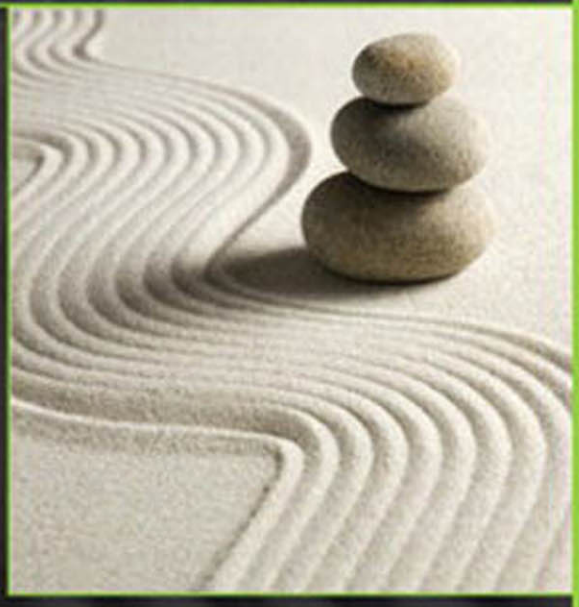
The Addison-Wesley Signature Series

A Mike Cohn Signature Book

Succeeding with Agile

Software Development Using Scrum

Mike Cohn



Foreword by Tim Lister

Praise for *Succeeding with Agile*

"Understanding the mechanics of an agile process is just not enough. Mike Cohn has compiled a superb and comprehensive collection of advice that will help individuals and teams with the intricate task of adopting and adapting agile processes to fit their specific challenges. This book will become the definitive handbook for agile teams."
—**Colin Bird**, Global Head of Agile, EMC Consulting

"Mike Cohn's experience working with so many different organizations in the adoption of agile methods shines through with practical approaches and valuable insights. If you really want agile methods to stick, this is the book to read."
—**Jeff Honious**, Vice President, Innovation, Reed Elsevier

"Mike Cohn has done it again. *Succeeding with Agile* is based on his experience, and all of our experience, with agile to date. He covers from the earliest days of the project up to maturity and offers advice for the individual, the team, and the enterprise. No matter where you are in the agile cycle, this book has something for you!"
—**Ron Jeffries**, www.XProgramming.com

"If you want to start or take the next step in agile software development, this book is for you. It discusses issues, great solutions, and helpful guidelines when scaling up in agile projects. We used the guidelines from this book extensively when we introduced agile in a large, FDA-regulated department."
—**Christ Vriens**, Department Head of MiPlaza, part of Philips Research

"If making the move to agile has always baffled you, then this book will unlock its mysteries. Mike Cohn gives us all the definitive, no-nonsense guide to transforming your organization into a high-powered, innovative, and competitive success."
—**Steve Greene**, Senior Director, Program Management and Agile Development, www.salesforce.com

"Mike Cohn is a great advisor for transforming your software organization. This book is a distillation of everything Mike has learned over the years working with companies that are trying to become more agile. If you are thinking of going agile, pick up this book."
—**Christopher Fry**, Ph.D., Vice President Development, Platform, www.salesforce.com

"Whether you're just starting out or have some Scrum experience under your belt, in *Succeeding with Agile*, Mike Cohn provides a wealth of information to guide you in your quest toward continuous improvement. Throughout the book, concepts are reinforced with practical everyday advice, including how to handle objections and thought-provoking 'things to try now.' An extensive list of recommended readings round this out to be a must have book."
—**Nikki Rohm**, Studio Director Project and Resource Management, Electronic Arts

considering the performance of each, the team will hopefully be able to choose the most appropriate ScrumMaster.

Bob Schatz and Ibrahim Abdelshafi of Primavera Systems point out another reason why rotating might be useful.

> With time the team can begin to treat this position as their manager. And the person in that position typically detects and dutifully fills the apparent need. The result is a breakdown in the team's self-management practice. By rotating the responsibility at the start of each sprint, it diffuses the role and makes it a shared team responsibility and establishes a balance of power. (2006, 145)

So, although it is possible to rotate the job of ScrumMaster, I recommend doing it only for specific reasons, such as those just given, and only temporarily. Rotating should not be a permanent practice. There are simply too many problems with it, including the following:

- Someone who has rotated into the role usually has other, non-ScrumMaster tasks to perform during the sprint, and these often take priority.
- It's hard to train enough people to do the role well.
- Some people will use their time as ScrumMaster to try to push through changes to the process.
- Designating someone as ScrumMaster for a sprint or two does not automatically make someone value the job, which can lead to ScrumMasters who think Scrum is a mistake.

