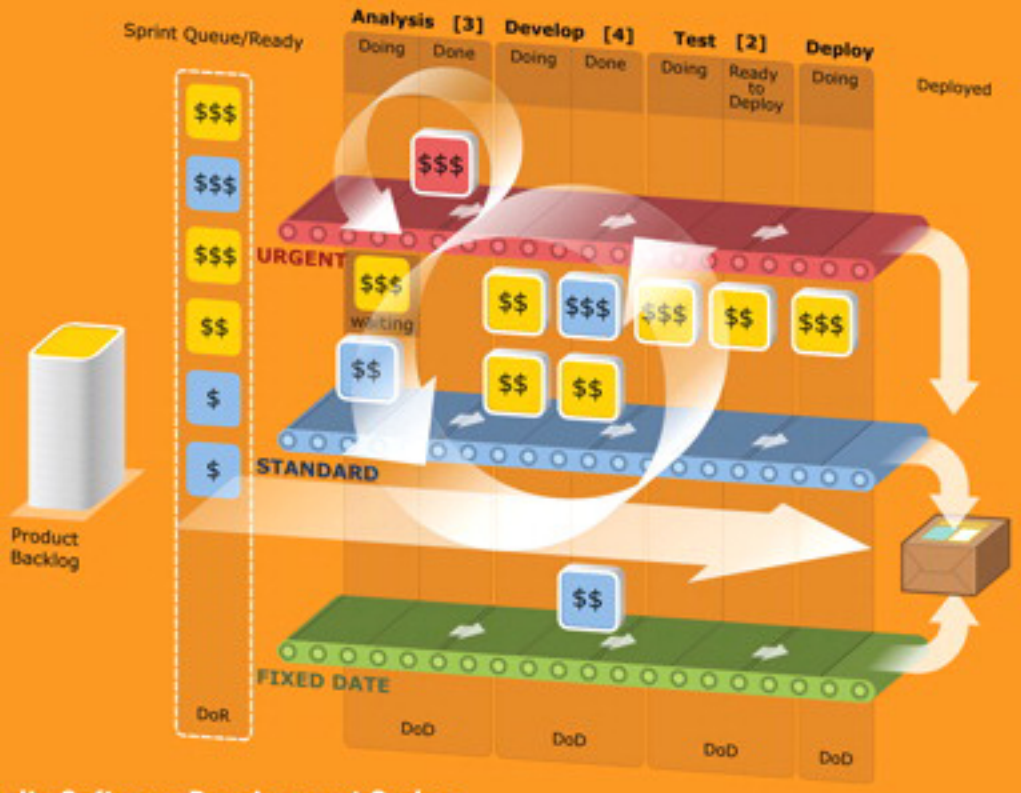
The Scrumban [R]Evolution

Getting the Most Out of Agile, Scrum, and Lean Kanban

Ajay Reddy

Forewords by ***David Anderson*** *and* ***Jim Benson***



Agile Software Development Series

Alistair Cockburn and Jim Highsmith, Series Editors

CONTENTS

Chapter 1

Manifestations: Scrumban Demystified

I estimate that 75% of those organizations using Scrum will not succeed in getting the benefits that they hope for from it . . . Scrum is a very simple framework within which the "game" of complex product development is played. Scrum exposes every inadequacy or dysfunction within an organization's product and system development practices. The intention of Scrum is to make them transparent so the organization can fix them. Unfortunately, many organizations change Scrum to accommodate the inadequacies or dysfunctions instead of solving them.

—Ken Schwaber, co-creator of Scrum

Scrum is an incredibly simple, effective, and popular software development framework; its value increases as teams and organizations develop their understanding and application of its core principles and practices.

Despite its simplicity, Scrum can be difficult to master. While it empowers both individuals and teams to discover factors that impede Agility and address how they should be tackled, it relies on their collective motivation, effort, and capabilities to make this happen. An entire industry now exists around delivering services to help individuals, teams, and organizations rise to higher levels of capability.

Scrumban is also a simple framework. It's a relative newcomer to the world of software development and has yet to fully evolve; it's also often misunderstood by people in Lean and Agile circles. Many people have never even heard of it. Some believe it to be nothing more than using virtual kanban systems within the Scrum framework, while others believe it to be a new software development framework that combines "the best" elements of Scrum and the Kanban Method. Neither of these viewpoints captures the true essence of Scrumban.

A Helpful Perspective

In the early 2000s, Alistair Cockburn introduced "Shu-Ha-Ri" to the software development world. It provided a way of thinking about the three distinct phases people pass through as they master a new skill or concept. Cockburn borrowed this concept

from Japanese martial arts and believed it could be effectively applied within the context of improving software development processes and practices.

I invite you to embrace a learning style that is loosely based on Shu-Ha-Ri in learning to understand Scrumban. Let's quickly review the three stages:

- ***Shu (Beginner)*:** The first stage of learning. New learners seek to reproduce a given result by following a set of instructions that are practice focused. They focus on how to perform a task with a basic understanding of the underlying principles. Success in this stage is measured by whether a procedure works and how well the student understands why it works.
- ***Ha (Intermediate)*:** Once a student has mastered basic practices, values, and principles, he or she begins to identify the limits of these practices and techniques. The student seeks to expand his or her awareness of alternative approaches, learning when an alternative applies and when it breaks down. Success in this stage is measured by how well the student learns to apply adaptive capabilities to varying circumstances.
- ***Ri (Advanced)*:** The student has become a master. It no longer matters whether he or she is following a given procedure or practice. His or her knowledge and understanding are the product of repeated thoughts and actions. The master has developed a fully adaptive capability within the context of his or her experience in the environment. Success is measured by consistently successful outcomes.

Informed readers should not confuse the concept of Shu-Ha-Ri with something like Carnegie Mellon University's Capability Maturity Model Integration (CMMI) process improvement training and appraisal program. Those who adopt and practice Scrumban principles would not be as inclined to direct harsh criticisms at the model as have some individuals in Lean and Agile circles.

