The Addison-Wesley Signature Series

A Mike Cohn Signature Book

Scrum Shortcuts without Cutting Corners

Agile Tactics, Tools, & Tips

Ilan Goldstein

Foreword by Mike Cohn
Illustrations by Colin Tan

Praise for *Scrum Shortcuts without Cutting Corners*

"Great books give you advice you can follow, and *Scrum Shortcuts without Cutting Corners* most definitely does. Written to suit newcomers or experienced practitioners who have a healthy interest in Scrum, the knowledge contained in this book can be game-changing. Ilan Goldstein shares his extensive global experience to produce a well-written and valuable insight into practicing and sustaining effective agile practices."

—Kevin Austin, Agile coach and transition lead, Fortune 50 investment bank

"A software team succeeds because it has the right people who are allowed to do their best work. Understanding the patterns and anti-patterns (my favorite anti-pattern—'test sprint') in this guidebook will help you know who the right people are and how to help them work well. These shortcuts focus on people, and that's why they work. Get your team (and the rest of your company) reading and discussing this today."

—Lisa Crispin, coauthor with Janet Gregory, of *Agile Testing: A Practical Guide for Testers and Agile Teams*

"Ilan Goldstein has earned a loyal following in the Agile community for his no-nonsense advice and practical solutions that deliver real results for teams. I'm thrilled that he's been able to distill this expertise into a book that's rich with insights and also very readable. I can't wait to use all his best ideas in my own practice!"

—Pete Deemer, CEO of Stormglass and author of *The Scrum Primer*

"This book is an outstanding reference for anyone using Scrum to build software. Whether you are an experienced practitioner or a beginner just starting out, you'll find something worthy in here that you can learn and apply right away. Ilan's casual and engaging writing style describes perfectly the real-world challenges that you may face when using Scrum, and gives you practical guidance for working through them."

—Ryan Dorrell, CTO, AgileThought

"I especially like the essay style. It invites me to skip around to find topics of interest, and makes it easy to find what Ilan thinks about things. Ilan takes us over ground we've covered before, but he gives us a fresh look at things. Very valuable!"

—Ron Jeffries, coauthor of *The Agile Manifesto* and founder of xprogramming.com

"Scrum is not a solution. Your solution will only become clear through a journey of inspection and adaptation. The journey is not straightforward, and you will make some mistakes as you try to customize Scrum to your organization. Ilan's work for me is a Hitch Hiker's Guide to Scrum, giving you insights, tools, approaches, and belief to support you. I really appreciate the openness in Ilan's stories and how he shares his experiences. I don't have to agree with every technique or idea, as Ilan is not trying to instruct. He is asking you to think; to challenge your assumptions and help you on your way."

—Martin Kearns, Scrum trainer and national Agile and innovation lead at SMS Management & Technology

"Most books about Scrum are on theory and speak from a distant third-person perspective. They are hard to read. Ilan has created the opposite. He has created a book that feels like a conversation. I kept nodding in recognition at all the real-world problems he identifies and laughing at the pragmatic, humorous, and spot-on wisdom he shares. His writing flows smoothly and draws you in to the point where you will hate to put this book down at the end."

—Clinton Keith, Scrum trainer and author of *Agile Game Development with Scrum*

"With *Scrum Shortcuts without Cutting Corners*, Ilan Goldstein has delivered the must-have text for Scrum teams. The fact that Scrum is a framework is often used to justify tinkering with its fundamental mechanisms—to the extent that what was once Scrum becomes something altogether different and less effective. Goldstein clearly identifies the delineation between those mechanisms that can be tailored and those that must remain true to ensure success."

—Arik Kogan, business intelligence manager at Cougar Software

"A refreshing perspective on Scrum. Ilan will take you beyond the theory and share his real-world experiences, offering practical advice for successful Scrum adoption and maturity within your organization. His insights and philosophical views on the subject will keep you one move ahead in the game."

—John Madden, program manager at HotelsCombined

"Ilan has done some great work here. This book is an insightful look at what it takes to grow as a Scrum Master, and provides practical real-world experience to guide you on the journey. It's part practical advice and part story-telling woven together to make a book that is useful and enjoyable to read at the same time. I enjoyed the book and look forward to having a copy on my bookshelf. Ilan has done a wonderful job."

—Kane Mar, Scrum trainer; cofounder and president of Scrumology.com

"Scrum is deceptively simple, but as someone said, it is easy to do this in a mediocre way. I have fallen into many of the traps myself. In his book, Ilan shows you how to succeed with Scrum. The book makes it easy to find the information that is most helpful to you right now. Each of the many short chapters is to the point and a pleasure to read. I couldn't recommend it more."

—Jens Meydam, head of development, Zahnärztekasse AG

"Ilan's book isn't the usual cookbook on how to use Scrum in the workplace, but rather a toolkit of practical advice that covers every aspect of setting up self-organizing, high-performing teams using Scrum. It is a highly engaging, enjoyable and meaningful read for all Scrum practitioners."

—Michael Rembach, applications development manager, Transport for New South Wales

"Ilan's valuable tactics, tools, and tips for applying Scrum clearly illustrate his own hard-fought, on-the-job experience. This is not a theoretical book about Scrum, but instead a practitioner-focused, get-your-hands-dirty guide to getting the job done with Scrum. It is a delightful read with topics presented concisely and in an easily digestible way. *Scrum Shortcuts without Cutting Corners* is a critical addition to the body of Scrum literature, and a perfect follow-on to my *Essential Scrum* book!"

—Kenny Rubin, managing principal, Innolution, LLC and author of *Essential Scrum: A Practical Guide to the Most Popular Agile Process*

"If Scrum and Agile were easy, everybody would be doing it! Now that so many are, this book is the virtual Agile coach I wish I had when I was on the early steps of my Scrum journey. Ilan is a world-class coach, and he has packed this book full of ideas and approaches to all of the common questions and issues that are bound to come up as you transform your world of work to Scrum."

—Craig Smith, Agile coach and editor at InfoQ

"If *The Scrum Guide* is the rule book, then *Scrum Shortcuts without Cutting Corners* is the experts' guide to playing the game. Ilan Goldstein reveals all the little secrets that every team should know to be effective in adopting Scrum. From sprint lengths, to splitting down work, to relative estimation, Ilan tackles the gray zones in Scrum, offering sage advice in a world of 'it depends.'"

