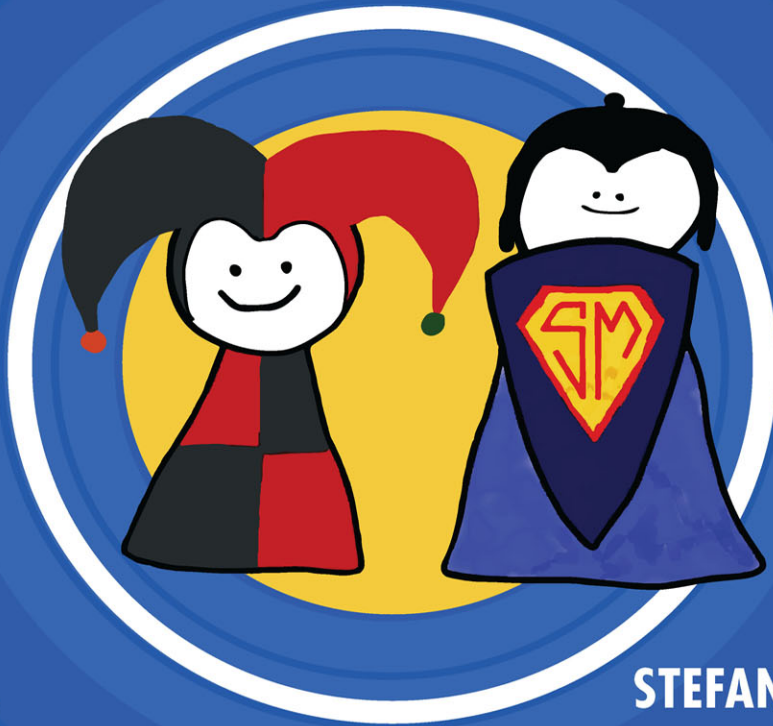


THE SCRUM ANTI-PATTERNS GUIDE


CHALLENGES EVERY SCRUM TEAM FACES
AND HOW TO OVERCOME THEM



STEFAN WOLPERS

Forewords by **DAVE WEST & JANNA BASTOW**



The Professional Scrum Series by  Scrum.org

FREE SAMPLE CHAPTER |



Stefan Wolpers has a remarkable ability to highlight underlying traps and issues for stakeholders, teams, and process. In *The Scrum Anti-Patterns Guide*, Wolpers documents sources of waste and frustration, an amazing compendium of typical ways progress becomes blocked. Depressing! He doesn't leave us there though. He also recommends insightful remedies. Uplifting!

—*Diana Larsen, speaker, advisor, author at dianalarsen.com*

This book is an invaluable treasure for every Scrum practitioner as it shows not only what can (and does) go wrong in Scrum teams, but also how to avoid and overcome these traps. Stefan shares his in-depth experience and provides concrete tips to help the whole Scrum team grow.

—*Jutta Eckstein, coauthor of Company-Wide Agility with Beyond Budgeting, Open Space & Sociocracy*

Stefan Wolpers' comprehensive collection of common Scrum anti-patterns is a great resource for anybody who wants to improve their application of Scrum and fully leverage the framework. Highly recommended!

—*Roman Pichler, author of Agile Product Management with Scrum*

There is so much potential with Scrum that sadly ends up wasted because teams misuse the process. Wolpers does a masterful job of identifying the anti-patterns that cause this waste and points out exactly how to overcome each one. This is a must-have book for all Scrum practitioners.

—*Jeff Gothelf, author of Lean UX and Sense & Respond*

The first step to fixing a problem is acknowledging you have a problem. Stefan Wolpers' wonderful book is the ultimate compendium of Scrum anti-patterns. After exposing common Scrum problems, the book serves as a cunning companion that masterfully guides you to transcend the dysfunctions it lays bare.

—*Maarten Dalmijn, author of Driving Value with Sprint Goals*

Stefan's groundbreaking book *The Scrum Anti-Patterns Guide* flips the script, spotlighting Scrum pitfalls over best practices. Packed with vivid examples, clear illustrations, and robust theory, this book accelerates learning by showing you what NOT to do. Not only insightful but delightfully entertaining!

—*Jorgen Hesselberg, author of Unlocking Agility: An Insider's Guide to Agile Enterprise Transformation and Co-Founder of Comparative Agility*

For years I've admired Stefan Wolpers' shared anti-patterns of various roles and activities within and around Scrum teams. His insights are simple, insightful, and powerful, while always rooted in his practical experience. He also artfully balances the anti-patterns away from criticism and toward improvement. This book contains his collective wisdom and I couldn't recommend it more to anyone desiring to improve their Scrum.

—*Bob Galen, Principal Coach at Agile Moose*

The Scrum Anti-Patterns Guide

The Professional Scrum Series by Scrum.org



Visit informit.com/scrumorg for a complete list of available publications.

The Professional Scrum Series from Pearson Addison-Wesley and Scrum.org consists of a series of books that focus on helping individuals and organizations apply Scrum and agile leadership to improve the outcomes of customers and organizations. Approaching the challenge from different perspectives, each book provides deep insights into overcoming the obstacles that both teams and organizations face as they seek to reap the benefits of agility.

All Scrum.org proceeds from the series go to Year Up, an organization whose mission is to close the Opportunity Divide by providing urban young adults with the skills, experience, and support to empower them to reach their potential through professional careers and education.

The Scrum Anti-Patterns Guide

**CHALLENGES EVERY SCRUM TEAM FACES AND
HOW TO OVERCOME THEM**

Stefan Wolpers

◆ Addison-Wesley

Cover image: Bakemon/Shutterstock

FIGB-07-Ralph Jochem

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Visit us on the Web: informit.com/aw

Library of Congress Control Number: 2023950677

Copyright © 2024 Pearson Education, Inc.

Hoboken, NJ

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions.

ISBN-13: 978-0-13-797796-3

ISBN-10: 0-13-797796-4

\$PrintCode

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where:

- Everyone has an equitable and lifelong opportunity to succeed through learning.
- Our educational products and services are inclusive and represent the rich diversity of learners.
- Our educational content accurately reflects the histories and experiences of the learners we serve.
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview).

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

- Please contact us with concerns about any potential bias at <https://www.pearson.com/report-bias.html>.

This page intentionally left blank

CONTENTS

Foreword by Dave West	xv
Foreword by Janna Bastow	xvii
Preface	xix
Acknowledgments	xxi
About the Author	xxiii
Introduction	xxv
Chapter I Scrum Master Anti-Patterns	I
Introduction	1
The Scrum Master According to the Scrum Guide	2
Possible Reasons Why Scrum Masters Leave the Path	2
Anti-Patterns from Acting as an Agile (Line) Manager	5
Scrum Master Anti-Patterns by Scrum Events	15
The Sprint Planning	15
The Sprint	17
The Daily Scrum	20
The Retrospective	22
Food for Thought	26
Conclusion	26

Chapter 2	Product Owner Anti-Patterns	29
	Introduction	29
	The Role of the Product Owner According to the Scrum Guide	30
	Product Backlog and Refinement Anti-Patterns	31
	Sprint Planning Anti-Patterns	40
	Sprint Anti-Patterns	42
	Product Owner Anti-Patterns during the Daily Scrum	46
	Sprint Review Anti-Patterns	50
	Food for Thought	52
	Conclusion	52
Chapter 3	Scrum Developer Anti-Patterns	55
	Introduction	55
	The Role of the Developers in Scrum	56
	Developer Anti-Patterns by Scrum Events	56
	Sprint Anti-Patterns	56
	Sprint Planning Anti-Patterns of the Developers	66
	Anti-Patterns during the Daily Scrum	68
	Developer Anti-Patterns Concerning the Sprint Review	75
	Sprint Retrospective Anti-Patterns of Developers	76
	Anti-Patterns at the Product Backlog Level	77
	Food for Thought	81
	Conclusion	81
Chapter 4	Scrum Stakeholder Anti-Patterns	83
	Introduction	83
	Common Scrum Stakeholder Anti-Patterns	84
	Scrum Stakeholder Anti-Patterns at the Organizational Level	84
	Sprint Anti-Patterns of the IT Management	96
	Incentivized Scrum Stakeholder Anti-Patterns	98
	Stakeholder Anti-Patterns at Scrum Event Level	104
	Product Backlog and Refinement Anti-Patterns	105
	The Daily Scrum	105
	Sprint Planning Anti-Patterns of Stakeholders	106
	The Sprint Review	106