Although I disagree that CMMI's prescribed "destinations" or "characteristics" correlate with capability in all contexts, this model does present a menu of potential catalysts to employ in a Scrumban framework when pursuing desired business outcomes. Viewed from the opposite direction, the flow management capabilities that Scrumban enables may provide a useful catalyst for organizations pursuing prescribed levels of CMMI capability to achieve their desired outcomes.

A Framework for [R]Evolution

When Corey Ladas introduced the world to Scrumban in his seminal book, *Essays on Kanban Systems for Lean Software Development* (Modus Cooperandi Press, 2009), he defined Scrumban as a transition method for moving software development teams from Scrum to a more evolved development framework. Over the past five years, my

own research and work experience have unearthed the many ways in which Scrumban itself has evolved.

Since Corey wrote his book, organizations have layered the Kanban Method alongside Scrum to help them achieve several different kinds of outcomes. For example, I've successfully helped organizations apply Scrumban principles and practices in a variety of contexts—from startups to Fortune 50 companies, in industries ranging from professional services to software product development. Across these contexts, I've used Scrumban for the following purposes:

- Help teams and organizations accelerate their transitions to Scrum from other development methodologies
- Enable new capabilities within teams and organizations to help them overcome challenges that Scrum (purposefully) causes them to confront
- Help organizations evolve new Scrum-like processes and practices that work for them—not to accommodate the inadequacies and dysfunctions that Scrum exposed, but rather to resolve them in a manner that is most effective for their unique environment

These experiences demonstrate that Scrumban has evolved to become a family of principles and practices that create complementary tools and capabilities. And like any living organism, these principles and practices will continue to evolve as practitioners share their experiences and learnings.

Now, let's consider the three different outcomes summarized previously within the context of Shu-Ha-Ri:

- Scrumban provides the discipline and structure needed by practitioners in the Shu phase of learning. The Scrumban framework enables teams and organizations to manage the introduction of the artifacts and ceremonies of Scrum or the enhanced metrics and flow management practices of Kanban—disciplines and structures that new learners require in limited scope.

  For example, Scrum's ceremonies are essential to creating desired levels of performance and agility. Although it is a relatively simple framework, Scrum can seem overwhelming when it is first introduced. In a misguided effort to ease adoption, many organizations modify or omit ceremonies or, even worse, ignore the importance of understanding the basic principles and values. This rarely, if ever, produces the desired outcomes.

  Additional capabilities provided by the Scrumban framework can substitute for the functions served by the modified or omitted Scrum ceremonies. Scrumban's visualizations and other mechanics improve the effectiveness while reducing the time and effort associated with conducting ceremonies. Scrumban more effectively connects the practice of ceremonies with the principles and values that the ceremonies serve.

- Scrumban exposes new tools and capabilities that aid the experiments and discoveries pursued in the Ha phase. Meeting the challenges that teams and organizations commonly face as they implement Scrum or other Agile practices represents one aspect of this dimension.

  Consider creating and managing a product backlog. This Scrum artifact, and the events surrounding it (grooming and planning sessions), is intended to manage risk and optimize value by enabling better decision making due to maximized transparency and understanding of work. This can be especially frustrating when organizations effectively assign multiple product owners to a backlog because individual limitations interfere with realizing a full set of capabilities, or because of wide variations in subjective assessments.

  The Scrumban framework visualizes information and integrates capabilities other frameworks don't inherently provide. It helps provide improved contextual understandings and more accurately measures the outcome of different approaches (directly appealing to the Ha phase practice and understanding). For instance, Scrumban visualizes all sources of work demands and a more objective economic impact over time (cost of delay) to help prioritize work, lending greater transparency to the overall picture and expanding ways to adapt.
- Scrumban is flexible enough to provide Ri-level masters with a robust process within which to operate at hyper-performing levels. By emphasizing systems thinking, experimentation, and discovery, Ri-level masters are free to mold their ways of working in whatever fashion will produce the best results—from both performance and worker satisfaction standpoints. It makes no difference whether the resulting process is true to any particular set of practices.
- Scrumban supports both "revolution" and "evolution." More importantly, it's structured in a way that strongly supports all levels of learning and understanding—at a level of quality that is greater than that provided by either Scrum or Kanban alone.

All of Scrumban's added capabilities can be *optionally* applied to a Scrum context in a variety of ways. They can also be extended across multiple areas of an organization to drive better business outcomes. Scrum's software development framework lies at its foundation, as does the Kanban Method. However, neither of these frameworks represents a prescribed destination for organizations practicing Scrumban.

Beyond representing a significantly evolved mindset from the framework expressed by Ladas, today's Scrumban is quite different from the definitions used by other respected leaders in the Lean/Agile community.[1] In many respects, these perspectives view Scrumban as a vehicle for moving teams from Scrum to another software development process. While this remains *one* potential outcome, real-world

1. See, for example, http://tiny.cc/badcomparisonsSK (July 2013).

applications demonstrate Scrumban has come to entail more than this across a broad range of contexts.

Over the years, Scrumban has been used to help teams and organizations accelerate their transitions to Scrum from other development methodologies. It's been used to help teams and organizations overcome a variety of common challenges that Scrum is designed to force them to confront. When the context requires, it's been used to help organizations evolve new Scrum-like processes and practices that work best for them—not simply as a means to accommodate inadequacies and dysfunctions Scrum exposed, but rather as a strategy to resolve those problems in a manner that is most effective for that environment. This latter outcome is obviously not something for which Scrum itself provides. These different paths reflect Scrumban's bottom line—the service-oriented pragmatism that most businesses value.