—Renee Troughton, Agile coach and author of *Agile Forest*

"Sharing what he has learned implementing Scrum over many years with many teams, Ilan goes beneath the surface and gives practical tips to help you raise your Scrum teams to the next level. With humor, trivia, and stories from personal experience, *Scrum Shortcuts without Cutting Corners* is an accessible and adaptable Scrum recipe book ScrumMasters can use in any environment. Whether you have a few months or many years of experience with Scrum, this book will give you new ideas on how to approach whatever challenges are facing your team."

—Liza Wood, ScrumMaster and blogger at *Sockets and Lightbulbs*

Scrum Shortcuts without Cutting Corners

Scrum Shortcuts without Cutting Corners

Agile Tactics, Tools, & Tips

Ilan Goldstein

Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Library of Congress Cataloging-in-Publication Data

Goldstein, Ilan.
Scrum shortcuts without cutting corners : agile tactics, tools & tips / Ilan Goldstein.
pages cm
Includes bibliographical references and index.
ISBN 978-0-321-82236-9 (paperback : alkaline paper)
1. Agile software development—Handbooks, manuals, etc. 2. Scrum (Computer software development)--Handbooks, manuals, etc. I. Title.
QA76.76.D47G645 2013
005.1—dc23 2013016569

ISBN-13: 978-0-321-82236-9
ISBN-10: 0-321-82236-6
Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.
First printing, July 2013

To my little Amy, the cutest impediment one could ever hope for,
and to my soul mate Carmen, the greatest ScrumMaster of all!

This page intentionally left blank

CONTENTS

Foreword

In the spirit of this book, I'll take a shortcut and come right to the point: Buy this book. I assure you, you'll find the wisdom in this collection of shortcuts extremely helpful.

However, experience tells us to be leery of shortcuts. Very few work out. Horror movies begin when a group of teenagers take a shortcut through the woods on a dark night. The driver on a family trip opts for what looks like a shortcut and is reminded for years about how it turned out not to be. We're told "there are no shortcuts to success," and that success comes from a combination of perseverance and skill.

Yes, in life many shortcuts do not work out. The shortcuts in this book are different. They work.

I first met Ilan Goldstein online when a web search led me to his blog of Scrum shortcuts. He hadn't written many shortcuts by then, but the few he had were tremendously helpful—and funny. Ilan's sense of humor shone through in every shortcut.

It didn't take a genius to see that Ilan was onto something with his shortcuts. And so I asked him if he'd consider writing a book of shortcuts. This book is the result. In it, Ilan offers thirty tips, covering the full gamut of a Scrum implementation. He offers tips on getting started, on requirements, on estimating and planning. There are tips about quality and metrics, about team members and roles, about managing bosses, and about continuous improvement. If you've struggled with it on a Scrum project, it's likely Ilan has a shortcut to help you.

Ilan has been there and done that. His tips come from his experience as a ScrumMaster and Certified Scrum Trainer. His shortcuts all come from routes he's traveled. They're practical, not theoretical. Further, I like that he's not afraid to take a stand. Too many books rely too often on the consultant's standard answer of, "It depends." You won't find that here.

Whether you are a month, a year, or a decade into Scrum, you will find shortcuts here that will help you improve. I wish you well on your Scrum journey. I know you'll arrive more quickly by following the shortcuts in this book.

—Mike Cohn
Co-founder of the Scrum Alliance and the Agile Alliance
Author of *Succeeding with Agile*

This page intentionally left blank

Preface

"Ah, so it's the opposite of Dilbert" was my psychiatrist friend's reply when I gave him the quick overview of Scrum. (And no, I wasn't seeing him to deal with the stress of writing my first book while my first baby kept me up all night!) Anyway, after a chuckle, I realized that not only had my friend distilled Scrum so simply and elegantly, but also, I had just found my opening quote!

Scrum and its agile cousins comprise the next serious evolution in vocational process and culture. I'm certainly not alone in observing that this is possibly the greatest leap forward since the advent of scientific management, aka Taylorism. (By the way, did you know that a certain Henry Gantt of painful stripy-chart fame was a disciple of Taylor?) Scrum throws away the dictatorial, power-is-cool, ego-driven management approach that views people as replaceable cogs in a defined-process machine. Instead, Scrum treats teams as groups of responsible, dedicated free-thinkers who, given the opportunity, will work in an optimal fashion to derive the most positive outcome.

It is tremendously exciting and a privilege to be in the vanguard of this change together with our early Scrum pioneers who are still energetically leading the charge. No doubt in decades to come, this period in time will be recognized as an era when a significant shift occurred in the way the workplace operates.

Why Did I Write This Book?

I recall a conversation that I had with Martin Kearns, another Australian-based Scrum trainer, who pointed out to me that like it or not, people are going to read this book (with its assortment of tactics, tools, and tips) and consider it to be an official user manual—something that they should follow to the letter. This highlighted a concern that I was already feeling: how could I offer specific, focused advice that cuts through the theory and straight to the chase without making it seem too prescriptive? The answer to that question is to explain that *Scrum Shortcuts without Cutting Corners* is about sharing with you *an* approach rather than *the* approach to implementing Scrum. How can you have more than one approach to Scrum? you might be wondering. This question is explained clearly by Kenneth Rubin in his book, *Essential Scrum* (2012):

> Scrum is based on a small set of core values, principles, and practices (collectively the Scrum framework). Organizations using Scrum should embrace the

> Scrum framework in its entirety; however, this doesn't mean that each organization's Scrum implementation will be the same. Rather, each organization will have its own unique implementation of the Scrum framework based on the specific approaches that it chooses to realize the Scrum practices. . . . An approach is a particular implementation of a Scrum practice.

There are many other approaches, with their own sets of tactics, tools, and tips that you can and should explore, but I hope that the ones that I lay out in this book will, at the very least, trigger some thought and offer you some tried and tested options.