	The Sprint Retrospective	107
	Food for Thought	108
	Conclusion	108
Chapter 5	Sprint Anti-Patterns	109
	Introduction	109
	The Purpose of the Sprint	109
	Sprint Anti-Patterns	111
	Sprint Anti-Patterns of the Product Owner	111
	Sprint Anti-Patterns of the Developers	112
	Sprint Anti-Patterns of the Scrum Master	113
	Sprint Anti-Patterns of the Scrum Team	113
	Sprint Anti-Patterns of the IT Management	118
	Sprint Review Anti-Patterns of Stakeholders	121
	Food for Thought	123
	Conclusion	124
Chapter 6	Sprint Planning Anti-Patterns	125
	Introduction	125
	Preparing the Sprint Planning	127
	Sprint Planning Anti-Patterns	127
	Sprint Planning Anti-Patterns of the Developers	127
	Sprint Planning Anti-Patterns of the Product Owner	132
	Sprint Planning Anti-Patterns of the Scrum Team	134
	Sprint Planning Anti-Patterns of the Scrum Master	138
	Food for Thought	139
	Conclusion	140
Chapter 7	Daily Scrum Anti-Patterns	141
	Introduction	141
	The Purpose of the Daily Scrum According to the Scrum Guide	142
	Daily Scrum Anti-Patterns	143
	Daily Scrum Anti-Patterns of the Scrum Team	143
	Daily Scrum Anti-Patterns of the Developers	146
	Daily Scrum Anti-Patterns of the Product Owner	150

	Daily Scrum Anti-Patterns of the Scrum Master	150
	Daily Scrum Anti-Patterns of the Stakeholders	151
	Food for Thought	154
	Conclusion	155
Chapter 8	Sprint Review Anti-Patterns	157
	Introduction	157
	The Scrum Guide on the Sprint Review	158
	Sprint Review Anti-Patterns	159
	Sprint Review Anti-Patterns of the Scrum Team	159
	Sprint Review Anti-Patterns of the Product Owner	162
	Sprint Review Anti-Patterns of the Developers	165
	Sprint Review Anti-Patterns of the Stakeholders	167
	Food for Thought	174
	Conclusion	175
Chapter 9	Sprint Retrospective Anti-Patterns	177
	Introduction	177
	The Scrum Guide on the Sprint Retrospective	178
	Sprint Retrospective Anti-Patterns	179
	Sprint Retrospective Anti-Patterns of the Scrum Team	179
	Sprint Retrospective Anti-Patterns of the Scrum Master	187
	Sprint Retrospective Anti-Patterns of the Organization	191
	Food for Thought	194
	Conclusion	195
Chapter 10	Product Backlog and Refinement Anti-Patterns	197
	Introduction	197
	The Product Backlog According to the Scrum Guide	198
	Common Product Backlog Anti-Patterns	200
	General Product Backlog Anti-Patterns	200
	Product Backlog Anti-Patterns of the Product Owner	207
	Product Backlog Anti-Patterns of the Developers	209
	Product Backlog Anti-Patterns of the Scrum Team	211
	Food for Thought	212
	Conclusion	213

Chapter 11	Sprint Backlog Anti-Patterns	215
	Introduction	215
	Sprint Backlog Anti-Patterns	216
	Sprint Backlog Anti-Patterns of the Scrum Team	217
	Sprint Backlog Anti-Patterns of the Developers	220
	Sprint Backlog Anti-Patterns of the Product Owner	225
	Food for Thought	228
	Conclusion	228
Chapter 12	Increment Anti-Patterns	229
	Introduction	229
	The Purpose of the Increment According to the Scrum Guide	230
	Increment Anti-Patterns	231
	Increment Anti-Patterns by the Scrum Team	231
	Increment Anti-Patterns of the Stakeholders or the Organization	240
	Food for Thought	242
	Conclusion	242
Chapter 13	Product Goal Anti-Patterns	245
	Introduction	245
	The Purpose of the Product Goal According to the Scrum Guide	246
	Product Goal Anti-Patterns	247
	Product Goal Anti-Patterns of the Product Owner	247
	Product Goal Anti-Patterns of the Scrum Team	251
	Product Goal Anti-Patterns of the Organization	254
	Food for Thought	257
	Conclusion	257
Chapter 14	Sprint Goal Anti-Patterns	259
	Introduction	259
	How to Create Sprint Goals	260
	Sprint Goal Anti-Patterns	261
	Sprint Goal Anti-Patterns of the Scrum Team	262
	Sprint Goal Anti-Patterns Induced by the Organization	267

	Sprint Goal Anti-Patterns by the Developers	269
	Sprint Goal Anti-Patterns by the Product Owner	271
	Food for Thought	272
	Conclusion	272
Chapter 15	Definition of Done Anti-Patterns	273
	Introduction	273
	Creating a Successful Definition of Done	275
	Definition of Done Anti-Patterns	277
	Definition of Done Anti-Patterns of the Developers	278
	Definition of Done Anti-Patterns of the Scrum Team	281
	Definition of Done Anti-Patterns of the Organization	289
	Definition of Done Anti-Patterns of the Product Owner	290
	Food for Thought	291
	Conclusion	291
Appendix A	How to Sabotage Scrum Masters and Product Owners at an Organizational Level	293
Appendix B	Toolbox	305
Index		369

FOREWORD BY DAVE WEST

Ken Schwaber, the co-creator of Scrum, often says, “Scrum takes minutes to learn but a lifetime to master.” This saying describes the paradox of Scrum: It is a simple framework that is easy to learn and understand, but practicing it well is challenging and takes time. Scrum is not complex, but the environment in which it is employed can be very complex. The complexity that Scrum works within includes the team, the organization, the problem domain, the solution domain, and the stakeholders. Each of these areas can manifest challenges to empiricism, self-management, and the ability to improve. Scrum Teams must overcome these challenges to achieve their full potential using Scrum.

As CEO at Scrum.org, I have seen first-hand the struggles of Scrum Teams. I have seen Product Backlogs that are merely lists of work to be done and Sprint Plans that were dictated to the team instead of being developed by the team to achieve a Sprint Goal. I have seen Product Increments that are intentionally not ready for users and Sprint Reviews in which Stakeholders simply look for progress against a plan instead of understanding the value that was delivered. The list goes on and on. The results in all these situations could have been better for everyone involved if they had understood the common dysfunctions of Scrum Teams and their organization and how to overcome them.

Many organizations look to consultants and practitioners experienced with Scrum to help them avoid these and many other dysfunctions. That’s nice if you can find

them; already having made the mistakes and learning from them can help prevent them in the future. In the absence of this personal experience, books, blogs, and articles written by experienced practitioners can help you understand where teams and their organizations can and often do go wrong and what to do about them.

In this book, Stefan describes the anti-patterns he has observed and provides simple and practical advice on their resolution. He has structured the book around the elements of Scrum to make it easy for you, the reader, to easily find anti-patterns. For example, if you are new to the Sprint Review or struggling with it, you can quickly dip into that chapter and see if anything resonates.

Each chapter has many pearls of wisdom, and I have experienced many examples in the organizations I have visited. Still, my favorite two are Scrum Sprint Planning and Sprint Review. Both these events are crucial to the success of Scrum but often need to be better executed, and improvements in these two events have a massive impact on the value of the Increment, team morale, and the relationship with Stakeholders.

In the agile community, we spend much time moaning about the lack of success, blaming the big things. Things include legacy systems, existing processes, HR practices, incentive programs, organization structures, job titles, and team structure. And, yes, many of those things make agile very difficult. But, at the heart of agile is the idea that teams break things down, deliver frequently, and do what they need to do. Agile is much more than teams; but it starts and finishes with teams. How they work, how they collaborate, and the choices they make collectively make them more or less agile. This book is all about those choices. And the first step in any choice is knowing you have a choice to make. A patterns approach is a great way to identify those issues and the choices you have to make.