Ultimately, Scrumban has become a framework of empowerment. David J. Anderson, pioneer of the Kanban Method, recently stated:

> *Empowerment isn't about letting people do whatever they want, or assuming they'll somehow self-organize to produce the right outcome. Empowerment is about defining boundaries, and we do the same with children when bringing them up; we tell them things like when their bedtime is, where they're allowed to play, whether they're allowed to go outside the yard of the house, they're allowed to swim at the shallow end of the pool, they aren't allowed to jump from the diving board . . . all these things. So empowerment is about giving people clear boundaries, and then letting them use their initiatives inside the boundaries.*[2]

Scrumban is distinct from Scrum because it emphasizes certain principles and practices that are quite different from Scrum's traditional foundation. These include the following:

- Recognizing the role of management (self-organization remains an objective, but within the context of specific boundaries)
- Enabling specialized teams and functions
- Applying explicit policies around ways of working
- Applying laws of flow and queuing theory

Scrumban is distinct from the Kanban Method in the following ways:

- It prescribes an underlying software development process framework (Scrum) as its core.
- It is organized around teams.
- It recognizes the value of time-boxed iterations when appropriate.
- It formalizes continuous improvement techniques within specific ceremonies.

2. http://tiny.cc/DavidAKanban (March 2013).

Stop Drinking the Kool-Aid

Mike Cohn, a leader in the Agile/Scrum community, recently "criticized"[3] Scrum teams for not being "focused on finding innovative solutions to the problems they [teams] are handed." He wasn't actually criticizing Scrum; rather, he was criticizing a mindset that has evolved among Scrum practitioners—a mindset that favors a safe approach to completing work over innovation.

I see a related phenomenon in my coaching engagements. The biggest impediment to improvement often lies with team members who are either unable or unwilling to think about improving the way work is done (or how their work is relevant to creating business value).

I believe the problem runs even deeper than this. A cult of Scrum has arisen that permeates the industry that has developed around Scrum. Not only are Scrum practitioners failing to pursue innovation in their work, but they are also failing to pursue innovation in the process. Scrum has evolved over the years as new information was discovered, yet there seems to be a growing resistance among its most ardent practitioners to reflecting on how to support its fundamental purpose.

I saw this reluctance most recently during a presentation to an Agile community in Boston. At the beginning of the presentation, I asked the audience how many of them were using Scrum in their environments. About half the audience members raised their hands. Then I asked how many had experienced challenges in adopting Scrum in their organizations. Not a single hand went down.

As I began describing some of the alternative ways Scrumban enables teams and organizations to achieve their desired purposes, debates erupted over whether prescribed or popular approaches associated with Scrum were ineffective. Despite my repeated emphasis that Scrumban simply represents *alternative* or *additional* paths when some aspect of the Scrum framework isn't fulfilling its intended purpose in a particular context, the majority of people in the room—even those who acknowledged challenges with their Scrum adoptions—were more interested in defending their existing process than in considering whether an alternative approach could help them overcome a challenge.

This cult-like mentality is not limited to the Scrum community. Pick your method or framework, and you'll find a lot of similar behavior—even in Kanban.

Fortunately, most day-to-day practitioners are pragmatists and realists. Simple pragmatism and a willingness to experiment are why Scrumban has evolved to become the framework described in this book. Unfortunately, there will always be a fringe set of "thought leaders" who perpetuate framework debates because they are unwilling to promote the benefits of an approach that "competes" with models in which they've invested significant amounts of both time and money. Scrumban may

3. http://tiny.cc/cohnscrumcricism.

be somewhat less threatening because of its familiar elements, but they will criticize it anyway.

Scrumban is a framework that almost forces its practitioners to accept the reality that good ideas can come from anywhere. It encourages people to actively pursue this reality at every turn. The question readers must answer for themselves is whether they're ready to accept this reality and embrace exploration, or whether they will remain trapped within a narrower perspective, justifying this mindset by the existence of a "debate" that is perpetuated more out of fear than out of fact.

Let's Get Started

One of the most powerful characteristics of Scrumban is the fact it can be implemented at any level of the organization—you don't need the authority of someone in command and control to begin making a difference in how you work (though it's certainly easier if you do have some degree of buy-in).[4] It also tends to be contagious.

So whether your goal is to have your company become a market leader or simply to gain better control over your own environment, I invite you to join the Scrumban community's Scrumban.io[5] LinkedIn group or www.facebook.com/Scrumban and to follow the Scrumban blog at www.scrumban.io.

4. This can be true even in environments where development processes are subject to audit, though the nature and extent of evolutionary change may be more limited than in less restrictive contexts.
5. The Linkedin group is available at www.theagilecollaborative.net, which redirects to https://www.linkedin.com/groups/Scrumbanio-7459316.

Chapter 5

MOBILIZE: ROLLING OUT SCRUMBAN

IN THIS CHAPTER

- How Framework Choices Influence Outcomes
- A Step-by-Step Guide to Getting Started
- Using Scrumban to Stabilize a Team before You Improve

Having outlined the foundational principles and mechanics that make Scrumban successful, it's time to discuss how teams and organizations can start employing Scrumban. Rolling out Scrumban doesn't have to require a lot of effort. In fact, the approach outlined in this chapter can be completed in a single day.

Your Starting Condition

There are essentially three potential starting conditions for rolling out Scrumban:

- You're working with a team or organization for which both Scrum and Scrumban represent new ways of working.
- You're working with an existing Scrum team or organization that will use Scrumban to improve its mastery of the responsibilities and ceremonies associated with the Scrum framework.
- You're working with an existing Scrum team or organization that will use Scrumban to monitor performance, diagnose issues, and adapt existing practices to the form best suited for their context.

It's best if all teams participating in a formal kickstart program share the same origin, but it won't necessarily be fatal if they don't.

When You're New to Scrum

Scrum can be difficult to master, even though it is a relatively simple framework. Recognizing this reality, we've made Scrumban our chosen framework for introducing Scrum to teams and organizations for the first time.

Because Scrumban seeks to minimize the disruptions associated with imposing new definitions and responsibilities upon employees, rolling out Scrum under this framework is substantially different from traditional approaches. Rather than starting out with Scrum-specific orientation and training, we emphasize discovery of existing systems and processes, then use the framework to gradually introduce elements of Scrum as warranted by the context.

This gradual introduction can be both role based and process based. For instance, the "daily scrum" is a process-based change the team can begin to employ within the context of a Scrumban framework, just as new Scrum masters can be eased into their responsibilities one element at a time.[1]

The Kickstart Process

Coaching Tip!

Prefer to bypass the background detail provided in this section? Check out our quick Kickstart reference in the Appendix.