I wrote this book because, along my journey, I have acquired a significant number of cuts and bruises from tripping over stumbling blocks and banging my head against brick walls—frankly, implementing Scrum is really tough! It makes a whole lot of sense when the theory is explained to you, but gee whiz, try to get it up and running effectively and it is anything but trivial. Over a number of years, and after working with several teams, I finally began to see some return on investment from the bodily harm that had been inflicted. I had created an adaptable (and emergent) Scrum recipe book that worked across numerous different teams and several organizations, and I realized my hard-earned knowledge could help out others working in similar environments.

Back to my chat with Martin, as he also offered me some helpful advice: noting that I had just become a new parent for the first time, he asked me whether I thought I should try to protect little Amy from every situation in which she might fall and hurt herself. My heart said, "Absolutely, I won't let anything hurt my little girl," but my brain realized that you have to let even those you care about trip over (on occasion) to learn what works and what doesn't. That being said, though, you certainly always want to be there to comfort them and to give them some helpful advice for next time. So, this book is just that—the helpful advice for "next time" to start limiting the extent of your cuts and bruises moving forward. I'm assuming that you have at least given Scrum a shot, so you are likely already carrying some old wounds, but with any luck, this book will protect you somewhat from the next round of knocks.

If, however, you are new to Scrum and really hate cuts and bruises, feel free to jump right in. Perhaps some of this advice will keep you injury-free . . . for a while. But bear this in mind: every project is different, every team is different, and every organization is different. So if you're expecting to successfully apply every piece of advice on every page, then I would like to realign your expectations right now before you are disappointed—honestly, there is no magic approach that will work across the board.

Finally, for those of you who feel you've pretty much got this whole Scrum thing all worked out and under control, I hope that by browsing this book you are able to find some interesting new tools to add to your Scrum toolbox.

Some Assumptions

Most of the lessons that I share throughout this book were obtained while I was a hands-on ScrumMaster in several organizations, and as such, the book's primary audience is the ScrumMaster. However, that certainly doesn't exclude others from benefiting, including product owners, developers, and senior stakeholders. Even my attorney wife, without an ounce of interest in the software scene, found it useful and interesting—so there you go!

I also assume that you are not brand new to Scrum. I expect that you have read a few books, perhaps attended some entry-level training, and even tried working with Scrum for a period of time. If you fall into this camp, this book is intended to help you reach the next level of Scrum efficiency and maturity by expanding and extending your Scrum toolbox.

If you are brand new to Scrum, never fear: this book contains many chapters (or *shortcuts*) that you will still be able to easily relate to. However, I recommend that you at least read one or all of the following short Scrum overviews:

- *Core Scrum* (Scrum Alliance, 2012)
- *The Scrum Guide* (Schwaber and Sutherland, 2011)
- *The Scrum Primer* (Deemer, Benefield, Larman, and Vodde 2010)

For a more comprehensive introduction to Scrum, I highly recommend you pick up Rubin's recently published book, *Essential Scrum* (Rubin 2012).

How to Use This Book

I didn't write this book sequentially, so don't feel obliged to read it in that manner. Although the book is broken up into logical sections, you can easily jump around to your heart's content without losing continuity.

The shortcuts are written to be quickly and easily absorbed. My goal is to ensure that they are so easy to digest that even in the heat of battle, they can come to your aid. Alternatively, during peaceful times, they can act as some useful yet entertaining reading while you wait in line for the office microwave at lunchtime.

Speaking of lunchtime, you can treat *Scrum Shortcuts without Cutting Corners* like a recipe book (or a spell book if you just so happen to be a wizard or witch)—simply flip to the shortcut you're after, decide whether the ingredients work for you, and if not, feel free to add your own spices . . . at your own risk. With any luck, out of the oven will materialize an immediately useful and highly practical approach to tackling a particular Scrum challenge.

My Goals

This book is not just about helping make Scrum work for you. It is also meant to help you elevate your Scrum teams to the next level of effectiveness and maturity. Most of what I've written is not covered in any Scrum guide (nor even in your typical Scrum-Master training course). Instead, these are real-world approaches to the Scrum practices that have been properly tested under fire.

A point worth reiterating is that I don't expect you to follow what is written in this book to the absolute letter. However, I do recommend that during your constant quest for continuous improvement, you at least experiment by inspecting a selection of these tactics, tools, and tips and adapting your own processes to see whether they lead to improvements. Ideally, you not only will benefit from my approach but will be able to further evolve it and teach me a thing or two!

Acknowledgments

I can humbly admit that I completely under-estimated the efforts required to turn my nebulous thoughts into the published work that you're reading right now. Looking back now, I still can't believe how it all eventually came together! Writing a book requires extreme focus, follow-through, and open-mindedness but most importantly it requires help. This help comes in many shapes and forms and without it, there would be no book. There are many people who went above and beyond to help make this book what it is today and I am extremely grateful to each and every one of them who gave me a hand along the way.

Let's start from the very beginning of the journey so that I can firstly thank my main man, Colin Tan—business partner, art designer, in-the-trenches proofreader and best pal. He was the one who prodded me in the first place to start sharing my Scrum-thoughts with the world and without him you wouldn't be reading this book—simple as that. I'm sure that you'll agree that his artwork adds a unique, creative dimension to this book that the words alone cannot deliver.

I would next like to deeply thank Superman, aka Mike Cohn. I call him Superman for a couple of reasons. Now, everyone knows how much of a huge positive impact Mike has made on the Scrum world through his seminal books and community involvement, but not many of you know that he is also a power-lifting champion who can bench-press 560 pounds! I kid you not. Superpowers aside, being invited to write a book for Mike's series is a career high-watermark that is going to be ridiculously hard to top.

To Chris Guzikowski, Olivia Basegio, and Chris Zahn at Pearson, who helped me traverse the unfamiliar world of publishing and who patiently answered all of my questions throughout the process—I thank you for everything. Also, I'd like to thank Carol Lallier and Elizabeth Ryan who did such a great job pulling everything together at the end.

I'd like to thank the other Signature Series authors: Lyssa Adkins, Jurgen Appelo, Lisa Crispin, Janet Gregory, Clinton Keith, Roman Pichler, and Kenny Rubin not only for welcoming me into the team but also for providing me with inspiration long before I started writing this book. I'd especially like to thank Kenny Rubin for all of the valuable tips that he was able to offer me, having completed his own fantastic book, *Essential Scrum*, at the same time that I started writing this one.

