4, 94, 95, 127
build and test automation, 92–93
Burndown charts (Product Backlog), 63–64
business value, forming teams around, 26–27
C
challenges of scaling Scrum, 94–96
charts, Product Backlog Burndown, 63–64
CI/CD (continuous integration/continuous delivery) consultant, 83
code base, opening, 25–26
code review, 26
commitment, 132
component teams, problems with, 76–77
connecting PBIs (Product Backlog Items) to value delivery, 51–52
consolidating Product Backlog, 35–37
continuous integration, 26
continuous integration/continuous delivery (CI/CD) consultants, 83
Conway’s Law, 76
courage, 132
crisis resolution, 97–98
Nexus Daily Scrum, 108–109
distributed teams and, 109
overcoming challenges with, 110–111
Nexus Sprint Planning, 102
large-scale distributed sprint planning, 102–103
mixed hardware/software development, 103–105
mixing Scrum/Waterfall approaches in, 107–108
teams working at different sprint cadences, 105–107
Nexus Sprint Retrospective, 118–119
Nexus Sprint Review, 118–119
NIT (Nexus Integration Team), 111–112
descaling, 112–113
Health Checks, 114–115
Scrumbling, 115–118
Product Backlog, refining, 98–102
“critical path” of work, identifying, 45
cross-functional teams, 24–25
cross-team Product Backlog refinement, 39–41
Cross-Team Refinement Boards, 45–47

D
Daily Scrum. See Nexus Daily Scrum
deliverables
definition of, 37
Impact Maps, 35–37
dependencies
Product Backlog item dependencies, 42–43
Story Mapping, 43–45
descaling, 112–113
Development Teams, 2–3, 77
distributed teams
challenges of, 98
large-scale distributed sprint planning, 102–103
Nexus Daily Scrum, 109
Product Backlog, refining, 98–102
diverge and merge techniques, 102–103
domain expertise, 42
“done,” definition of, 19, 93

E
emergency response, 97–98
Nexus Daily Scrum, 108–109
distributed teams and, 109
overcoming challenges with, 110–111
Nexus Sprint Planning, 102
large-scale distributed sprint planning, 102–103
mixed hardware/software development, 103–105
mixing Scrum/Waterfall approaches in, 107–108
teams working at different sprint cadences, 105–107
Nexus Sprint Retrospective, 118–119
Nexus Sprint Review, 118–119
NIT (Nexus Integration Team), 111–112
descaling, 112–113
Health Checks, 114–115
Scrumbling, 115–118
Product Backlog, refining, 98–102
enterprise improvement backlog, creating, 132–133
establishing Nexus Goals, 49
estimating PBIs (Product Backlog Items), 49–51
events, 9
Nexus Daily Scrum, 9, 108–109
benefits of, 122
definition of, 13
distributed teams and, 109
evolving, 86–87
explained, 15, 58–61
overcoming challenges with, 110–111
Nexus Sprint Planning, 9, 47–48, 102
definition of, 13
explained, 13–15
large-scale distributed sprint planning, 102–103
length of, 54
mixed hardware/software development, 103–105
mixing Scrum/Waterfall approaches in, 107–108
Nexus Goals, establishing, 49
Nexus Sprint Backlog, 52–54
PBIs (Product Backlog Items), 49–52
Scrum Team Backlogs, 52–54
teams working at different sprint cadences, 105–107
Nexus Sprint Retrospective, 121
definition of, 13
emergency response, 118–119
evolving, 89–90
explained, 16–17
Nexus Daily Scrum mentions in, 122
NIT (Nexus Integration Team) in, 123
parts of, 68–73
purpose of, 68
questions to ask, 17
Nexus Sprint Review definition of, 13
emergency response, 118–119
evolving, 87–88
explained, 15–16, 66–67
“exposition” format for, 67
offline review techniques for, 67–68
Refinement, 9
definition of, 12
Product Backlog, 38–47, 83–85, 98–102
questions to ask, 13
evolving teams, 75
acceptance test-driven development, 94
build and test automation, 92–93
challenges of scaling Scrum, 94–96
component teams, problems with, 76–77
Development Teams, 77
Nexus Daily Scrum, 86–87
Nexus Sprint Planning, 85–86
Nexus Sprint Retrospective, 89–90
Nexus Sprint Review, 88
NIT (Nexus Integration Team) expansion, 83
organization around features, 77–81
organization around personas, 81–82
plans for improvement, forming, 93–94
Product Backlog, updating and refining, 83–85
technical debt, dealing with, 91
unavailable Product Owners, 91–92
work versus progress, 90–91
expanding NIT (Nexus Integration Team), 83
“exposition” format for Nexus Sprint Reviews, 67
external dependencies, 43, 46