The kickstart process covered here is particularly effective when you're faced with limited time or resources—in other words, when teams or organizations need to better manage workflow, but don't have sufficient resources to develop those better ways. Naturally, this is not the only way to introduce Scrumban to teams.[2] Always use your understanding of Scrumban principles to modify the process to fit your context.

Incidentally, this process is modeled after a Kanban kickstart process developed by Christophe Achouiantz, a Lean/Agile coach, and Johan Nordin, development support manager at Sandvik IT. Sandvik needed a way to empower its company's teams to begin a process of continuous improvement that wouldn't involve a significant initial investment of time or effort. Achouiantz and Nordin have documented this process and made it available to the public as a reference guide. While the particulars

1. In some contexts, the role of Scrum master may even remain optional. In working with a variety of organizations over the years, Frank Vega notes that one of his earliest efforts to help a team move toward more iterative and incremental ways of working dispensed with this role altogether. It remains one of the most effective teams he's ever worked with. I don't necessarily consider this an ideal approach, but mention it only to underscore the importance of contextual discovery and decision making.
2. For example, the approach taken by Siemens Health Services (see the Appendix for the full case study) would not have called for individual teams to engage in a process of systems discovery and definition.

may not apply directly to your needs, it's an outstanding summary of the key points and objectives relevant to any process.

Preparation

As systems thinkers, it's critical to first understand the context of the environment before attempting to kickstart anything. At a minimum, preliminary preparation should include the following steps:

- Identifying the current conditions and organizational priorities (e.g., increased quality, productivity, predictability)
- Developing working relationships with key managers and team leads
- Ascertaining areas of desired change
- Introducing Kanban/Scrumban as a framework for evolutionary change
- Starting to educate staff on Scrumban's core principles and practices
- Identifying any risks and potential impediments to introducing Scrumban to targeted teams

Regarding that last item, it's important to recognize that some teams or contexts may not benefit from introducing Scrumban until certain conditions or impediments are addressed. These can include teams in an active state of reorganization, teams with serious internal conflicts, and so forth. Your context will determine how you address these situations. For example, Sandvik's needs dictated that such teams be bypassed until circumstances changed.

Though it's possible for teams to initiate Scrumban adoption without organizational buy-in, as previous chapters indicate, broader engagement is necessary to achieve the full breadth of desired outcomes and to sustain them over time. As such, Scrumban is not substantially different from a pure Scrum context.

PMO/Program Managers:

If you're overseeing a pilot program or broad-based transformation effort, we strongly encourage you to consider the recommendations outlined here. The choices you make in the early stages will greatly influence your ultimate success.

In his book *Succeeding with Agile: Software Development Using Scrum* (Boston: Addison-Wesley, 2009), Mike Cohn addresses a variety of considerations that apply when you're selecting the project context in which to roll out a new process (these considerations are made with an eye toward building off demonstrated success). Though less critical in a context where the Scrumban or Kanban framework is employed, they nonetheless represent worthy

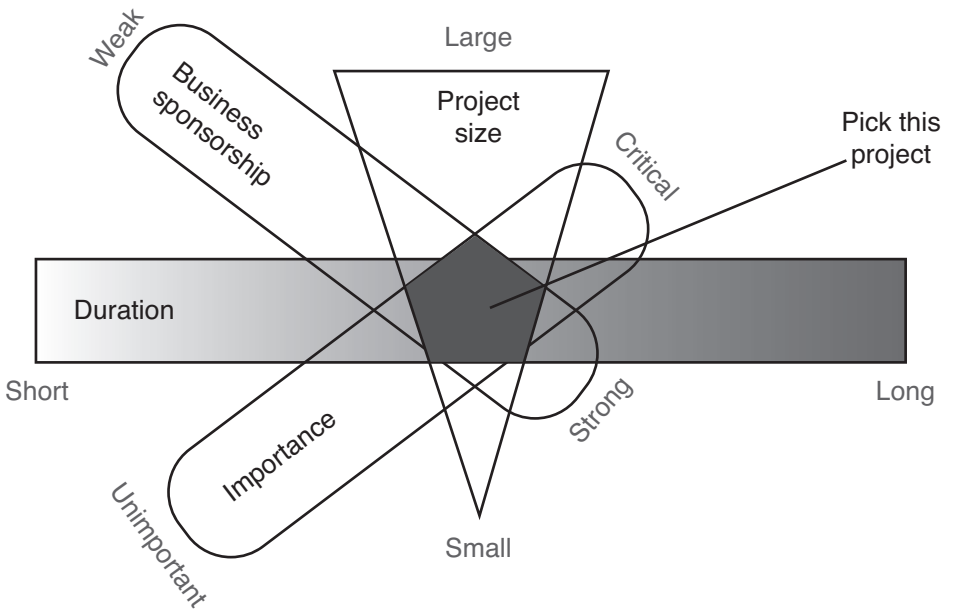


FIGURE 5.1 The convergence of key considerations in selecting a pilot project for introducing Scrum. These same considerations apply to Scrumban.

Source: Michael Cohn. *Succeeding with Agile: Software Development Using Scrum* (Boston: Addison-Wesley, 2009).

considerations (see Figure 5.1 for a graphical illustration of the convergence of these considerations):

- ***Project size***: For Scrum pilots, Mike suggests selecting a project that will not grow to more than five teams, and limiting initial efforts to just one or two of those teams. Coordinating work among more Scrum teams represents more work than you can effectively tackle.

  For Scrumban rollouts, there's substantially more leeway. With fewer concepts to learn and master (again, we start out respecting current roles and processes), working with a slightly larger number of teams is feasible.
- ***Project duration***: Select a pilot project of average duration for your organization, ideally around 3–4 months. This is plenty of time for teams to begin mastering the framework and seeing benefits. It's usually also sufficient for demonstrating that your new approach will lead to similar success in other contexts.
- ***Project importance***: Mike suggests that with Scrum pilots, it's critical to select high-profile projects. Unimportant projects will not get the organizational

attention necessary to make Scrum successful, and team members may be disinclined to do all the hard things Scrum requires.