The phrase “dream big, but start small” is an excellent guide to using Scrum, and I hope this book can provide you with some tips and a frame for changing the world, one choice at a time.

ScrumOn!

— Dave West
October 2023
Weston, MA

FOREWORD BY JANNA BASTOW

I once found myself amidst the bustling energy of a newly funded startup, right at the awkward juncture of scaling. We had a couple dozen developers who seemed to know their stuff but we weren't jiving as a team yet. Our boss brought in a Scrum Master, complete with a pile of certifications and textbooks and promises of a transformation. And indeed, we did see a transformation! I've never seen a group of four development teams whipped into shape so fast. Their velocity was off the charts... and yet they were delivering absolutely nothing of value. The disconnect between development processes and product needs was massive, so no impact was being made. It was a classic case of a Scrum anti-pattern—mistaking motion for progress.

I wish I'd read this book all those years ago, as Stefan Wolpers' guide would have saved us from a lot of wasted effort.

Over the years, I've worked with hundreds of companies, and I've seen Scrum fail in a spectacular kaleidoscope of ways. Stefan spots all of these pervasive anti-patterns and calls them out, guiding the reader on how to avoid them at every turn. This isn't just another book on Scrum; it's a lifeline for those of us who have found ourselves trapped in a quagmire of "doing Scrum right" but not reaping the benefits. At its heart, this book is all about impact. It's about making use of the tools and

processes that Scrum gives us to get the most out of it for your team, your product, and your business.

But Stefan doesn't stop at the basics. He delves deep, guiding you through the good, the bad, and the downright ugly metrics that will shape your Scrum journey. He tackles tough subjects around stakeholder management with good humor. And for those grappling with the realities of today's distributed teams, there's invaluable advice on establishing Communities of Practice and effective remote management.

One of the standout features that you'll love with this guide is the sheer practicality. It's the kind of book you'll want to keep within arm's reach on your desk, not just for yourself, but to share with your developers, your product owner, your Scrum Master, and anyone else baffled by the Scrum process. Stefan's humorous takes on how to (hypothetically) sabotage your product owner or Scrum Master are not only entertaining but serve as useful cautionary tales, highlighting what not to do (or what to stop doing!). And let's not forget the iconic comics sprinkled throughout the book. These illustrations, both witty and insightful, perfectly capture the essence of the situations discussed, making your journey to better Scrum more digestible.

In classic Stefan fashion, there's no beating around the bush. Every page, every chapter, is packed with value, devoid of unnecessary fluff. It's a testament to his expertise and his commitment to delivering actionable insights to the Scrum community.

In essence, if you're looking to make a tangible difference with your work, this book is a must-read. Dive in, apply its lessons, and see the transformation for yourself.

Janna Bastow,
Co-founder & CEO, ProdPad

PREFACE

The Scrum Anti-Patterns Guide offers a different perspective on how to apply Scrum successfully by applying the principle of “inversion.” Instead of explaining how to apply Scrum properly, the book looks at the many things that may go wrong when practicing Scrum—the anti-patterns.

The principle of inversion refers to a change in order or position, specifically, changing perspective to gain different insights. This principle is a powerful learning tool because it encourages viewing problems from an alternative standpoint, revealing new dimensions and possible solutions.

In the ideal case, inversion helps us confront the constraints, potential pitfalls, or worst-case scenarios upfront, which can illuminate the path to success by highlighting what to avoid.

Even under less-than-ideal circumstances, the principle of inversion can still provide immense value, even when you’re already making the mistakes you should avoid.

Here’s how:

Identifying root causes: Through inversion, you can better understand the root causes of the mistakes. Often, we focus on symptoms rather than causes. By asking,

“What actions or conditions led to these mistakes?” you can start to address the underlying issues.

Preventing recurrence: Once the mistakes are made, inversion can help prevent them from recurring. By analyzing what led to the error, you can implement safeguards and checkpoints to prevent the same mistakes in the future.

Learning and adaptation: Making mistakes is a part of the learning process. Inversion promotes learning from these mistakes and adapting your practices accordingly. The goal is not to avoid all mistakes but to continuously learn and improve.

Prioritization of actions: Not all mistakes have the same impact. Inversion can help identify the most damaging errors, effectively prioritizing countermeasures.

Promoting a proactive mindset: Rather than merely reacting to errors, inversion encourages a proactive mindset, fostering preventative actions and forward-thinking solutions.

Consequently, applying the inversion principle to Scrum can produce profound results as Scrum is grounded in feedback-informed learning and continuous improvement, concepts that align well with inversion.

The book started as a series of blog posts in 2017/2018 when I began curating all my observations over the years, particularly in larger organizations.

Register your copy of *The Scrum Anti-Patterns Guide* on the InformIT site for convenient access to updates and/or corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN (9780137977963) and click Submit. Look on the Registered Products tab for an Access Bonus Content link next to this product, and follow that link to access any available bonus materials. If you would like to be notified of exclusive offers on new editions and updates, please check the box to receive email from us.

ACKNOWLEDGMENTS

I want to thank Scrum.org for supporting this book, particularly David West and Kurt Bittner.

Another big thank you goes to Haze Humbert of Pearson for her inexhaustible patience.

Finally, I would like to thank Oliver Guillard for turning my sketches into useful graphics.

JOIN THE COMMUNITY

If you would like to talk to other readers of this book, consider joining the community. All you have to do is provide your credentials, and I will invite you.

You can join via this link: <https://bit.ly/sag-community>. Alternatively, use the following QR Code, see Figure 00.1.



Figure 00.1 Sign up for the Reader Community.

ABOUT THE AUTHOR

Stefan Wolpers is a Professional Scrum Trainer at Scrum.org, Agile Coach, and Scrum Master, specializing in guiding agile transformations through practices like Scrum, LeSS, Kanban, Lean Startup, and professional product management. He's a licensed Agile Fluency™ Team Diagnostic facilitator with a history of senior leadership roles.

His agile coaching focuses on scaling product delivery for fast-growing, venture-capital-backed startups as well as transitioning existing teams in established enterprises.

Stefan curates the widely followed “Food for Agile Thought” newsletter, engaging 50,000+ global Agile enthusiasts. He shares insights on Age-of-Product.com, hosts a vibrant Slack community of 18,000+ agile practitioners, and has authored e-books on agile themes, garnering over 100,000 downloads.

This page intentionally left blank

INTRODUCTION

PURPOSE OF THIS BOOK

The primary purpose of *The Scrum Anti-Patterns Guide: Challenges Every Scrum Team Faces and How to Overcome Them* is to help Scrum practitioners identify, understand, and address commonly observed patterns of behavior that undermine the effectiveness of Scrum in achieving its goals.

The book addresses these challenges—from the misaligned understanding of roles and practices to organizational issues—highlighting these anti-patterns and pointing to practical solutions. The guide seeks to enhance the application of Scrum, making it more effective in driving innovation, productivity, and overall value delivery.

In real life, stakeholders are not usually interested in how we solve their problems as long as we ethically play by the rules of our society in alignment with our organization's culture. Instead, they are interested in the regular delivery of valuable Increments.

In other words, we do not get paid to practice Scrum but to solve our customers' problems within the given constraints while contributing to the sustainability of our organization. Unfortunately, this typical attitude also implies that no matter how

well versed your Scrum Team is in practice, no one outside your team will care unless your team regularly delivers valuable Increments in every Sprint.

So, Scrum can contribute to solving your customers’ problems if you choose Scrum in the proper context: solving complex adaptive problems. Notably, Scrum can also be a complete waste of everyone’s time when selected for the wrong reasons: If you know exactly what to build, there is no need to employ Scrum.¹

As if choosing the proper context for employing Scrum wasn’t challenging enough, Jeff Sutherland and Ken Schwaber designed Scrum’s framework as “purposefully incomplete.”² While this is true of any framework—otherwise we would call them methodologies—many practitioners interpret the phrase as an invitation to change and augment the framework in basically any way they see fit, delivering value being the justifying means; see Figure INT.1.