Overcoming Common Problems

Some of the common problems you may face in making sure that each team has the appropriate ScrumMaster and what you can do to address them include

Someone inappropriate takes the role. Sometimes the decision of who should be the ScrumMaster is made for you: someone just says, "I'll do it," and takes on the role. Often this is great—after all, good ScrumMasters are likely to be the ones who take on additional responsibility before being asked. But what if the person who volunteers is inappropriate for the role? Your response to this will depend on your role in the organization.

If you have some authority over the inappropriate ScrumMaster, the team, or the adoption of Scrum, meet with the volunteer and explain why you need someone different in the role. If appropriate, give the volunteer specific things you would like him to do to be considered as a ScrumMaster candidate later. And if the inappropriate person is already in the role of ScrumMaster? Even though it will be a bit more difficult, I still suggest removing the person from the role if

you are convinced the person is truly inappropriate. In either case, act swiftly. An inappropriate ScrumMaster should be changed as soon as possible; I haven't met a team yet who felt an inappropriate ScrumMaster was removed too soon.

If you do not have authority over the ScrumMaster, team, or process, I still suggest you pursue a conversation with the person who has inappropriately assumed the ScrumMaster role. Approach the discussion from the perspective of having the team's best interests in mind. Try to accentuate the ScrumMaster's strengths and suggest that he might be able to find a better way of applying them to the project if he steps out of the ScrumMaster role.

The ScrumMaster is also a programmer/tester/other on the team. When it is impractical to have a dedicated ScrumMaster for a team, a decision must be made between a ScrumMaster who splits time between two or more teams and a ScrumMaster who is both a ScrumMaster and a programmer, tester, or other on the same team. Although either of these approaches can work suitably well, I tend to prefer that when necessary a ScrumMaster's time is split between two teams. Having a ScrumMaster who is also an individual contributor on the team carries many risks.

One risk is that the person may not have adequate time to devote to both roles. Another is that someone in a combined role will probably need to stay away from critical path activities because the person could be interrupted with ScrumMaster duties at any time. A more subtle risk is that other team members will not easily know whether they are talking to their ScrumMaster or to another individual contributor. Yet another risk is the ScrumMaster will have less credibility when protecting the team from outsiders. A dedicated ScrumMaster will have more credibility when saying, "We can't help. The team is busy," than will the ScrumMaster/individual contributor whose same message can be interpreted as "We can't help. I'm busy."

As risky as it can be for someone to be both ScrumMaster and a technical contributor on the project, it is a common situation. Awareness of these issues and a willingness to work through them as they arise is often the best solution.

The ScrumMaster is making decisions for the team. This problem can arise for two completely different reasons: it could be because the ScrumMaster misunderstands or is uncomfortable in the new role, or it could be because the team is used to someone else making decisions. In either case, the solution is the same. The ScrumMaster should be taken aside and reminded that being a ScrumMaster is about providing guidance, not answers.

As a new ScrumMaster, one of the first things I had to learn was how to count. When we'd be in a meeting, struggling with some vexing problem, the team would look to me to tell them the solution. Having previously been the

team leader I was tempted to blurt out "the answer." But I needed the team to learn how to find the right answers themselves, and so I sat there quietly counting to myself. 1, 2, 3…. I counted well into the hundreds on a few occasions, but this helped me learn to keep my mouth shut. And it helped the team learn not only how to make those decisions but also that I wouldn't do it for them.

The Product Owner

I think of the ScrumMaster as the person who ensures that the team is working well together, that impediments to progress are quickly removed, and that the team is moving efficiently toward its goal. I think of the product owner as the person who makes sure the team is aimed at the right goal. A good team needs both roles to succeed. The product owner points the team at the right target; the ScrumMaster helps the team get to that target as efficiently as possible.