There's slightly more leeway with Scrumban. Building momentum and trust through success is still an important objective, but the framework's service-oriented agenda and active management of psychological barriers to change represent additional capabilities that can be leveraged to ultimately satisfy these needs.

Executives:

Your efforts to educate your organization's nontechnical employees about how this undertaking is important to the business, and encouraging their active engagement in the process, will go a long way toward creating long-term success.

- *Business owner's level of engagement*: As previously mentioned, Scrum and Scrumban ultimately require changes in the way both the business and the technology sides of the house approach things. Having an engaged player on the business side can be invaluable for overcoming challenges and evangelizing success across the organization.

  Scrumban modifies the weight of this factor substantially. To function properly, Scrum requires active engagement on the business side—its prioritization and feedback functions depend on it. Scrumban won't function well without business engagement, but it does afford teams the ability to create positive changes by allowing knowledge discovery to "stand in" for disengaged business players.

  In a similar vein, Scrumban is ultimately designed to respond to business demands. If the business isn't demanding change, then Scrumban's pragmatic "response" is to reflect the absence of any need to change (although a culture of continuous improvement will continue to spawn incremental changes to make what you're already doing well even better).

Initial Considerations

Many topics can be addressed during a kickstart. What constitutes "information overload" for a particular group will vary from context to context, and must always be judged against the level of ongoing support available following the kickstart (for example, you will likely cover more material if teams will have coaching support following the kickstart than if they will have none at all).

With these considerations in mind, incorporating themes that reinforce the service-oriented philosophy and the organization's core values will go a long way toward positioning teams to evolve more effectively. There are many ways to do this, and it's one area where an experienced practitioner or coach will really add value to the process.[3]

One special consideration for existing Scrum teams may be to introduce the concept of Sprint 0 (if this tactic is employed within the involved team or organization). There's much debate about the "appropriateness" of Sprint 0 in Scrum circles. These special sprints are often described as necessary vehicles for managing what needs to be done before a Scrum project can start. For example, the team may need to be assembled, hardware acquired and set up, and initial product backlogs developed. Purists' feathers may be ruffled by the notion of differentiating sprints by type and, more importantly, by the idea that any sprint would be structured in a way that fails to deliver value.

The Scrumban framework obviates the need for debate on these issues. The tasks associated with ramping up a new project can be easily accommodated and managed within the Scrumban framework as a special work type. If it's important for your environment to ensure Scrum ceremonies hold true to their Agile objectives, then Scrumban provides a ready solution.

It also makes sense to assess existing Scrum practices with an eye toward understanding their impact on performance. Larry Maccherone and his former colleagues at Rally Software have analyzed data from thousands of Scrum teams in an effort to assess how differences in the way Scrum is practiced affect various elements of performance. This data mining exposed some interesting realities about how our choices involving Scrum practices can influence various facets of performance. Incidentally, Maccherone's analysis is consistent with data mined from hundreds of teams using my own Scrum project management platform (ScrumDo.com).

Maccherone elected to adopt a "Software Development Performance Index" as a mode of measuring the impact of specific practices. The index is composed of measurements for the following aspects:

- ***Productivity*:** Average throughput—the number of stories completed in a given period of time
- ***Predictability*:** The stability of throughput over time—how much throughput values varied from the average over time for a given team
- ***Responsiveness*:** The average amount of time work on each user story is "in process"
- ***Quality*:** Measured as defect density

3. As previously noted, the Siemens Health Group case study (see the Appendix) represents a great example of how expert assistance played a critical role in contributing to the depth of the company's success.

How You Choose to Organize Makes a Difference

Especially For:
Managers & Executives

As noted, Larry Maccherone's analysis reflects a definite correlation between certain choices and software development performance. Although correlation does not necessarily mean causation, we can use these relationships as a guide for tailoring a kickstart process to address specific factors relevant to a particular implementation. Teams can be guided to position their visualizations and practices to better understand and discover which choices regarding Scrum practices are likely to work best for their desired outcomes.[4]

Determining Team Size

Humans are the slowest-moving parts in any complex organization. Teams can help us counteract this reality by making us smarter and faster. However, this outcome is possible only if we get teams right. Team dynamics is Scrum's strong suit—and an arena Kanban addresses only tangentially, if at all. In fact, providing a framework that helps us improve team dynamics is a great example of how Scrum boosts Kanban's capabilities.

To this end, size stands out as the most significant predictor of team success. There's a right size for every team, and like so many other aspects of managing systems, that size should be dictated by overall context. Even so, having guidelines doesn't hurt.

The military is a great starting point for gaining perspective on ideal team size. The basic unit in the U.S. Army's Special Forces is the 12-person team. The army arrived at this number after recognizing there are certain dynamics that arise only in small teams. For example, when a team is made up of 12 or fewer people, its members are more likely to care about one another. They're more likely to share information, and far more likely to come to each other's assistance. If the mission is important enough, they'll even sacrifice themselves for the good of the team. This happens in business, too.

The founders of Scrum understood these dynamics. Ask anyone in the Scrum community about ideal team size, and you'll hear 7 members plus or minus 2.

The reasons for maintaining small team sizes are valid, but not every 12-person military unit is right for a particular assignment. Likewise, not every 9-person Scrum team is right for a given environment. Your system ultimately will dictate your needs, and it may very well require expanding your teams to incorporate a larger number of specialists or address some other need.

---

4. In sharing this data, Maccherone has been diligent in emphasizing that there is no such thing as "best practices" relevant to all contexts. Each team and organization must discover what works best at any given point in time.

Scrumban supports this flexibility through its enhanced information sharing and visualization capabilities. Even the cadence of daily stand-ups is different: We've seen daily stand-ups in a Scrumban environment involving almost 100 individuals that are both effective in addressing organizational needs and capable of being completed within 15 minutes. This would be impossible in a Scrum context.

Iteration Length

Many organizations seek to establish uniform iteration lengths because they mistakenly believe this is a good approach for aligning different systems to the same cadence. Yet for many years, the general consensus among Scrum practitioners has been to recommend two-week iterations. What does the data tell us?