F
FBI’s Sentinel Project, 113
“feature team” model, 71
features, organizing teams around, 71, 77–81
focus, 132
formation of teams, 21–23
  around increments of business
  value, 26–27
code base, opening, 25–26
cross-functional teams, 24–25
independence of teams, 22–23
NIT (Nexus Integration Team),
  32–34
self-organizing teams, 28
  stages of, 89–90
forming stage of team formation, 90
formulating Nexus Goals, 14
Fowler, Martin, 91, 107
frequency of releases, 124

**G**
goals. See Nexus Goals
Gothelf, Jeff, 133
growing teams, 29
  pairing and “internships” 30–31
  starting small, 29–30

**H**
hardware dependencies, 46
Health Checks, 114–115

**I**
Impact Maps, 35–37
impacts
  definition of, 37
  Impact Maps, 35–37
improvement, areas for
  Product Owners, scaling,
    128–129
  Scrum values, 131–132
  skill development, 129
  technical debt management,
    127–128
  transparency and trust, 130

increments of business value,
  forming teams around, 26–27
independence of teams, 22–23
inspection, 3–4
Integrated Increment, 18
Integration Team. See NIT (Nexus
  Integration Team)
“internship” model, 30–31
item dependencies
  Product Backlog, 42–43
  Story Mapping, 43–45

**J-K-L**
large-scale distributed sprint
  planning, 102–103
Lean Manufacturing Andon cord,
  115–116

**M**
“The Magical Number Seven, Plus
  or Minus Two” (Miller), 8
mapping
  Impact Maps, 35–37
  Product Backlog Treemaps, 62–63
  Story Mapping, 43–45
measures, connecting PBIs (Product
  Backlog Items) to, 51
meetings. See events
Miller, George, 8
mixed hardware/software
  development, 103–105
Moore, Geoffrey, 1
morale (team), 125
The Mythical Man-Month (Brooks),
  4, 94, 95, 127