Figure INT.1 A “purposefully incomplete” framework will result in many different approaches, opinions, and practices on complementing Scrum best, often contradicting each other or disrespecting first principles.

-
1. Scrum excels at figuring out what is worth building while mitigating risk. Consequently, its iterative and incremental approach comes at a price; we plan a lot in Scrum, which is not free.
 2. Ken Schwaber and Jeff Sutherland, “Scrum,” in *The 2020 Scrum Guide*TM, 2020, <https://scrumguides.org/scrum-guide.html#scrum-definition>.

The Scrum Anti-Patterns Guide seeks to equip all practitioners with the insights and strategies necessary to transform these anti-patterns into learning, growth, and continuous improvement opportunities. The book comprises 15 chapters and two appendices:

Chapter 1: Scrum Master Anti-Patterns studies points such as the “agile” manager, tracking flawed or useless metrics, and defining technical solutions to problems likely stemming from misunderstandings, dogmatism, impatience, indifference, opportunism, frustration, or inexperienced practice.

Chapter 2: Product Owner Anti-Patterns dissects observations such as monopolizing Product Backlog management, poor communication with stakeholders and teammates, and overemphasizing administrative tasks, issues that critically limit the Product Owner’s potential to maximize the product’s value.

Chapter 3: Scrum Developer Anti-Patterns reminds us of Developers’ crucial contribution to making a Scrum Team successful. Nevertheless, anti-patterns such as corner-cutting to meet deadlines, side gigs, and delivering undone work undermine Developers’ effectiveness.

Chapter 4: Scrum Stakeholder Anti-Patterns points to the significant influence that problems at this level, such as a lack of transparency, inadequate leadership support, and cherry-picking individual practices, have on Scrum Teams, inhibiting the team’s progress toward its Product Goal.

Chapter 5: Sprint Anti-Patterns addresses challenges to providing a framework to deliver value and foster predictability. Anti-patterns such as variable Sprint length, undermining a Scrum Team’s autonomy, and bypassing the Product Owner negatively affect the result.

Chapter 6: Sprint Planning Anti-Patterns delves into familiar problems such as the lack of capacity checks, output focus, and imposed forecasts, which hinder Scrum Teams from aligning on delivering customer value.

Chapter 7: Daily Scrum Anti-Patterns reminds us of the event’s purpose: facilitating inspection, adaptation, and short-term planning toward the Sprint Goal. Anti-patterns such as treating it as a status report, assigning tasks directly to Developers, and stakeholder interference can inhibit the Daily Scrum’s effectiveness.

Chapter 8: Sprint Review Anti-Patterns stresses the importance of the Scrum event for assessing progress toward the Product Goal. Unfortunately, anti-patterns such as customers' absence, approval gates, and passive stakeholders can undermine this approach.

Chapter 9: Sprint Retrospective Anti-Patterns examines the crucial event for continuous Scrum Team improvement. Yet, anti-patterns such as rushed Retrospectives, unsuitable venues, and the absence of psychological safety can render its outcome questionable at best.

Chapter 10: Product Backlog and Refinement Anti-Patterns analyzes typical issues of many Scrum Teams, from an oversized Product Backlog, including outdated items, to excessive detailing, to a lack of regular refinement, impeding the core artifact crucial for Scrum Team success.

Chapter 11: Sprint Backlog Anti-Patterns points to the Sprint Backlog's temporary nature as an actionable repository and plan for achieving the Sprint Goal. Consequently, anti-patterns such as overly detailed planning, fixed scope, and changing work items mid-Sprint can undermine its effectiveness.

Chapter 12: Increment Anti-Patterns looks at Scrum's advocating for regular releases of Increments for swift user feedback to accelerate learning. Anti-patterns such as postponed releases, releasing exclusively to the complete user base, and Increments not meeting the Definition of Done can easily disrupt this crucial iterative process.

Chapter 13: Product Goal Anti-Patterns delves into the Product Goal's purpose of aligning strategy to tasks and promoting focus and transparency. Yet, anti-patterns, from imposed Product Goals to the lack of communication, can undermine these intentions.

Chapter 14: Sprint Goal Anti-Patterns analyzes how the Scrum Team is impeded from delivering outcomes valued by customers by unsuited Sprint Goals, overambitious plans, inconsistent achievements, and misplaced focus.

Chapter 15: Definition of Done Anti-Patterns underscores the Definition of Done's crucial role in Scrum, as it helps align the team on product quality standards, fosters continuous improvement, and creates value. Its absence or inadequacy, however, may expedite failure.

Appendix A: How to Sabotage Scrum Masters and Product Owners at an Organizational Level uses a reverse-thinking exercise to identify and tackle

potential challenges Scrum Masters and Product Owners might face in organizations.

Appendix B: Toolbox provides many useful tools for practitioners in Scrum, including interview questions to help newly appointed Scrum Masters get up to speed; ways to build a community of practice, create a Definition of Done, apply Liberating Structures to remote Scrum events; and more.

You can read the book cover to cover or refer to an individual chapter; its content is “snackable.”

WHO SHOULD READ THIS BOOK

The Scrum Anti-Patterns Guide is designed as an essential resource for

- Scrum Masters seeking to deepen their practice,
- Product Owners aiming for a valuable Product Goal, an actionable Product Backlog, and improved stakeholder relations, and
- Developers striving to create more customer value.

By providing a deep understanding of potential pitfalls and offering practical solutions to overcome them, this book empowers all Scrum practitioners to maximize the value of Scrum and foster a culture of continuous improvement.

HOW THIS BOOK IS ORGANIZED

The book is structured as a repository of 180-plus insights based on Scrum roles, events, artifacts, and commitments in the form of snackable content. While you can read it sequentially from beginning to end, it is not written to be consumed in that way.

When you want to better understand what challenging behaviors and practices you may encounter in a particular setting—for example, if you are having trouble with your Sprint Review—refer to the respective chapter. Use the chapter as a briefing on

the possible reasons for teammates', stakeholders', and managers' behavior and how those anti-patterns may be overcome. Typically, you will find more information on the topic in numerous footnotes pointing to additional details.

You may find reading *The Scrum Guide Reordered*³ alongside this book useful to compare theoretical patterns derived from Ken Schwaber and Jeff Sutherland *Scrum Guide*—how Scrum is supposed to be—to the reality in your organization. *The Scrum Guide Reordered* is based on about 95 percent of the text of The 2020 Scrum Guide, extended with categories such as self-management, commitments, and accountability.

Besides the Scrum anti-patterns in this book, Appendix B, “Toolbox,” contains a set of practices and exercises that are well-suited to address some of the underlying reasons for the described Scrum anti-patterns.

The map of Scrum anti-patterns presented in this book is not the “territory” but merely an abstraction of reality⁴ to help you learn and continue the exploration. Every organization applying Scrum faces its unique set of challenges. There is no one-size-fits-all approach to solving them.

3. Stefan Wolpers, *The Scrum Guide Reordered*, 2020, <https://age-of-product.com/scrum-guide-reordered>.

4. Shane Parrish and Farnam Street, “The Map Is Not the Territory,” <https://fs.blog/map-and-territory>.

SCRUM MASTER ANTI-PATTERNS

INTRODUCTION

The reasons Scrum Masters violate the spirit of the Scrum Guide are multifaceted. Typical Scrum Master anti-patterns run from ill-suited personal traits to complacency, pursuit of individual agendas, and misunderstanding of the role itself; see Figure 1.1.