Now to the reviewers: I will always be thankful for the time you took out of your busy lives to share your opinions and offer your thoughts. First I'd like to thank Cecil Goldstein, aka Dad: I bet you thought that your homework-proofing days were long over, but nevertheless, you happily came out of retirement, and your feedback was as valuable as ever.

I would also like to extend my gratitude to the following people who offered such helpful feedback throughout the long journey: Kevin Austin, Joel Bancroft-Connors, Jeremie Benazra, Charles Bradley, Mario Cueva, Pete Deemer, Ryan Dorrell, Caroline Gordon, Doug Jacobs, Ron Jeffries, Martin Kearns, Joy Kelsey, Richard Kaupins, Arik Kogan, Venkatesh Krishnamurthy, John Madden, Kane Mar, Jens Meydam, Nicholas Muldoon, Bryan O'Donovan, Michael Rembach, Matt Roadnight, Peter Saddington, Lisa Shoop, Craig Smith, Hubert Smits, Michael Stange, Renee Troughton, and Jason Yip.

To the Scrum and broader agile community, I thank you collectively for maintaining such an amazing culture of openness and for always being so willing to share ideas. I truly hope that this book will further contribute to the hugely positive influence that this community has already made in transforming the world of work.

Finally, to my amazing wife, Carmen, who did everything possible to give me every spare Pomodoro of time to write in the throes of all sorts of chaos, including precariously navigating through our first round of parenthood! You really are a Scrummy Mummy! And to little Amy, for allowing me just enough sleep to write at least semi-coherently, but mostly, for just being the most amazing inspiration and making everything worthwhile.

About the Author

Ilan Goldstein is an avid agilist with over a decade of practical hands-on experience. He is a globally recognized Certified Scrum Trainer (CST) working with start-ups, market leaders, government agencies, and public companies around the world, helping them to improve their agility through the implementation of Scrum. He is a regular conference speaker, guest university lecturer, and founder of both AxisAgile—a leading provider of agile training and consulting services—and Scrum Australia—a national not-for-profit organization focused on growing and enriching the Scrum community Down Under.

Ilan is a dedicated Scrum practitioner, relishing the time that he has spent as a ScrumMaster, developer, and product owner (not at the same time of course!) on a number of projects within a variety of environments and industries. He is acutely aware of what it takes to transform theory into practice and this knowledge is shared both within this book and the Scrum training that he conducts around the world.

Ilan holds a number of professional certifications that augment the war wounds that he has accumulated achieving his numerous in-the-trenches Scrum victories. These include: Certified Scrum Trainer (CST), Certified Scrum Professional (CSP), Certified ScrumMaster (CSM), Certified Scrum Product Owner (CSPO), Project Management Professional (PMP), as well as Agile Certified Practitioner (PMI-ACP).

He lives in Sydney, Australia, with his wife Carmen and daughter Amy and in his spare time volunteers for Compeer, the award-winning, global mental health program.

For more information, visit www.axisagile.com. Ilan can also be found on Twitter as @ilagile and can be contacted via e-mail at ilan@axisagile.com.

This page intentionally left blank

Chapter 3
Planning and Protecting

Now that your organization is eager to adopt Scrum and a team has been selected with the right attitudes and abilities, it is time to snap into action and get the show on the road.

The following three shortcuts not only help you to set the team on their course but also give you some tips and tricks to keep the project on track.

Shortcut 7: Setting the Scrum Stage lays down a range of suggestions to ensure proper foundations have been established to support a successful Scrum team. Shortcut 8: Plan the Sprint, Sprint the Plan provides specific, practical advice to ensure an effective sprint planning session. Finally, Shortcut 9: Incriminating Impediments offers advice to help control the impact of impediments during sprint execution.

Shortcut 7: Setting the Scrum Stage

Scrum teams require chemistry, and just as in a science lab, successful "chemical reactions" are much easier to trigger when the broader organization provides the suitable ingredients and environment to work with. As Mike Cohn (2009) astutely recognizes, "The changes required to reap all of the rewards being agile can bring are far reaching. These changes demand a great deal from not only the developers but the rest of the organization as well."

Let's examine some of the key organizational and environmental preconditions that should ideally be considered as part of your Scrum adoption plan.

Ensure Team Stability

Tom DeMarco and Timothy Lister (1999) identify a key mantra that any organization seeking close-knit teams should adopt: "Preserve and protect successful teams."

I am comfortable admitting that I have worked on Scrum projects that I would consider to be less than successful. I can easily pinpoint the core reason for these subpar results: my inability as a ScrumMaster to keep the team together for the duration of the project. This problem often occurs when key developers are dragged off one project to work on a more urgent project (see Figure 3.1). Couple this problem with continual corporate restructuring (that seems to be happening more and more in these times of global financial difficulty), and it can be challenging to preserve great teams.

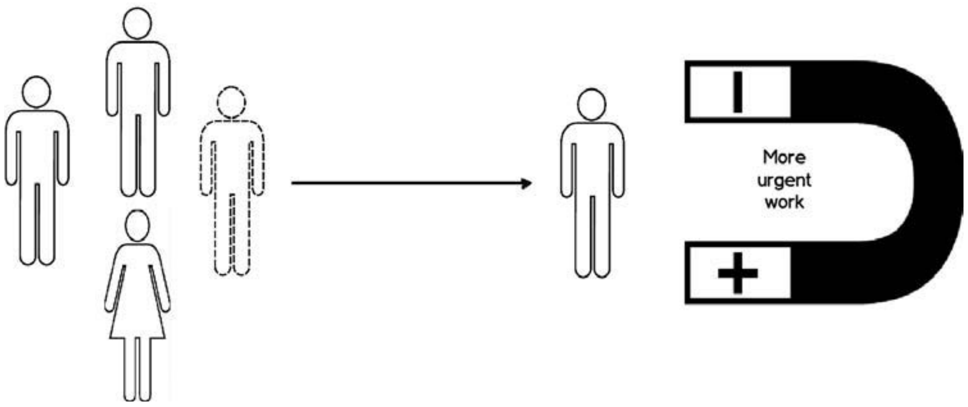

FIGURE 3.1 Beware of "more urgent" projects trying to drag team members away.