Roman Pichler, author of *Agile Product Management with Scrum: Creating Products that Customers Love*, stresses the importance of the product owner: "The product owner has the authority to set a goal and shape the vision. The product owner is not just a project manager who now also writes requirements and does a little bit of prioritization." Thinking of the product owner as the provider of a team's goal helps make certain aspects of the product owner's job clear. For example, the product owner is clearly responsible for defining and prioritizing the product backlog that expresses the goal. Similarly, the product owner is responsible for making sure the project earns a good return on the investment made in it.

Responsibilities of the Product Owner

Compiling an exhaustive list of the responsibilities of the product owner would be difficult. Every application exists within its own context of company culture, individual and team competencies, competitive forces, and so on. This context strongly influences how the product owner role is performed in different companies. So instead of providing a checklist of product owner responsibilities ("must attend sprint planning meeting"), I find it more helpful to think in terms of two things that a product owner provides the team: a vision and boundaries.

SEE ALSO

See *Agile Product Management with Scrum: Creating Products that Customers Love* by Roman Pichler for a thorough discussion of the product owner role.

Providing Vision

Many of the product owner's responsibilities involve establishing and communicating the vision for the product. The best teams are those whose passion has been ignited by a compelling vision shared by the product owner. Who will we be selling to? What is unique about our product? What are our competitors doing? How will our product evolve over time? Of course, the questions are different for an application or service that is being delivered to a group of in-house users, but

having a shared vision is important for motivating a team and creating a long-term connection between those developing the product and those using it.

Beyond having a clear vision in mind, the product owner must elucidate that vision for the team. The product owner does this through creating, maintaining, and prioritizing the product backlog. There is a lot of dissension among ScrumMasters and teams as to whether the product owner is the one who actually writes the product backlog. I am firmly in the camp of I-don't-care. It doesn't matter to me who performs the physical act of writing the product backlog; what does matter is that the product owner is the one who makes sure it happens. If the product owner delegates this to a business analyst and the analyst gets sidetracked and fails to write the product backlog, it is still the product owner who is responsible.

SEE ALSO

The product backlog is a prioritized list of features to be added to a product. It is fully described in Chapter 13, "The Product Backlog."

Beyond ensuring that the product backlog exists, the product owner adds detail to the vision by answering questions team members will have: Do you want it to work this way? What did you mean when you said such-and-such? Although the product owner can delegate or distribute the responsibility for answering these questions, the product owner cannot delegate the responsibility that they indeed get answered. A product owner can say, "Talk to Nirav if you have questions about how the shopping cart and checkout features should work," but if Nirav isn't responsive or helpful, a good product owner will step in and answer questions personally, find out why Nirav is unable to do so, designate a different person, or find some other solution.

Providing Boundaries

Vision and boundaries can be thought of as competing aspects of the project. The vision shows what the product can become. The boundaries describe the realities within which the vision must be realized. Boundaries are provided by the product owner and often come in the form of constraints, such as

- I need it by June.
- We need to reduce the per-unit cost by half.
- It needs to run at twice the speed.
- It can use only half the memory of the current version.

Often when I tell groups that the product owner is allowed to dictate things such as this—especially the date—I am met with angry responses. "No," they tell me, "estimates are up to the team. All the product owner does is prioritize the work." Although those statements are true, the product owner is also responsible for defining the boundaries that will determine the success of the product.

Most experienced Scrum team members will readily agree that it is within the product owner's purview to say, "We need to develop at least this much of the product backlog or the product won't be worth shipping." But many of these same experienced people resist when similar statements are made about deadlines.

But, let's see what Takeuchi and Nonaka had to say in their study of the six teams that formed the foundation of Scrum and that were the subject of the first paper on Scrum back in 1986.

> Fuji-Xerox's top management asked for a radically different copier and gave the FX-3500 project team two years to come up with a machine that could be produced at half the cost of its high-end line and still perform as well. (139)

Here, we clearly have a team that is given a challenging problem—match the performance of the company's best current copiers but at half the cost—and a deadline for solving the problem. There is nothing wrong with this. Product owners go wrong when they overly constrain a problem or when they make a solution impossible. Had Fuji-Xerox's management given that team the same problem but only one month to solve it, the team would have seen the futility in the situation and not even tried. The problem as presented to that team presumably left the team plenty of operating room in which to find a solution. A part of the product owner's job that is more art than science is providing just enough of a boundary around the project so that the team is motivated to solve the difficult problem before them but not providing so many boundaries that solving the problem becomes impossible.