Rally Software's data shows that almost 60% of the software development teams using its tool adopt two-week sprints, and I've seen similar patterns among users of ScrumDo. The data also suggests a team's overall performance tends to be greatest at this cadence (Figure 5.2).

Dig more deeply, however, and differences begin to show up among the various components of the index. Throughput, for example, tends to be greatest among teams adopting one-week iterations (Figure 5.3).

Quality, in contrast, trends highest among teams adopting four-week cadences (Figure 5.4).

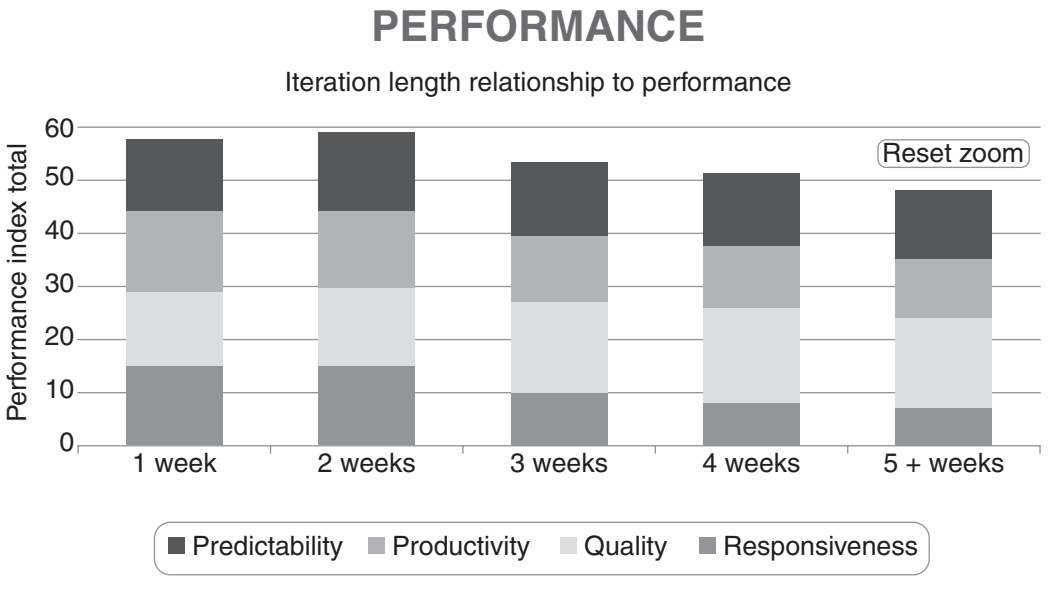


FIGURE 5.2 A view into the relationship between iteration length and team performance.

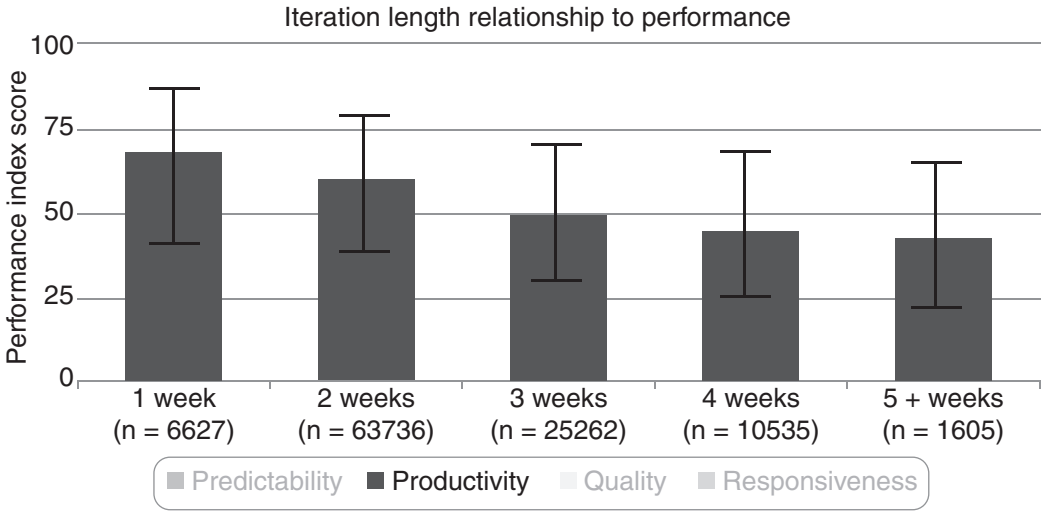


FIGURE 5.3 A closer look at productivity.

Source: © 2014 Rally Software Development Corp. All rights reserved. Used with permission.

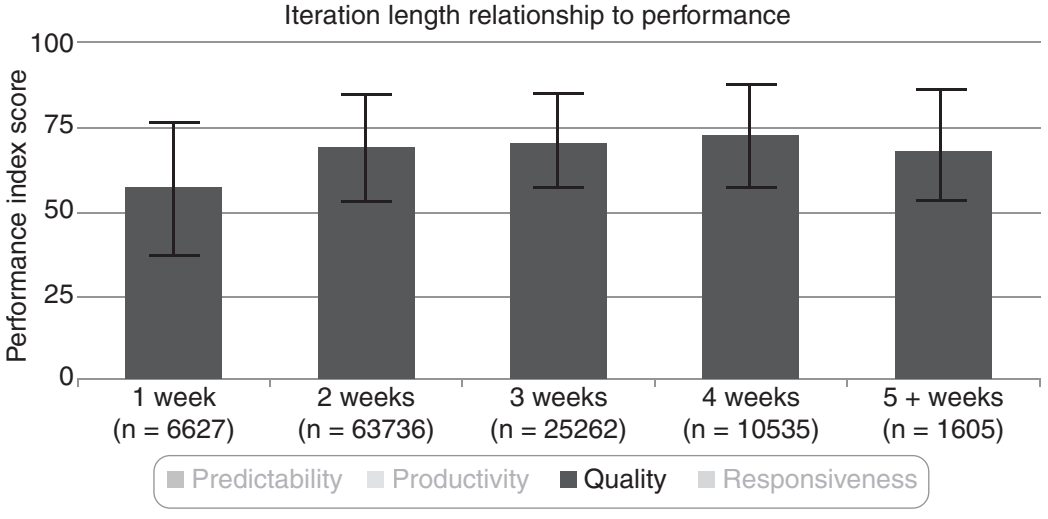


FIGURE 5.4 A closer look at quality.