DeMarco and Lister (1999) also quantified the damage caused when rotating staff, concluding that "a reasonable assessment of startup cost (for a new team member) is therefore approximately three lost work-months per new hire." This estimate doesn't even take into account the less tangible costs such as lost momentum, damage to morale, and loss of valuable, tacit knowledge.

Adjust the Physical Environment

Without question, some of my most successful Scrum projects were those in which I was able to physically separate the Scrum teams from the rest of the organization.

DeMarco and Lister (1999) surmise why this may be the case:

> It almost always makes sense to move a project . . . out of corporate space. Work conducted in ad hoc space has got more energy and a higher success rate. People suffer less from noise and interruption and frustration.

I've worked with Scrum teams that had to operate in large, open spaces near the sales team who were frantically on their phones all day, every day. I've also had teams that had been hamstrung by the corporate facilities department who wouldn't allow them to move a small, measly round meeting table into their areas. I could go on and on, but the bottom line is that separation and environmental independence is the holy grail.

Irrespective of whether you are able to reach this lofty goal, you should do everything in your power to ensure that the Scrum team sits together. Scrum can certainly work for distributed teams where collocation isn't possible, but it's not optimal.

Apart from the obvious daily logistical benefits that sitting in close proximity offers, James Shore and Shane Warden (2007) offer an even more important rationale for the collocation of the team: “Sitting together is the most effective way I know to build empathy. Each group member gets to see that the others are working just as hard.” Shortcut 3 goes into more detail regarding other key inclusions to incorporate into the physical working environment to ensure a physical space conducive to Scrum.

Estimates Are Not Guarantees

How is this for an infuriating scenario? The project sponsor casually strolls over to a team member and asks how long feature XYZ is going to take. The team member takes off her headphones, breaking focus from what she was working on, glances up and throws out a rough estimate to appease the sponsor. Lo and behold, the estimate proves to be inaccurate. The sponsor then applies immense pressure on the entire team to deliver on the promised “commitment,” and dammit, if that means missing your kid’s end-of-year concert, then that’s the price of sticking to commitments!

News flash: An estimate is not a guarantee. If it were, there would be no need for the word. An estimate is simply a prediction based on known information and input at a given point in time. This definition needs to be clearly understood by the project stakeholders before the project kicks off!

Work toward Reciprocity

Mary and Tom Poppendieck, authors of *Leading Lean Software Development*, put forward the notion that there are two kinds of companies in this world: remuneration companies and reciprocation companies:

> People who work in a remuneration company have this agreement with their company: “I will show up for work and you will pay me for my time. If you want more than that, pay me more.” On the other hand, people who work in a reciprocity company have this agreement: “I will treat you the way you treat me. I expect fair compensation, but if you want care and commitment on my part, then you agree that you will demonstrate care and commitment toward me, and you will help me develop my potential to its fullest extent.” (Poppendieck and Poppendieck 2009)

The best Scrum teams consist of committed and caring individuals, so it naturally follows that companies that embrace the reciprocity model are more likely than remuneration companies to have greater success with Scrum, especially in the long term.

Support Sustainable Development

Shortcut 1 mentioned that one of Scrum’s guiding principles is that team members should work at a sustainable pace.

In *Agile Product Management with Scrum*, Roman Pichler (2010) points out that

> developing a product is like running a marathon. If you want to finish, you have to choose a steady pace. Many product owners make the mistake of pressuring the team to take on more work.

Any organization that maintains a culture of late-night martyrdom and continues to not only respect but also explicitly reward ludicrous overtime is at conflict with one of the principles of Scrum (or any other agile framework, for that matter). Overtime should be the exception, not the rule, and as recognized by Kent Beck in *Extreme Programming Explained* (1999), it should be recognized as "a symptom of a serious problem on the project," not simply business as usual.

Run a Pilot Project

Although there are certainly some advantages to taking the Big Bang approach to rolling out Scrum across an organization, I don't advocate it. Instead, I'm a big believer in initially running a pilot project. I recommend this approach even if the business is champing at the bit to roll Scrum out en masse. Why do I recommend investing this additional time if not to help obtain validation and buy-in? Mike Cohn (2009) explains the reason perfectly:

> [A] pilot project is undertaken to provide guidance to subsequent projects; it pilots the way in doing something new. . . . As an industry we have enough evidence that Scrum works; what individual organizations need to learn is how to make Scrum work inside their organizations.

I always run pilot projects before rolling out across a broader group, and in fact, without the experimental freedom that a pilot project offers, I'm not sure if you would even be reading this book today!

It may be tempting to select a project to pilot that is low value and therefore low risk. This is a false economy. Shore and Warden (2007) reinforce this point:

> Avoid taking a project with low value as a "learning opportunity." You'll have trouble involving customers and achieving an organizational success. Your organization could view the project as a failure even if it's a technical success.

Regarding team stability and the less-than-successful projects I experienced: the reason I couldn't keep those teams together was that the pilot projects we were working on were not of high enough priority and value. When push came to shove and shared resources were being stretched between the pilot project and the "more important" projects, no guesses as to which lost out.

How long should a pilot project last? Well, if you've been reading this section carefully, you will conclude that your pilot project should be no different from any other important project. As Roman Pichler (2010) states, "There is no rule in Scrum that mandates how long a project can last. But, it is common for agile projects to take no longer than three to six months."

Have Realistic Expectations

Change takes time, and very often with change, we need to take one step back to take two forward. Adopting Scrum requires a significant shift in organizational mindset that includes breaking entrenched habits, and this feat doesn't happen overnight.

There is going to be lead time before the developers feel comfortable working in cross-functional teams and before the old command-and-control attitudes disappear. Based on this premise, the organization should not be naïve and expect amazing gains immediately. Patience and nurturing is the name of the game, and a supportive organization will no doubt see its investment reaping dividends in the near future.

Shortcut 8: Plan the Sprint, Sprint the Plan

As one of the "elements of a chemistry-building strategy for healthy organizations," DeMarco and Lister (1999) recommend providing "lots of satisfying closure." I totally agree with their advice and suggest another complementary element: the provision of lots of clean, fresh starts. Luckily for us, the sprint time-box offers both closure and fresh starts, and in this shortcut we explore the Scrum activity that offers us the regular fresh start: *sprint planning*.