When brainstorming solutions to a challenging problem, common advice is to think "outside the box." However, there is evidence that better solutions emerge more easily from thinking that is done "inside the box" as long as the box has been properly framed (Coyne, Clifford, and Dye 2007). When we're told to think outside the box, they say, the total lack of constraints can be unsettling.

> Imagine a random product you are trying to improve in a typical facilitated brainstorming session. Outside-the-box possibilities could include making the product bigger or smaller, lighter or heavier, prettier or more rugged (or changing its appearance in any of a hundred ways). Further ideas could involve making the product more expensive or less or maybe breaking it into parts or bundling it with other products. They could involve changing the product's functionality, durability, ease of use, or the way it fits with other products. Or its availability, affordability, or repairability. How do you know which dimensions are fruitful to explore? Without some guidance, people cannot judge whether they should continue in the direction of their first notion or change course altogether. They cannot handle the uncertainty, and they shut down. (2007, 71)

The product owner's job is to create the new box—the boundaries—in which the team will think. This new box prevents the team from getting lost in

the infinitude of possible solutions and gives team members a basis for making and comparing choices. Boundaries for that new box are determined by the most important constraints for the business, which may involve things like minimum guaranteed functionality, dramatically faster performance, reduced resource consumption, and, yes, in some cases the date.

SEE ALSO

Schedule is one side of the infamous iron triangle of scope, schedule, and resources. The iron triangle is discussed in Chapter 15, "Planning."

Each Team Needs Exactly One Product Owner

On a team that is new to Scrum, the ScrumMaster job can be very time consuming. The ScrumMaster will be busy training team members on Scrum itself, encouraging them to think in different ways about the problems they encounter, removing impediments to the team's progress, and more. Early on, this might even be a full-time job, depending on the newness of the team and the types of impediments team members face. Over time, however, things improve. Eventually the ScrumMaster has removed many recurring impediments, and the team itself has begun to master Scrum and has embraced its self-organizing nature. As these changes occur, the team needs less and less of their ScrumMaster's time. If we were to graph a team's demands on its ScrumMaster's time, it would look something like Figure 7.1.

SEE ALSO

Self-organization is discussed in detail in Chapter 12, "Leading a Self-Organizing Team."

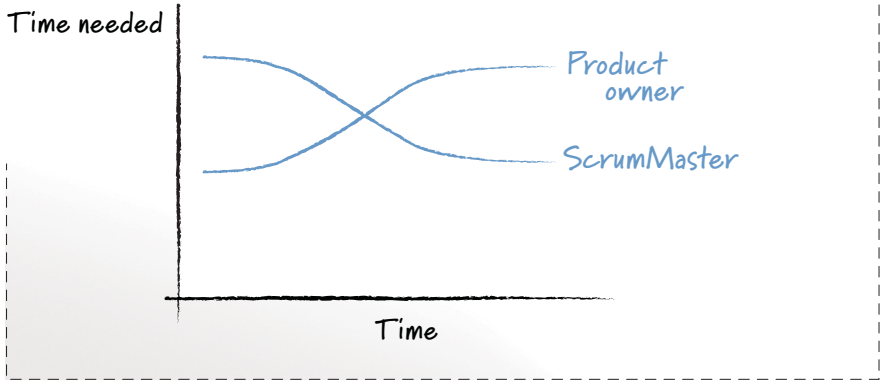


FIGURE 7.1

A team's time demands on their product owner and ScrumMaster move in different directions.

Contrast this with the team's need for its product owner. When the team first adopts Scrum, it will not be very good at it. It will struggle with how much detail to put on the product backlog, how much work can be completed in a sprint, how to work well together within the sprint, and so on. Team members will be learning new practices and new ways of working together. The team will not be very fast—at least not compared to how fast it will be after it gets good at Scrum. As the team speeds up (through its own improvements and from the ScrumMaster gradually removing impediments), it will be completing more work each sprint. This means members will have more questions for their product owner. Therefore,