**N**
Net Promoter Scores, 37
“The New New Product
  Development Game” 125, 127
Nexus advantages, 5, 7–8
Nexus case studies, 134
Nexus Daily Scrum, 9, 108–109
   benefits of, 122
   definition of, 13
distributed teams and, 109
evolving, 86–87
explained, 15, 58–61
overcoming challenges with,
   110–111
Nexus Goals
   definition of, 18
   establishing, 49
   formulating, 14
   in Impact Map, 37
Nexus Integration Team. See NIT
   (Nexus Integration Team)
Nexus online guide, 134
Nexus Sprint Backlog, 8
   building, 52–54
   definition of, 18
Nexus Sprint Planning, 9, 47–48,
   102. See also Product Backlog
   definition of, 13
evolving, 85–86
explained, 13–15
large-scale distributed sprint
   planning, 102–103
length of, 54
mixed hardware/software
   development, 103–105
mixing Scrum/Waterfall
   approaches in, 107–108
Nexus Goals, establishing, 49
Nexus Sprint Backlog, 8
   building, 52–54
   definition of, 18
PBIs (Product Backlog Items)
   connecting to value delivery,
      51–52
   estimating and sizing, 49
Scrum Team Backlogs, 52–54
   teams working at different sprint
   cadences, 105–107
Nexus Sprint Retrospective, 9, 121
   definition of, 13
emergency response, 118–119
enterprise improvement backlog,
   creating, 132–133
evolving, 89–90
explained, 16–17
Nexus Daily Scrum mentions in, 122
NIT (Nexus Integration Team)
   and, 123
parts of, 68–73
Product Owner, scaling, 128–129
productivity, 125
purpose of, 68
questions to ask, 17
release frequency, 124
Scrum values, 131–132
self-organization, 125–126
skill development, 129
technical debt management,
   127–128
transparency and trust, 130–131
Nexus Sprint Review, 9
   definition of, 13
emergency response, 118–119
   evolving, 87–88
explained, 15–16, 66–67
   “exposition” format for, 67
   offline review techniques for,
      67–68
NIT (Nexus Integration Team)
   accountability, 34, 111
   authority, 123
   benefits of, 123
descaling, 112–113
   emergency mode, 111–112
   expanding, 83
formation of, 32–34
Health Checks, 114–115
members of, 10–12, 34
purpose of, 9
responsibility, 34
Scrumbling, 115–118
“No Silver Bullet—Essence and Accident in Software Engineering” (Brooks), 127
norming stage of team formation, 90
number of teams, 31

O
OEM (Original Equipment Manufacturer) agreement, 98
offline review techniques for Nexus Sprint Reviews, 67–68
“open source” coding approach, 78–81
Open Space principles, 39, 103
opening code base, 25–26
openness, 132
organizational authority, 43
organizing teams
around features, 77–81
around personas, 81–82
“open source” coding approach, 78–81
Original Equipment Manufacturer (OEM) agreement, 98
outcomes, 26–27, 51
Overeem, Barry, 40

P
pairing, 30–31
Patton, Jeff, 43, 44
PBIs (Product Backlog Items)
connecting to value delivery, 51–52
estimating and sizing, 49–51
validating, 36–37
people dependencies, 42
performing stage of team formation, 90
personas, 26–27, 81–82
Plan-Do-Check-Adapt Sprint Retrospective Board, 72
planning. See also Product Backlog
Nexus Daily Scrum, 9, 108–109
benefits of, 122
definition of, 13
distributed teams and, 109
evolving, 86–87
explained, 15, 58–61
overcoming challenges with, 110–111
Nexus Sprint Planning, 9, 47–48, 102
definition of, 13
evolving, 85–86
explained, 13–15
large-scale distributed sprint planning, 102–103
length of, 54
mixed hardware/software development, 103–105
mixing Scrum/Waterfall approaches in, 107–108
Nexus Goals, establishing, 49
Nexus Sprint Backlog, 52–54
PBIs (Product Backlog Items), 49–52
Scrum Team Backlogs, 52–54
teams working at different sprint cadences, 105–107
planning visibility, 65
plans for improvement, forming, 93–94
prerequisites for Nexus, 19–20
Product Backlog
consolidating, 35–37
definition of, 17
PBIs (Product Backlog Items)
connecting to value delivery, 49–51
estimating and sizing, 49–51
validating, 36–37
Product Backlog Burndown charts, 63–64
Product Backlog Treemaps, 62–63
refining, 38–39, 83–85
cross-team Product Backlog refinement, 39–41
Cross-Team Refinement Boards, 45–47
emergency response, 98–102
item dependencies, 42–43
Story Mapping, 43–45
questions to ask, 13
release frequency, 124
requirements for Nexus, 19–20
respect, 132
responding to emergencies. See emergency response
responsibility versus accountability, 34
NIT (Nexus Integration Team), 34
Retrospective. See Nexus Sprint Retrospective reviews
code review, 26
Nexus Sprint Review, 9
definition of, 13
explained, 15–16, 66–67
“exposition” format for, 67
offline review techniques for, 67–68
Scrum Team Sprint Reviews, 66
roles (NIT)
members of, 10–12
purpose of, 9
Rook, Michiel, 107
running sprints, 57–58. See also sprints
Nexus Daily Scrum, 58–61
Nexus Sprint Retrospective parts of, 68–73
purpose of, 68
Nexus Sprint Review explained, 66–67
“exposition” format for, 67
offline review techniques for, 67–68