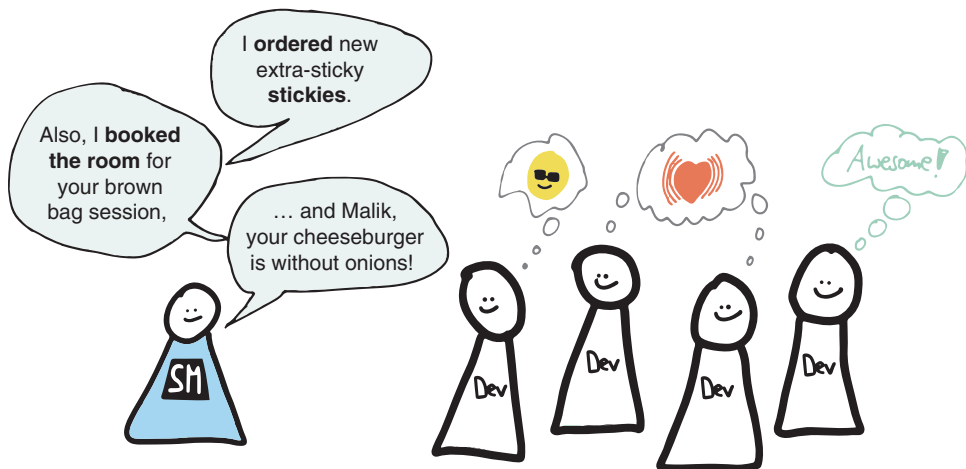


Figure 1.1 Beware of becoming a Scrum parent.

THE SCRUM MASTER ACCORDING TO THE SCRUM GUIDE

Before we start dissecting probable reasons for and manifestations of Scrum Master anti-patterns, let us revisit how the Scrum Guide defines the role of the Scrum Master:¹

- Scrum Masters are accountable for establishing Scrum, aiding all in understanding its theory and practice.
- They are responsible for the Scrum Team’s effectiveness, enabling practice improvements within the Scrum framework.
- Scrum Masters act as servant leaders for the Scrum Team and the larger organization.
- They serve the Scrum Team by coaching self-management and cross-functionality, focusing on high-value Increments, removing impediments, and ensuring effective Scrum events.
- They aid the Product Owner through effective Product Goal definition and Product Backlog management and by promoting understanding of clear backlog items, enabling empirical planning, and facilitating stakeholder collaboration.
- Serving the organization, Scrum Masters lead Scrum adoption, plan and advise on implementations, promote understanding of empirical approaches, and remove barriers between stakeholders and Scrum Teams.

The keystone of the Scrum Master role is leadership. (Too bad that it is officially no longer called “servant leadership”; I found that description to be better suited.) Unfortunately, in many cases of Scrum Master anti-patterns, it is precisely this idea that an individual does not meet.²

POSSIBLE REASONS WHY SCRUM MASTERS LEAVE THE PATH

Following are some often-observed reasons for Scrum Masters not behaving as intended:

- **Ignorance or laziness:** Some Scrum Masters assume that one approach to Scrum fits every team. They learned Scrum in a specific context and apply that same

1. Ken Schwaber and Jeff Sutherland, “Scrum Master,” The 2020 Scrum Guide™, 2020, <https://scrumguides.org/scrum-guide.html#scrum-master>.

2. See also the detailed lists of services rendered to the Product Owner, the Developers, and the organization by the Scrum Master, as defined by the Scrum Guide.

approach in every organization in which they are active, no matter the circumstances. Why go through the hassle of teaching, coaching, and mentoring if you can shoehorn the “right way” directly into any Scrum Team?

- **Lack of patience:** Patience is a critical resource that a successful Scrum Master needs to exhibit in abundance. But, of course, there is no fun in readdressing the same issue several times, rephrasing it probably, if the solution is obvious—from the Scrum Master’s perspective. So, why not tell the team how to do it “right” all the time and save all that effort? Too bad that Scrum cannot be pushed but needs to be pulled—that’s the essence of self-management. (You may observe a similar behavior resulting from boredom.)
- **Dogmatism:** Some Scrum Masters believe in applying the Scrum Guide literally, which unavoidably causes friction because Scrum is a framework, not a methodology; see Figure 1.2. Nevertheless, teaching Scrum that way feels good: team members come and ask for help; now, the Scrum Master has a purpose:
 - When Scrum Team members follow the rules, the Scrum Master has influence or authority.
 - Being among the chosen few who interpret the Scrum Guide “correctly” secures status and respect among teammates and the broader organization.
 - The Scrum Master may easily attribute the Scrum Team’s progress or success to the Scrum Master’s teaching; now, they also have proof regarding their mastery of Scrum.
 - Finally, their mastery of Scrum is a convincing argument for the organization to keep the dogmatic Scrum Master on the payroll; apparently, the Scrum Teams need an interpreter of the Scrum Guide to reap the framework’s benefits.
- **Laissez-faire turned into indifference:** Pointing the team in a direction where the team members can find a solution for an issue is good leadership. However, letting them run without guidance all the time sooner or later may turn into indifference or an I-do-not-care mentality.
- **The opportunist:** Secretly, the Scrum Master believes that this Scrum thingy is a fad but recognizes that it offers a well-paid position: “I will weather the decline in demand for project managers by getting a Scrum Master certificate. How hard can this be—the manual is merely 13 pages?” However, this conviction will inevitably bring out the opportunist’s true colors over time.



Figure 1.2 Scrum is a practical yet imperfect tool to solve complex, adaptive problems but is not an infallible methodology.

- **Frustration:** The Scrum Master has been working diligently for months, but the team is not responding to the effort. The level of frustration is growing; see Figure 1.3. There are many potential reasons for a failure at this level. Here are just a few:
 - The agile effort lacks sponsorship from the C-level of the organization.
 - There is a widespread belief that Agile is just the latest management fad and thus is ignorable.
 - The team composition may be wrong—perhaps some team members despise Scrum.
 - There is no psychological safety to address the team’s problems.
 - The company culture is antagonistic toward transparency and radical candor.
 - Individual team members harbor personal agendas unaligned with the team’s objective.

Ultimately, the Scrum Masters cannot solve this issue by themselves; its fixing is an effort of the whole Scrum Team.

- **The tactical Scrum Master:** These Scrum Masters drank HR’s Kool-Aid and believe that Scrum Master is a position, not a role. Moreover, there is a career path from junior Scrum Master to VP of Agile Coaching. Consequently, they constrain their work strictly to the Scrum Team level until being promoted.

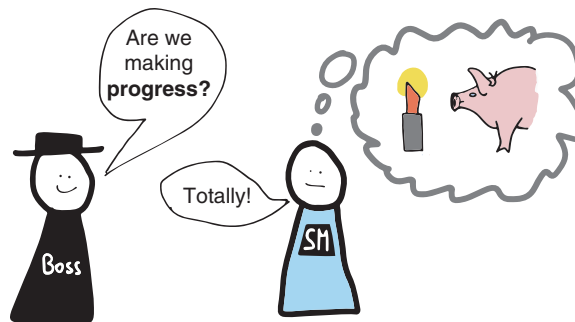


Figure 1.3 Scrum Masters are only human, too. Lend them your hand when things make no progress.

- **The rookie:** The Scrum Master might merely be inexperienced. Given that we all need to learn new skills regularly, cut them some slack in this case, and reach out to support their learning effort. Scrum is a journey, not a destination, and no one on the team travels alone.

ANTI-PATTERNS FROM ACTING AS AN AGILE (LINE) MANAGER

Agile Management is an oxymoron: you manage systems but lead people. The primary purpose of any agile practice is to empower those closest to a problem to find a solution and to help the team to self-manage over time. Self-organizing teams need coaches, mentors, and servant leaders; they do not need directive managers. Watch out for the Scrum Master anti-patterns corresponding to this agile manager attitude.

I. Agile Manager

Observation: Your Scrum Master acts like an agile line manager, dismissing Scrum's idea of self-managing teams of capable individuals who solve customer problems autonomously.

Background: Self-management does not mean the absence of management: Why would a Scrum Master or Scrum Team assume, for example, responsibility for payroll? Would that help with creating value for the customer? No. Being a self-managing team does not mean the absence of management per se, it simply means a different kind of management that is focused on helping people to grow their skills rather than telling people how to do their work.