Source: © 2014 Rally Software Development Corp. All rights reserved. Used with permission.

Every organization possesses characteristics that drive different outcomes, which is where Scrumban's visualization and added metrics can help teams discover what's best for their unique circumstances. The higher throughput typically associated with two-week cadences won't magically materialize if the historical lead time for most of your completed work items is three weeks. Similarly, delivering completed work every two weeks won't magically produce better outcomes if the customers can't accept it at that rate.

Also, be wary of inferences drawn from data associated with teams that have adopted longer iterations. For example, Frank Vega recently made an interesting observation on this topic, recounting instances where teams chose longer iteration periods as a means of opposing (consciously or subconsciously) Agile transformation efforts because agreeing to a shorter iteration cycle would counteract their position that nothing of any real value could be delivered in such a short time period.

Estimating Story Points

Scrum prescribes the Sprint Planning ceremony as an event that determines which specific stories can be delivered in the upcoming sprint and how that work will be achieved.

Although Scrum itself doesn't prescribe a particular approach or method that teams should use to assist with this decision making (other than expecting it to be based on experience), most Scrum teams use story points and team estimation techniques as components of this process.

Another process that teams use to aid their estimation efforts is Planning Poker—a consensus-based technique developed by Mike Cohn. With this approach, which is used primarily to forecast effort or relative size, team members offer estimates by placing numbered cards face-down on the table (rather than speaking them aloud). Once everyone has "played," the cards are revealed and estimates discussed. This technique helps a group avoid the cognitive bias associated with anchoring, where the first suggestion sets a precedent for subsequent estimates.

Scrumban enables teams to begin measuring the correlation between story point estimates and the actual time spent completing development, measured from the time work on a story begins until the story is completed. Some teams reflect a strong correlation between their estimates and actual work time. Others are more sporadic, especially as the estimated "size" of stories grows. Teams using exponentially based story size schemes tend to be more precise with their estimates. Comparing the two charts in Figure 5.5 and Figure 5.6, it's fairly evident that exponential estimation reflects a tighter band of variability in lead time—especially among smaller stories. The spectrum of variability is much wider across the Fibonacci series.

As with the data on sprint length, the way teams estimate the size and effort of work tends to correlate with overall performance (Figure 5.7).

It is not uncommon to find a lack of correlation between a team's velocity, the average number of story points completed during a sprint, and actual throughput. Nonetheless, there are many instances when the estimating process works well for

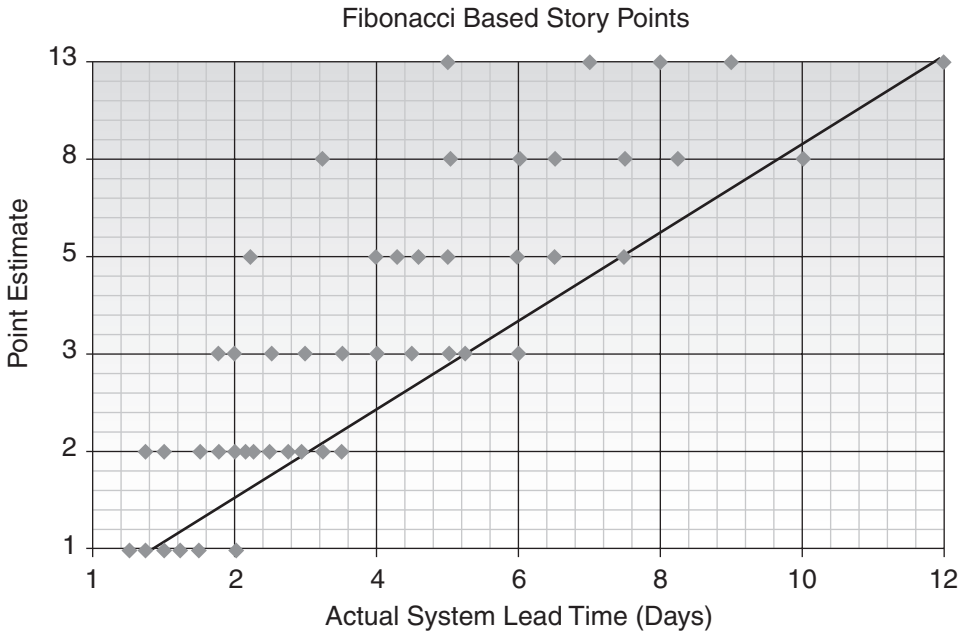


FIGURE 5.5 There tends to be only a loose correlation (if any at all) between estimated story points and actual lead time.