Q-R
Refinement, 9
definition of, 12
Product Backlog, 38–39, 83–85
cross-team Product Backlog refinement, 39–41
Cross-Team Refinement Boards, 45–47
emergency response, 98–102
item dependencies, 42–43
Story Mapping, 43–45
questions to ask, 13
release frequency, 124
requirements for Nexus, 19–20
respect, 132
responding to emergencies. See emergency response
responsibility versus accountability, 34
NIT (Nexus Integration Team), 34
Retrospective. See Nexus Sprint Retrospective reviews
code review, 26
Nexus Sprint Review, 9
definition of, 13
explained, 15–16, 66–67
“exposition” format for, 67
offline review techniques for, 67–68
Scrum Team Sprint Reviews, 66
roles (NIT)
members of, 10–12
purpose of, 9
Rook, Michiel, 107
running sprints, 57–58. See also sprints
Nexus Daily Scrum, 58–61
Nexus Sprint Retrospective parts of, 68–73
purpose of, 68
Nexus Sprint Review explained, 66–67
“exposition” format for, 67
offline review techniques for, 67–68
transparency, 61–62
planning visibility, 65
Product Backlog Burndown charts, 63–64
Product Backlog Treemaps, 62–63

S
scaling
Product Owners, 128–129
simplicity and, 6
Schwaber, Ken, 28
Scrum Framework
advantages of, 2–3
Development Teams, 77
elements of, 3–4
mixing with Waterfall approach, 107–108
Nexus as extension of, 8–9
Scrum Guide, 4
“Scrum of Scrums” approach, 29
values, 28, 131–132
Scrum Guide, 4
Scrum Masters, 2–3, 9, 10, 29, 59
“Scrum of Scrums” approach, 29
Scrum Team Backlogs, 52–54
Scrum Teams
Scrum Team Backlogs, building, 52–54
Scrum Team Sprint Reviews, 66
size of, 80
Scrumbling, 115–118
Scrum.org community blog, 134
Secure property, 36
Seiden, Josh, 133
distributed teams, 28, 77, 125–126
Sense and Respond (Gothelf and Seiden), 133
simplicity, as key to scaling, 6
sizing
PBIs (Product Backlog Items), 49–51
Scrum Teams, 80
skill development, 129
software dependencies, 46
Spotify Health Check model, 114
Sprint Backlog. See Nexus Sprint Backlog
sprint cadences, differences in, 105–107
Sprint Planning. See Nexus Sprint Planning
Sprint Retrospective. See Nexus Sprint Retrospective
Sprint Review. See Nexus Sprint Review
sprints
Nexus Daily Scrum, 108–109
distributed teams and, 109
explained, 58–61
overcoming challenges with, 110–111
Nexus Sprint Backlog, 8
building, 52–54
definition of, 18
Nexus Sprint Planning, 9, 47–48, 102
definition of, 13
evolving, 85–86
explained, 13–15
large-scale distributed sprint planning, 102–103
length of, 54
mixed hardware/software development, 103–105
mixing Scrum/Waterfall approaches in, 107–108
Nexus Goals, establishing, 49
Nexus Sprint Backlog, 52–54
PBIs (Product Backlog Items), 49–52
Scrum Team Backlogs, 52–54
teams working at different sprint cadences, 105–107
Nexus Sprint Retrospective, 9, 118–119, 121
definition of, 13
emergency response, 118–119
enterprise improvement backlog, creating, 132–133
evolving, 89–90
explained, 16–17
Nexus Daily Scrum mentions in, 122
NIT (Nexus Integration Team), 123
NIT (Nexus Integration Team) and, 123
parts of, 68–73
Product Owner, scaling, 128–129
productivity, 125
purpose of, 68
questions to ask, 17
release frequency, 124
Scrum values, 131–132
self-organization, 125–126
skill development, 129
technical debt management, 127–128
transparency and trust, 130–131
Nexus Sprint Review, 9, 118–119
definition of, 13
emergency response, 118–119
evolving, 87–88
explained, 15–16, 66–67
“exposition” format for, 67
offline review techniques for, 67–68
running, 57–58
Nexus Daily Scrum, 58–61
Nexus Sprint Retrospective, 68–73
Nexus Sprint Review, 66–68
transparency, 61–65
starting small, 29–30
storming stage of team formation, 90
Story Mapping, 43–45
story slicing, 40
Strangler Application Pattern, 107
Sutherland, Jeff, 28