By collectively resetting the goals for the upcoming sprint every few weeks, the team can start afresh rather than remain stuck on a seemingly endless treadmill of ongoing work. Further, without this regular and expected planning session, significant disruption is caused when team members are rounded up on an ad hoc basis to plan and design.

Product Backlog Refinement

Before the team is gathered in the planning room, I recommend a few preliminary steps to ensure that the product backlog is refined appropriately. First, ensure that the product owner (with relevant assistance) not only has determined the next priority requirements for the upcoming sprint but also has fleshed them out in just enough detail to allow the developers to get started. Doing so might mean including more detailed acceptance criteria as well as any wireframes or mock-ups if applicable (see Shortcut 11). Additionally, it helps if the product owner has taken some time to collaborate in advance with any specialist testers to develop a set of initial test cases (based on the acceptance criteria) to fully describe the inner workings of the required functionality.

Goals Are Good

I would say that most of us enjoy working toward goals, so it is helpful to determine a centralized sprint goal that is omnipresent throughout the sprint. This focal goal typically maps to the main theme of the sprint. For example, the sprint goal might be *Enhance the Messaging Engine*; this doesn't mean that other bits and pieces can't also be worked on during the sprint, but it does indicate that the majority of the work will target the messaging engine. A sprint goal also helps with decision making by ensuring that everyone remains focused rather than deviating and heading off on tangents.

How Long Should a Sprint Be?

Back in the day, it was recommended that the sprint duration should be 30 days—no more, no less. These days, things have become somewhat more flexible, and it is now pretty much universally accepted that the sprint length can vary from team to team. If you speak to any Scrum team, you will find that the vast majority of sprints run from 1 to 4 weeks. I have tried them all out, and in my opinion, 1 week is too short, 4 weeks is too long, leaving me sitting on the fence between 2 and 3 weeks. For a new project, I make the decision based on two factors:

- **Team preference:** Some people prefer the longer duration to help gain more momentum, whereas others prefer the simpler planning that a shorter sprint offers.
- **Volatility of requirements:** If the product owner is likely to change requirements more often than not due to the product domain or market conditions, then I definitely recommend the shorter duration (see Figure 3.2).

Now, an important point: When the preferred sprint length is confirmed (it may take some experimenting in the early days), it should be locked in and rarely changed. There are specific reasons to avoid sporadically adjusting the sprint length, including the following:

- For team focus, a regular rhythm helps the team better understand how to pace itself.

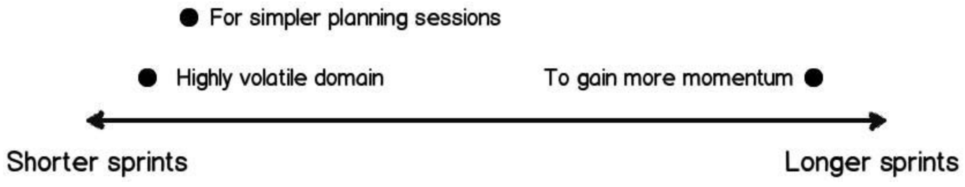

FIGURE 3.2 Factors to take into account when considering sprint length.

- The velocity metric (see Shortcut 13) relies on a consistent sprint duration; otherwise, it becomes less meaningful and more difficult to calculate.
- If you change the duration of sprints, your sprint review, retrospective, and planning sessions will not fall on the same day of the week. Such irregularities can prove to be a logistical headache, especially if you have to share meeting rooms with others in the organization.

Capacity Planning

Before diving into the sprint planning process, the team needs to first determine its sprint capacity. First, remember that not everyone will have full capacity for every sprint. Some team members may need to work across multiple projects—certainly not an ideal situation, but it can happen (see Shortcut 6). If this is the case, make sure that these developers aren't overallocated. Also, don't forget to take into account any public holidays, training, or scheduled leave.

Second, don't fall into the trap of believing that those who are dedicated full time to the sprint will be able to spend their entire working day on sprint-related tasks.

For example, in a team that I recently worked with (using 2-week sprints), a full-time developer was typically allocated a capacity of 9 days × 6 hours per day = 54 hours per sprint to work on tasks.

First, we used 9 days because the equivalent of 1 full day was dedicated to the sprint planning, review, and retrospective sessions. Six hours a day was allocated because in a typical 8-hour day we found that an individual would usually get only about 6 hours of solid sprint-focused work. The rest of the time was often taken up by various other activities unrelated to the current sprint, such as refining the product backlog (in anticipation for the upcoming sprint) and more general tasks required to be a good citizen of the organization (such as responding to email and assisting others not in the Scrum team). Please note that the proposed capacity per day will vary depending on the team and environment. As such, 6 hours a day should not be considered a universal standard, and I recommend using historical sprint interference statistics (see Shortcut 19) to help you determine your team's estimated sprint capacity.

Let's now take a look at the flow of the actual sprint planning session, which I like to split into two distinct parts:

Part 1: The What

This segment is all about the product owner presenting the next-highest-priority product backlog items (PBIs) to the development team as well as fielding any specific questions. This task is conducted for each of the PBIs that are being targeted for completion in the upcoming sprint. I recommend using the team's velocity (see Shortcut 13) as a rough guide to determine how many PBIs the product owner should be prepared to run through during this session.

Part 2: The How

Moving on, it is then time for the development team to break the PBIs into more granular technical tasks and to estimate each task to the nearest hour. Although estimating in hours may still be inaccurate at times, it helps the team make more informed design trade-off decisions and assists in establishing more confidence in what will likely be delivered by the end of the sprint. As Cohn (2007) explains further:

> The goal is not the hours but the hours are often a good tool to use to ensure we have discussed things (mostly the technical and product design of those things) at a level sufficient to enter the sprint with a good feeling that we'll be able to finish all the work of the sprint.

I don't expect product owners to hang around for this second part (unless they particularly want to). That being said, I stipulate that although product owners don't necessarily need to be in the room, they certainly need to be on call in case any further clarification is required. Nothing stalls a sprint planning session more than an unavailable product owner!