There are various reasons why a Scrum Master may behave in this way; for example:

- **Lack of understanding:** Misunderstanding the Scrum Guide can lead a Scrum Master to act as a line manager. They may not fully comprehend the principle of self-managing teams due to a lack of training, coaching, and mentoring.
- **Fear of loss of control:** Some Scrum Masters, due to their previous roles as managers, may fear losing control and relevance. They need to learn and embrace the idea that by helping the team to self-manage, their ability to have a positive effect is multiplied, not reduced.
- **Organizational culture:** Scrum Masters may resort to old managerial behaviors if the organization still values and rewards traditional management practices. This behavior may persist, especially if the organization hasn't fully embraced agile principles, particularly building projects and products around motivated individuals and trusting them to do the job.
- **Resistance to change:** Change can be uncomfortable and difficult. Some "Scrum Masters" may resist the change to a servant leadership role, as stated in the Scrum Guide, from a traditional command-and-control management style due to ingrained habits, comfort, and familiarity.
- **Lack of trust:** A Scrum Master who doesn't trust the team's ability to self-manage might act as a line manager. This behavior contradicts the Scrum Guide's vision of a Scrum Master serving the team and fostering an environment where the team can work at its highest potential.
- **Overemphasis on delivery:** If the organization focuses overly on meeting deadlines and delivery, the Scrum Master may adopt a managerial approach to push the team, contrary to promoting sustainable development.
- **Insufficient support:** Without enough support from upper management and other departments to foster an agile environment, a Scrum Master may revert to line management out of frustration or necessity. This scenario contradicts the Agile Manifesto's principle of giving the team the needed environment and support.

Remedy: Some steps the organization and the teammates can take to help the Scrum Master overcome the "agile manager" behavior are, for example:

- **Mentorship and coaching:** Pair the Scrum Master with an experienced mentor to help them learn and adopt appropriate behaviors. The mentor can give real-time

feedback, helping the Scrum Master to understand when they're slipping into a line management role.

- **Continuous learning and training:** Promote ongoing education to ensure the Scrum Master understands the principles of the Scrum Guide and the Agile Manifesto. They need to comprehend that the role of the Scrum Master is to serve the team, not manage it. Regular refresher courses and workshops can reinforce these principles.
- **Encourage dialogue:** Team members can initiate a conversation with the Scrum Master about their concerns, being open and respectful. It's important to refer to the Agile Manifesto and Scrum principles when discussing how the Scrum Master's behavior affects the team's self-management.
- **Feedback:** Consider implementing a 360-degree feedback system, allowing team members to provide anonymous feedback to the Scrum Master if team members feel uncomfortable addressing the issue in Retrospectives.
- **Performance metrics:** Align Scrum Master performance metrics with agile principles. Instead of assessing based on output and meeting delivery deadlines, focus on team satisfaction, quality of work, and progress toward self-management and accomplishing the team's Product Goal.
- **Patience and persistence:** It takes time to unlearn ingrained behaviors and adopt new ones. Be patient with Scrum Masters who are transitioning from a traditional management role. Regularly reinforce the importance and benefits of agile practices while addressing any concerns or resistance they might have.

Food for Thought: As a Scrum Master, gaining work experience in a Scrum Team as a Developer or Product Owner might be helpful in building empathy.

2. Running Scrum Events by Allowing Someone to Speak

Observation: The Scrum Master hosts all Scrum events down to the level where team members wait for permission to contribute.

Background: When team members seek approval from the Scrum Master before speaking out, the Scrum Master has already left the facilitation role in favor of the supervisor mode. While Scrum Masters support their teams by facilitating events, it does not imply that they always run the show with a domineering attitude.

Their job is to help their teammates to accomplish the purpose of the respective Scrum event, not to enforce an orderly procedure from their perspective.

Remedy: There are a few things the teammates can do to encourage their Scrum Master to relinquish an overly controlling role:

- **Initiate open discussion:** Address the issue respectfully within the team, perhaps during a Retrospective. Make the conversation about the process and roles, not the person.
- **Assert autonomy:** Begin contributing without waiting for the Scrum Master's prompt. Show initiative in discussing and deciding tasks, thus reinforcing the concept of a self-managing team.
- **Encourage peer facilitation:** Suggest a rotation of the facilitator role for Scrum events to promote shared ownership and demonstrate that the Scrum Master doesn't always have to be in control.
- **Educate on role:** Gently remind the Scrum Master about the core aspects of their role: serving the team and the Product Owner, removing impediments, and not managing tasks or people.
- **Seek external guidance:** If internal efforts aren't effective, consider contacting an agile coach or experienced Scrum Master from another team for advice or mediation.

Each of these steps aligns with the values of the Agile Manifesto, encouraging individuals and interactions over processes and tools and promoting a more effective, collaborative way of working.

3. The Administrative Assistant

Observation: The Scrum Master pampers the Scrum Team to a level that keeps the team dependent on their services. For example, they might

- Organize meetings,
- Purchase stickies and Sharpies,
- Take notes, write protocols,
- Update the Product Backlog on behalf of the team (whether in a tool or not),
- Organize budget for tools, education, and team activities,

- Preselect applicants for open team positions, and
- Channel stakeholder communication—you get the idea.

Background: Sometimes, Scrum Masters believe that being supportive of their teams requires assuming the role of the personal team assistant. That is an unfortunate misinterpretation of their role; they are supposed to help their teammates become self-managing, and everyone knows how to order or “organize” stickies and Sharpies. More critical, however, is when Scrum Masters decide to keep the team in the dark about principles and practices in order to secure their job.

What might be the motivation for such behavior? Some aspects are as follows:

- **Desire for power and control:** The Scrum Master may relish the feeling of indispensability and control they gain by keeping the team reliant on their services, appearing more knowledgeable and indispensable.
- **Positioning:** If they view administrative tasks as “beneath” the team members, they might attempt to create an image of themselves as self-sacrificing leaders by taking on these tasks.
- **Avoidance of accountability:** By focusing on administrative tasks and not their actual responsibilities—such as coaching the team and the wider organization about Scrum—they might be sidestepping more challenging, accountability-laden aspects of their role.
- **Comfort in familiar tasks:** They might find comfort in administrative tasks, especially if they previously held roles that required such tasks, giving them a sense of control and accomplishment.
- **Lack of confidence:** They might lack confidence in the team’s ability to manage these tasks or coach and facilitate effectively, causing them to retain control over seemingly trivial matters.

Remedy: Similar to the Agile Manager anti-pattern, there are a few steps the teammates can take to help the Scrum Master overcome this behavior:

- **Peer feedback:** Offer constructive feedback to the Scrum Master. Encourage teammates to do the same, creating a unified voice that calls attention to the issue.
- **Proactive participation:** Don’t wait for the Scrum Master to assume tasks. Proactively take ownership of tasks such as updating the task board or organizing events, demonstrating your ability to self-manage.

- **Ask for coaching:** Request that the Scrum Master focus more on facilitating, coaching, and mentoring than on taking over administrative tasks. Suggest working sessions to help deepen understanding of Scrum principles and practices.
- **Education:** Share resources such as books, articles, or videos on the role of a Scrum Master, or organize team sessions with Scrum Masters from other teams where they share their success patterns.
- **Engage management:** If the Scrum Master's behavior persists, you may need to involve senior management or human talent department. Maintaining a productive work environment is crucial, and sometimes it requires outside intervention to resolve such issues.

The aim is not to antagonize the Scrum Master but to help them understand and fulfill their role more effectively. Always approach these steps with respect and an eye toward improving team dynamics and productivity.

4. Tracking Flawed or Useless Metrics

Observation: The Scrum Master tracks metrics that focus on output or individual performance, like the number of story points completed, personal velocity, or estimation accuracy.

Background: Generally speaking, metrics help to clarify the current situation and allow us to gain insight on change over time. Without metrics, assessing any effort or development will be open to gut feeling and bias-based interpretation.

A metric should therefore be a leading indicator for a pattern change, providing an opportunity to analyze the cause in time. The following three general rules for agile metrics have proven to be useful:

1. Track only those metrics that apply to the team. Ignore those that measure the individual.
2. Do not measure parameters because they are easy to track. (Various project management tools offer out-of-the-box reports.)
3. Record context as well. Data without context, for example, the number of available team members or the intensity of incidents during a Sprint, may turn out to be nothing more than noise.

While utilizing predefined reports of the team's Product Backlog management software is probably a sign of ignorance or laziness, keeping track of individual performance metrics—such as story points per Developer per Sprint and reporting them to that person's line manager—is a critical warning sign.