t

teams
component teams, problems with, 76–77
Development Teams, 77
distributed teams
challenges of, 98
large-scale distributed sprint planning, 102–103
Nexus Daily Scrum, 109
Product Backlog, refining, 98–102
evolving, 75
acceptance test-driven development, 94
build and test automation, 92–93
challenges of scaling Scrum, 94–96
component teams, problems with, 76–77
Development Teams, 77
Nexus Daily Scrum, 86–87
Nexus Sprint Planning, 85–86
Nexus Sprint Retrospective, 89–90
Nexus Sprint Review, 88
NIT (Nexus Integration Team), expanding, 83
organization around features, 77–81
organization around personas, 81–82
plans for improvement, forming, 93–94
Product Backlog, updating and refining, 83–85
technical debt, dealing with, 91
unavailable Product Owners, 91–92
work versus progress, 90–91
“feature team” model, 71
formation of, 21–23
code base, opening, 25–26
cross-functional teams, 24–25
forming around increments of business value, 26–27
independence of teams, 22–23
NIT (Nexus Integration Team), 32–34
self-organizing teams, 28
stages of, 89–90
growing, 29
pairing and “internships” 30–31
starting small, 29–30
morale, 125
NIT (Nexus Integration Team) accountability, 34, 111
authority, 123
benefits of, 123
descaling, 112–113
emergency mode, 111–112
expanding, 83
formation of, 32–34
Health Checks, 114–115
members of, 10–12, 34
purpose of, 9–12
responsibility, 34
Scrumbling, 115–118
number of, 31
organizing
around features, 77–81
around personas, 81–82
“open source” coding approach, 78–81
Scrum Teams
Scrum Team Backlogs, building, 52–54
size of, 80
self-organizing teams, 28, 77, 125–126
teams working at different sprint cadences, 105–107
technical debt management, 91, 127–128
technical expertise, 42–43
testing
acceptance test-driven development, 94
automated API-based testing, 26
build and test automation, 92–93
“thinly sliced” backlog items, 40
transparency, 3–4, 61–62, 91, 130–131
artifacts, 18–19
planning visibility, 65
Product Backlog Burndown charts, 63–64
Product Backlog Treemaps, 62–63
Treemaps (Product Backlog), 62–63
trunk-based development, 26
trust, 130–131
Tuckman, Bruce, 86
U
Ulwick, Tony, 27
unavailable Product Owners, 91–92
updating Product Backlog, 83–85

V
validating Product Backlog, 14, 35–37
Value Areas, 26–27
value delivery, connecting PBIs (Product Backlog Items) to,
51–52
values (Scrum), 28, 131–132

VCF (Virtual Case File) project, 113
velocity, visualizing, 63–64
versioned API management, 26
Virtual Case File (VCF) project, 113
visibility, planning, 65

W–X–Y–Z
Waterfall approach, mixing with Scrum, 107–108
What Customers Want (Ulwick), 27
work versus progress, 90–91
World Café method, 103
Zuill, Woody, 49