Even though I like to use velocity-based planning as a guide for Part 1, I like to also use what is commonly known as *commitment-based planning* to determine the number of specific tasks to include in the sprint backlog. Here are the steps that the development team typically runs through during commitment-based planning:

1. Start with the highest-priority PBI.
2. Deconstruct the PBI into tasks with estimates in hours.
3. Identify any specific task dependencies.
4. Continue this cycle until the team's collective capacity is full.
5. If the output from the velocity-based approach (from Part 1) doesn't match the output from the commitment-based approach, simply call back the product owner (if she has left the room) to add additional PBIs (if there is still capacity) or explain to her why there will be fewer PBIs targeted than initially expected (if the capacity is filled earlier than initially expected).

Task Definition

I typically set a few parameters for the generation of tasks:

- Each task needs to be a small, testable slice of the overall PBI (see Shortcut 10) and needs to factor in all activities required to meet the task's definition of "done" (see Shortcut 11).
- Each task should take no longer than about 8 hours (the shorter, the better, though).

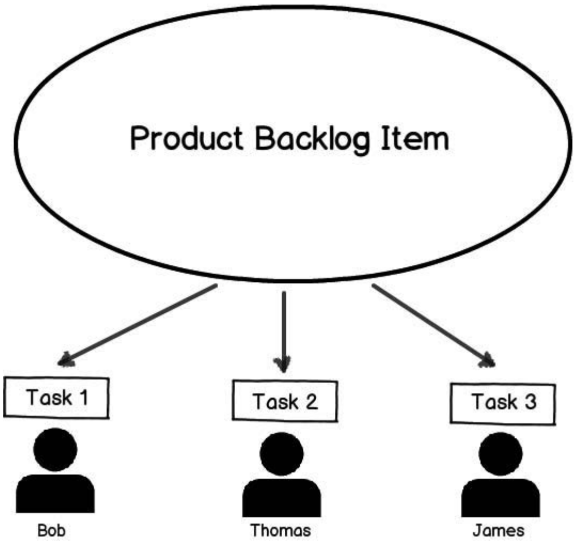

FIGURE 3.3 Although a single PBI can be worked on by multiple developers, each task should be worked on by only one developer.

- Although more than one developer can work on a single PBI, there should be only one developer (or developer pair) working on a task (see Figure 3.3).
- Don't forget to include tasks for the sprint review preparation (see Shortcut 22), such as preparing demo data if required.

You now have the sprint backlog compiled, including the tasks and corresponding estimates for them. The original estimates for the tasks can be aggregated to form the sprint's initial remaining time, and this time can then be recalculated each day and tracked on the sprint burndown chart (see Shortcut 19). Before going home each day, everyone on the development team should adjust the remaining time for any tasks they had been working on that day to ensure that up-to-date data is being fed into the sprint burndown chart.

The Right Number of Requirements

In a perfect world, after sprint planning is all said and done, you will have a nice neat whole number of PBIs that the team anticipates it will be able to complete in the forthcoming sprint. What you will find occasionally is that there will be a small amount of expected capacity left over that isn't quite enough to fit a whole new PBI into. That's okay—simply acknowledge as a team that the intention is to commence the next-highest-priority requirement without setting the expectation that it will be completed by the end of the sprint. I prefer this approach to trying to identify a small enough PBI (lower down on the product backlog) that could fit in nicely because I feel that it is more important to focus on working on the highest business value items.

The 7 Ps

As the British army adage goes, “Proper planning and preparation prevents piss-poor performance,” so a thorough and well-conducted sprint planning session is important to help generate a forecast that is as accurate as possible.

The sprint won’t always go according to plan, and no doubt adjustments will need to be made at times. However, if this session is well run, everyone will have a much better idea of what the collective objectives are, and this information will make the coordination and alignment of expectations a great deal easier.

Shortcut 9: Incriminating Impediments

You’ve trained your new Scrum troops, and they’re ready for their first mission. The team is pumped, and the project is up and running. Things are going well; the daily scrums are happening, the continuous integration server is humming along, tasks are moving across the board, so life is pretty sweet. Then, from out of nowhere, the bullets start flying, the mines start exploding, and your troops are no longer moving forward. This is it, ScrumMaster—time to step up!

Okay, so perhaps in reality, it isn’t a spray of enemy fire impeding your team. Instead, it might be a constantly breaking build, an interfering project sponsor, or perhaps the loss of a key team member. The bottom line is that anything impeding your team’s progress becomes the number-one priority for the ScrumMaster to tackle.

Defining Impediments

Let’s start by defining what an *impediment* is. Here is the definition I choose to use:

> An event that impedes any of the developers from working to their anticipated sprint capacity.

If you recall from Shortcut 8, it isn’t wise to allocate a full-time developer a sprint capacity of 8 hours a day (for a typical 8-hour working day). *Why not?* you ask. Well, you have to be realistic: there is no way that people are going to spend every second of their operational time working on their sprint tasks. We must take into account the various meetings that will pop up, other extended collaboration time, unplanned company events, and important head-clearing breaks, just to name a few. Now don’t get me wrong: on some days, some team members will be able to maintain strong focus on their sprint tasks and will max out or even exceed the 8 hours. However, on other days, constant interruptions may make it difficult to maintain even a couple of hours of sprint-focused work.