Remedy: What action may a Scrum Team take to convince the Scrum Master to switch to measuring meaningful agile team metrics?

There are a few options; for example:

- **Explain metrics in context:** The team can educate the Scrum Master about agile values, principles, and practices and their impact on identifying agile metrics. For example, it's essential to highlight that individual-focused metrics undermine team collaboration and often lead to poor results.
- **Propose relevant metrics:** The team can propose tracking more relevant and beneficial metrics such as lead time, cycle time, or team health. These metrics focus more on the team's performance and value delivery than on individual output. It is helpful if the team also explains why these metrics are more meaningful and how they improve the team's effectiveness and efficiency.
- **Self-organization:** As a self-managing team, members can take control of the metrics themselves. They can start tracking the proposed metrics and demonstrate their value over several Sprints. Seeing the positive results might convince the Scrum Master to change their approach.
- **External facilitator:** If the situation does not improve, the team could suggest having an external agile coach or another experienced Scrum Master facilitate a workshop. The team and Scrum Master can openly discuss the issue and find a solution. The external facilitator may be able to provide a neutral viewpoint and effective advice.

5. Spying for Management

Observation: During the Sprint, the Scrum Master reports to the management whether the Scrum Team will meet the current forecast.

Background: I took this from a job offer I received: “You will coordinate and manage the work of other team members, ensuring that timescales are met and breaches are escalated.”

Besides completely misunderstanding the Scrum Master's role, there are some other reasons that may motivate a Scrum Master to disregard Scrum values:

- **Pressure from management:** The Scrum Master may be under pressure from management to provide frequent updates and assure progress. This demand could stem from a lack of understanding of Agile and Scrum principles among the leadership, particularly the emphasis on trust and empowerment over monitoring and control.
- **Lack of trust in team:** The Scrum Master may lack trust in the team's ability to meet their forecast and thus feels the need to keep management informed. The Scrum Master's lack of trust may even contribute to their ineffectiveness by reducing their motivation.
- **Insecurity about role:** The Scrum Master might feel insecure about their role and value within the organization. By reporting to the management, they may try to demonstrate their importance and control over the team's work.

Remedy: What action should a Scrum Team take to convince the Scrum Master that this behavior is inappropriate? I would start with the first and second of these suggestions and keep the third in reserve:

1. **Communicate openly:** Team members should initiate a general conversation with the Scrum Master about their concerns, explaining how this behavior undermines trust and self-management principles. They should pursue this respectfully and constructively, using specific instances as examples, for instance, during a Retrospective. The Retrospective's outcomes could also be shared with the broader organization, helping others understand the team's way of working and their commitment to the Scrum values.
2. **Self-reporting:** The team can propose a self-reporting mechanism whereby they take responsibility for reporting their progress. The process may include visual radiators or dashboards that everyone, including management, can access. In addition, the team may consider publishing a newsletter to make stakeholder communication as simple as possible.
3. **Involve a neutral party:** If direct communication doesn't yield results, involving a neutral party like an agile coach or a trusted leader in the organization could be helpful. They can facilitate a discussion and help resolve the issue.

6. Team Harmony over Conflict Resolution

Observation: The Scrum Master sweeps conflict and problems under the rug by not using Sprint Retrospectives to address them openly.

Background: There are several reasons why a Scrum Master reverts to peacekeeping; for example:

- **Overemphasis on positivity:** The Scrum Master may erroneously believe that Retrospectives should focus only on what went well to keep the team's morale high. However, this view neglects the equally important aspect of addressing what didn't work and finding ways to improve, a key element of the Agile Manifesto's emphasis on continuous improvement.
- **Fear of confrontation:** The Scrum Master may be uncomfortable with conflict and avoid addressing it openly. This behavior often arises from a misunderstanding of the nature of conflict, viewing it as harmful rather than an opportunity for improvement and team growth.
- **Preservation of the status quo:** The Scrum Master might be afraid of upsetting the existing emotional balance of the team, even if this equilibrium isn't producing the best results. This idea may result from a lack of understanding of the continuous improvement principles underpinning Agile and Scrum.
- **Personal relationships:** If the Scrum Master has close personal relationships within the team, they may fear damaging the relationship by bringing up contentious issues, directly violating the Scrum value of openness.
- **Misunderstanding the role:** The Scrum Master may not fully understand their role and responsibility to facilitate effective Retrospectives. They may see their role as a peacekeeper rather than an improvement facilitator.
- **Lack of skills or training:** The Scrum Master may lack the necessary skills to facilitate a Retrospective effectively, such as being able to manage conflicts, drive constructive discussions, and aid the team in deriving actionable insights. This skill gap points to a need for further professional development in these areas.
- **Fear of repercussions:** The Scrum Master may be apprehensive about the potential fallout from exposing issues. This fear could be due to a corporate culture that penalizes mistakes rather than viewing them as learning opportunities, contradicting Agile's principle of regular reflection for effective adjustment.

- **Avoidance of accountability:** By not discussing issues openly, the Scrum Master may be trying to avoid assigning or taking accountability for problems, which directly conflicts with the Scrum value of commitment to achieving team goals.
- **Belief in privacy:** The Scrum Master may believe that certain issues are private or sensitive and thus not suitable for open discussion in a Retrospective.

Remedy: So, what can we do about this anti-pattern? How can we help the Scrum Master overcome their flawed understanding of Scrum values, openness, conflict, and Retrospectives? A few things come to mind; for example:

- **Feedback and communication:** Encourage the team to openly communicate with the Scrum Master about how their action of ignoring conflicts is affecting the team's performance and morale. Honest feedback may serve as a wake-up call.
- **Coach for transparency and openness:** Foster an environment where transparency, openness, and constructive feedback are encouraged and appreciated. Show the Scrum Master the value of these practices by modeling them in action.
- **Promote a learning culture:** Cultivate a culture that views mistakes and conflicts as opportunities for learning and growth, not as failures. This approach aligns with the Scrum principle of inspecting and adapting to improve.
- **Address fear of conflict:** If the Scrum Master avoids conflict, address this issue directly. Discuss the benefits of constructive conflict and how it leads to better solutions and stronger team cohesion.
- **Invite an agile coach or experienced Scrum Master:** In persistent cases, it could be beneficial to invite an experienced agile coach or Scrum Master to guide your Scrum Master. They can provide an outside perspective and suggest effective ways to handle conflicts and problems.

At the same time, be also aware that self-management is a privilege earned through delivering valuable outcomes. Suppose the team fails to deliver due to internal strife and proves incapable of overcoming the situation. In that case, there may be a moment when a Scrum Team should take action against an ineffective Scrum Master, Product Owner, or any team member.

If a team can't improve or fix itself, stakeholders may lose confidence, cancel the project, and potentially dissolve the team, emphasizing the responsibility of management to intervene when necessary.

SCRUM MASTER ANTI-PATTERNS BY SCRUM EVENTS

THE SPRINT PLANNING

The anti-patterns in this section focus on the Sprint Planning.

7. No Slack Time

Observation: The Developers regularly bow to the hard-pushing Product Owner: “Last Sprint, you delivered twenty-five story points, and now you are choosing only seventeen?” Consequently, they accept more issues into the Sprint Backlog than they can stomach without the Scrum Master’s intervention.

See Chapter 3, anti-pattern 8, for a detailed description.

8. Unrefined Work Items

Observation: The Scrum Master does not address accepting “unrefined” Product Backlog issues into the Sprint Backlog during Sprint Planning. (This observation refers to regular work, not emergencies or last-minute changes of high importance.)

Background: The Scrum Master’s choice increases the risk that the Developers will probably not accomplish the Sprint Goal, rendering a core Scrum principle useless: reliably providing valuable, potentially shippable Increments every single Sprint.