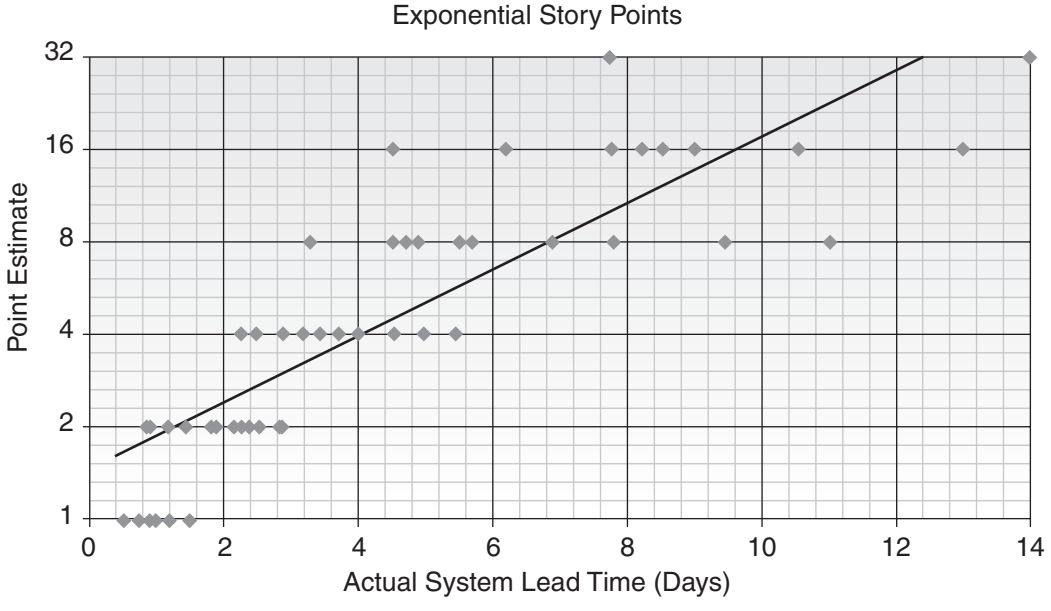


FIGURE 5.6 The correlation is slightly better in an exponential points scheme.

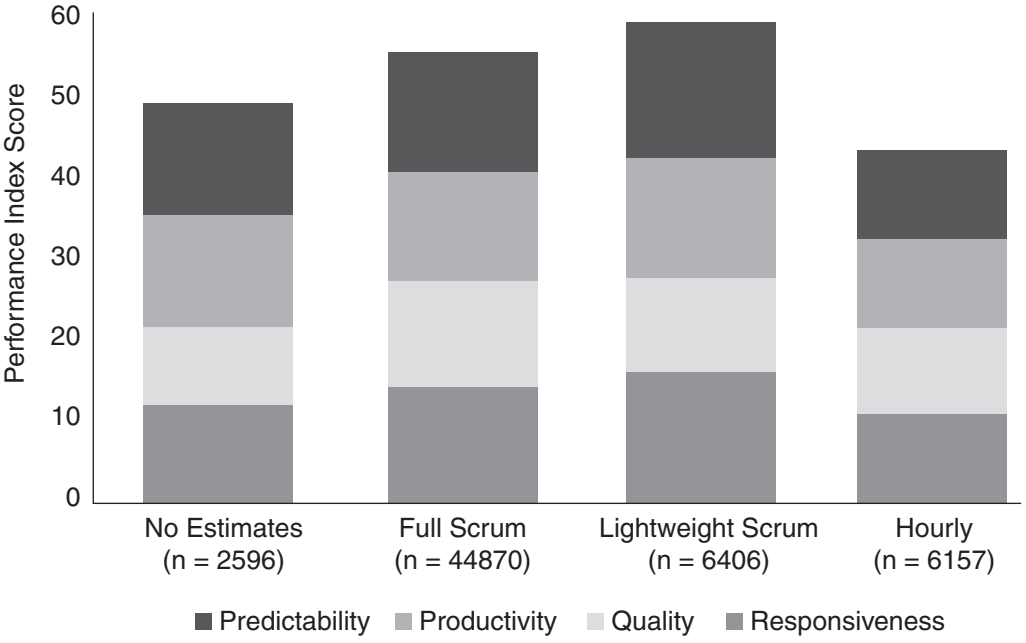


FIGURE 5.7 The relationship between team performance and estimation schemes.

individual teams. As mentioned previously, it's more challenging to work with this metric on a portfolio level involving multiple teams. Calculating relative correlation is one way that Scrumban helps teams gain a better sense of whether the ceremonies they engage in, such as estimation events, are providing sufficient value to justify the investment of time and effort.

Scrumban Stories: Objective Solutions

Objective Solutions' teams showed a lack of correlation between their story point estimates and actual lead time, but this discrepancy wasn't noticed until after the company's implementation of the Scrumban framework. As the organization opted to continue using team estimates for forecasting purposes, its solution for improving predictability was to apply a "velocity factor" derived from the difference between these values. To avoid the influence of Parkinson's law (the tendency for the time needed to complete work to expand to the time allowed for it), these velocity factors were not communicated to the team members, and only used by the product owners and Scrum masters engaged in planning.

Read the full story of Objective Solutions in the Appendix.

Contextual Responsibilities

As teams begin to employ Scrumban, members will assume a range of new "responsibilities."

First and foremost, team members must be groomed to challenge and question their team's policies, facilitating team interest in and ownership of how they work. Fortunately, this basic concept should not be foreign to people already used to a Scrum context. A great way to promote and support this mindset is through the establishment of "working agreements."[5]

Not surprisingly, Scrum's existing ceremonies and artifacts implicitly support the notion of working agreements. Scrumban's framework goes one step further, calling for the team to make such agreements explicit and to visualize them on their boards as appropriate. In many respects, they are akin to establishing an internal service level agreement between team members.

Ultimately, we want to develop leaders who help all team members acquire the ability to see and understand concepts like waste, blockers, and bottlenecks. Good leaders also ensure teams don't get stuck in a comfort zone, by prodding team members to always seek out ways to improve.

As previously mentioned, although teams will experience work improvements simply from employing Scrumban independently, an active management layer is essential to realizing benefits at scale. Systemic improvements are less likely to occur without a manager who maintains contact, helps resolve issues outside the team's domain, and oversees the extent to which teams devote time and effort on pursuing discovered opportunities for improvement. Setting expectations for this need at the kickstart helps ensure a more sustainable implementation.

Scrumban Stories: Objective Solutions

Like Siemens Health Services, Objective Solutions had successfully adopted Scrum/XP practices in its development operations but was struggling with a number of challenges at scale. The company adopted elements of Scrumban to address these challenges. Not surprisingly, this journey allowed the organization to improve many of its Scrum and pair programming practices that had been negatively influencing throughput and quality.

For example, Objective Solutions was experiencing challenges with managing the pair programming process: Keyboard work was not balanced, the time required to complete work unnecessarily expanded, and mentoring benefits were not being fully realized. Scrumban allowed this company to recognize specific problems, and to implement processes for managing its pair programming practices in the course of managing its regular workflow.

Read the full story of Objective Solutions in the Appendix.

5. See, for example, http://tiny.cc/AgileTeamAgreements.