Many Shapes and Sizes

Impediments come in all shapes and sizes. Following is a small sample of indicative impediments (both operational and systemic) to keep a careful eye on:

- **Meetings of large magnitude:** Scrum projects really should have no need for these extraneous, long-winded bad boys, so when they pop up, they are typically triggered by other areas of the business or by unforeseen issues.
- **Illness:** Illness can strike unannounced at any time. Not much you can do about it, but I highly recommend you avoid a culture of "toughing it out" when sick. That expectation is just stupid. Work quality suffers, germs spread very quickly in an open Scrum environment, and it just annoys everyone.
- **Broken builds:** Without a healthy build (see Shortcut 18), development cannot continue. If a build is broken, it must be the top priority of every team member to fix it.
- **Issues with the tools of the trade:** Whether it is a hardware malfunction, software problem, or network connectivity issues, any problems with the working environment can seriously hamper progress and lead to immense frustration.
- **Unreliable supplier:** This is possibly one of the most frustrating impediments due to the lack of control that the ScrumMaster and team might have in dealing with an overburdened supplier. Poorly supported components or add-ons can lead to black holes that can seriously suck time from your sprints.
- **Unrefined product backlog:** A sprint should never start without the product owner knowing exactly what requirements should make their way into the sprint backlog. Further, these requirements should include enough detail for the development team get their teeth into. If these requirements are not ready for the sprint planning session then the sprint will not start smoothly at all (see Shortcut 11).
- **Absent or unempowered product owner holding up key decisions:** Product owners should be available throughout the sprint to field specific questions about the sprint backlog. If they are regularly absent or constantly having to seek approval from elsewhere, the development team might find itself paralyzed with uncertainty.
- **Incentive schemes focused on the individual:** Many organizations maintain performance reviews (and associated incentive schemes) that are based entirely on individual performance. Hopefully by now you've absorbed the

fact that there is no "I" in "Scrum team," so unless reviews also incorporate a significant focus on team collaboration, the organization will be sending a contradictory message to the team members.

Impediment ConTROL

I like to run through the following five-step approach when dealing with impediments: confirm, triage, remove, outline, learn (ConTROL).

- **Confirm:** Obviously it is necessary to confirm what the impediment is. Typically, impediments are raised in the daily scrum, but urgent impediments should be raised in real time rather than waiting for the daily scrum. The sprint retrospective can uncover further impediments that may have slipped through the cracks during the actual sprint. All impediments should be tracked and monitored until they are resolved.
- **Triage:** If you are bombarded with simultaneous impediments, then unless you are Superman (being obsessed with super heroes doesn't automatically give you their special powers), you will be able to tackle only one or two at a time. An impact and urgency assessment may be needed to help you determine where to start.
- **Remove:** Ideally, the Scrum team can remove any impediment that gets thrown its way, but it isn't always the reality. To avoid delays, it is important to know when to seek further help from other groups to get things back on track (see Shortcut 26).
- **Outline:** Both the Scrum team and stakeholders should be made aware of any impediments as they come up. You especially want to avoid surprises for product owners (and project sponsors) when it comes to any scuttled plans so that they have as much time as possible to iron out any cascade effects.
- **Learn:** The sprint retrospective (see Shortcut 23) is the main session during which impediments are analyzed. It is important to learn from these issues to avoid their recurrence and/or capture how they were dealt with so that if they strike again, the impact is less pronounced.

Blocks versus Impediments

Many teams use the terms *block* and *impediment* interchangeably, but I like to differentiate between the two (see Figure 3.4). I do so to clearly identify an obstruction that has *stopped* progress on a particular task but hasn't necessarily slowed down overall progress (a block) versus an obstruction that is *slowing down* the team's sprint progress (an impediment).

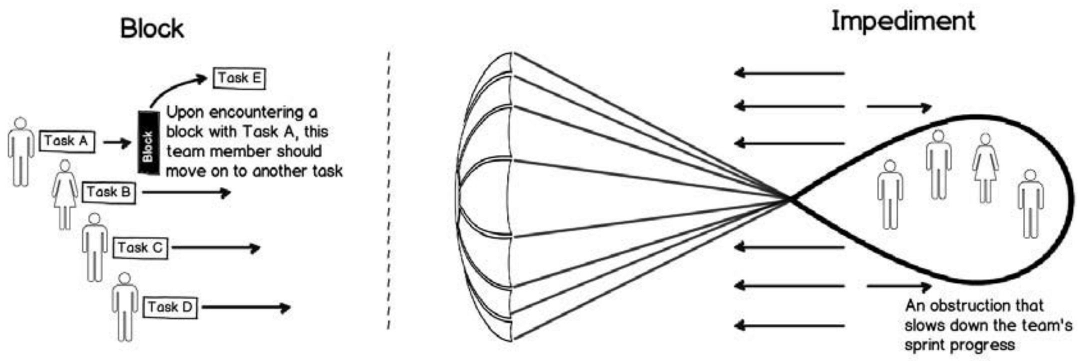
Block
Task E
Upon encountering a block with Task A, this team member should move on to another task
Task A
Block
Task B
Task C
Task D
Impediment
An obstruction that slows down the team's sprint progress

FIGURE 3.4 A block affects only a single task, whereas an impediment acts like a parachute, slowing down overall progress.

A typical block occurs when a task has a dependency that has been held up for some reason. A short, temporary block is a reasonably common occurrence and nothing to get too concerned about, because in most cases, other work can be taken on while the dependency is being taken care of. The important thing to note is that you want clear visibility of all blocked tasks, irrespective of how temporary the block may be. The way I like to track blocked tasks is to simply spin the corresponding sticky-note 45 degrees so that it looks like a diamond and stands out on the task board. This is a clear signal and allows you to immediately jump into detective mode to ensure that the block is removed as quickly as possible.

Understand the Terrain

If you're like me, you will be stunned when you think back to all of the impediments that occurred throughout a project. It is often only when you reflect on the compiled list of tracked impediments that you can really appreciate the difficult terrain your Scrum troops have had to negotiate.

This impediment list can also prove invaluable if ever you must defend the team should it fall under an uncomfortable spotlight due to incremental delays that have started to impact release dates. I certainly found that if you carefully ConTROL impediments when they decide to rear their ugly heads, these uncomfortable situations will be few and far between.

Wrap Up

The three shortcuts discussed in this chapter focused on a selection of tactics, tools, and tips to help you to set your team on course and keep them on track. Let's recap what was covered:

Shortcut 7: Setting the Scrum Stage

- The importance of collocating team members whenever possible
- A selection of cultural adjustments that are required to ensure that Scrum can thrive
- How and why pilot projects can assist in longer-term Scrum adoption

Shortcut 8: Plan the Sprint, Sprint the Plan

- Factors to consider when selecting your sprint length and goal
- How to determine a realistic sprint capacity
- Options for structuring your sprint planning session

Shortcut 9: Incriminating Impediments

- Definitions of impediments and blocks
- Types of impediments to watch out for
- How to ConTROL your list of impediments

INDEX