What reasons may a Scrum Master have to ignore the issue? Some things come to mind:

- **Overreliance on team autonomy:** The Scrum Master may think it is the Developers’ sole responsibility to identify the Product Backlog items for the Sprint Backlog without interference.
- **Complacency with current practices:** The Scrum Master might be complacent with the current practices, especially if the team has delivered despite the absence of refined Product Backlog items. However, this practice goes against the Agile Manifesto’s principle of continuous attention to technical excellence and good design.
- **Perceived lack of time:** The Scrum Master may believe there isn’t enough time for thorough refinement, not objecting to the Developers’ pushing ahead with unrefined Product Backlog items.

- **Lack of experience:** A Scrum Master new to the role may not fully understand the implications of accepting unrefined Product Backlog items into the Sprint Backlog, underestimating the associated risk.
- **Avoidance of conflict:** The Scrum Master might avoid conflict within the team or with stakeholders by not insisting on refinement. They may perceive that challenging the Developers' decision may lead to disagreements.

Remedy: The inherent risk that repeating an exception turns it into the new normal is obvious. So, how can we counter this anti-pattern of not addressing the increased risk of accepting unrefined Product Backlog items into the Sprint?

Some suggestions are as follows:

- **Promote the benefits of refinement:** The Scrum Team can work together to emphasize the value of Product Backlog refinement. Highlighting its role in reducing uncertainties, managing risks, and enabling more accurate estimations may help the Scrum Master understand its importance.
- **Establish refinement practices:** As a Scrum Team, schedule regular Product Backlog refinement sessions to ensure this activity becomes a part of the team's routine. The Scrum Master can facilitate these sessions until the team can handle them autonomously.
- **Foster transparency:** Continually reinforce the Scrum value of transparency. Show how it's essential for every team member to understand what the team will work on and the potential risks associated with unrefined Product Backlog items.
- **Promote shared understanding:** Ensure that everyone on the Scrum Team has a shared understanding of each Product Backlog item before it's pulled into a Sprint. The Scrum Team can support the creation of a shared understanding by having discussions, asking questions, and seeking clarifications during Sprint Planning and Product Backlog refinement sessions.
- **Feedback and inspection:** Encourage regular inspection of the process and feedback. If the team experiences negative consequences from working on unrefined Product Backlog items, use this opportunity to discuss the issue and brainstorm solutions.
- **Encourage stakeholder education:** If external pressure is an issue, consider sessions where the Scrum Team educates stakeholders about the principles of Scrum and the importance of Product Backlog refinement. Make it clear that pushing unrefined

items into the Sprint can hurt the quality of the product, the predictability of the team's work, and can hinder accomplishing the Sprint Goal in general.

- **Consider a Definition of Ready:** Finally, the team could agree on a Definition of Ready, specifying the conditions a Product Backlog item must meet before Developers can select it during Sprint Planning. (However, beware of turning such a practice into a dogmatically applied approval gate.) Learn more about the Definition of Ready in Chapter 15, anti-pattern 3.

9. No Improvements

Observation: While the Scrum Team agrees on improvement items during Retrospectives, they never consider them during Sprint Planning.

See Chapter 6, anti-pattern 16, for a detailed description of this anti-pattern.

THE SPRINT

The following anti-patterns focus on the mishandling of the Sprint itself.

10. Flow Disruption

Observation: The Scrum Master is unable to prevent stakeholders from disrupting the workflow of the Scrum Team during the Sprint.

Background: There are many manifestations of how stakeholders can interrupt the Scrum Team's flow during a Sprint, impeding the team's productivity and endangering accomplishment of the Sprint Goal. Some classic examples follow:

- **Misusing Scrum events:** Stakeholders or managers turn the Daily Scrum into a reporting session.
- **Disrupting Developers:** Stakeholders constantly address Developers directly during the Sprint, ignoring that interruptions harm the Scrum Team's effectiveness.
- **Team membership disruption:** Line managers take team members off the Scrum Team, assigning them to other tasks, or add members to the Scrum Team without prior consultation of the team members.

The last example is particularly challenging, as creating a Scrum Team is expensive due to the inevitable drop in productivity during the norming and storming phases of the team-building phase. Hence, changing their composition is a critical decision

that shall include the team members. Scrum Teams are not talent pools in disguise at the disposal of line managers.

Remedy: Scrum Masters should not restrict stakeholders' access to team members, but they should educate stakeholders on how to best communicate with the Scrum Team to foster an effective relationship. Some suggestions on how to proceed are as follows:

- **Set boundaries:** Establish clear guidelines about when and how stakeholders can interact with the Scrum Team, aligning with the Scrum Guide's assertion of minimizing disruptions.³
- **Educate stakeholders:** Educate stakeholders on the impact of disruptions on the Scrum Team's productivity and how they jeopardize the Sprint Goal.
- **Assert team autonomy:** Uphold the principle that Scrum Teams manage their composition and workflow, underscoring their right to self-organize. (Advocating for this principle proves challenging in many organizations.)
- **Preserve Scrum events:** Protect the integrity of Scrum events, emphasizing their importance in supporting team coordination and communication, not for reporting purposes.
- **Strengthen Product Owners:** Emphasize the Product Owner's role as the first line of communication between stakeholders and the Scrum Team to maintain focus.

II. Defining Technical Solutions

Observation: An engineer turned Scrum Master is now "suggesting" how the Developers implement issues. Alternatively, the Product Owner or an outsider, such as a technical lead, is pursuing the same approach.

Background: Why might a Scrum Master overstep the boundaries of their role and get involved in implementation questions when the Scrum Guide clearly states that any decision on how to turn a Product Backlog item into a done Increment is solely the decision of the Developers? Some reasons may include:

- **Old habits:** The Scrum Master, an engineer in the past, may naturally gravitate toward providing solutions based on their technical experience. They may

3. Check out Jimmy Janlén's interruption buckets in *Toolbox for the Agile Coach: 96 Visualization Examples: How Great Teams Visualize Their Work* (Crisp Publishing, 2015).

inadvertently revert to their old role out of habit or believe they are genuinely helping the Developers.

- **Underestimation of team skills:** The Scrum Master may underestimate the Developers' capabilities, believing that the team needs their guidance in technical matters. This perception could stem from a lack of trust or unfamiliarity with the team's skills and competencies.
- **Pressures and self-preservation:** If management or stakeholders pressure the Scrum Master to deliver specific results, they might feel compelled to interfere in the Developers' work. They could think their role is on the line if the team doesn't meet these expectations.
- **Control issues:** The Scrum Master, because of personal insecurities or a perceived lack of progress, may feel the need to control the process. They may believe they can speed up the process by directing the work rather than allowing the Developers to self-manage.
- **Inadequate transition:** Transitioning from a technical role to a Scrum Master is a significant shift. It requires a change in mindset from being a doer to becoming an enabler. If this transition is not adequately supported or understood, the Scrum Master might fall back into the comfort zone of their previous role.
- **Fear of obsolescence:** The Scrum Master may worry that by not involving themselves in the technical decisions, they will become disconnected from the actual work, which could make them feel less valuable or even obsolete.

Remedy: So, what can we do about this Scrum Master anti-pattern? How can we convince the Scrum Master to let go of their former self as a Developer and doer and embrace the role of an enabler? Consider the following:

- **Open dialogue:** Have a frank conversation, perhaps during a Retrospective, about the Scrum Master's role. Highlight the principles of the Agile Manifesto and the Scrum Guide, emphasizing that the Scrum Master's role is to facilitate, not dictate, the implementation. Discuss the benefits of self-organizing teams and how they improve team productivity and creativity.
- **Training and education:** Encourage the Scrum Master to attend further training or workshops to better understand the nuances of their role. It's beneficial to learn from the experiences of other Scrum Masters who have transitioned from technical positions.

- **Peer feedback:** Encourage the Developers to provide feedback whenever they feel the Scrum Master is overstepping their boundaries. They should give this feedback constructively, emphasizing how such behavior affects the team's productivity and autonomy.
- **Role modeling:** Share examples of successful Scrum Masters who have effectively transitioned from a technical role, showing the Scrum Master that stepping back from implementation does not diminish their value, reputation, or standing.
- **Mentoring or coaching:** A more experienced Scrum Master or agile coach can guide the transitioning Scrum Master, providing them with strategies to combat the urge to delve into technical details.
- **Self-reflection:** Encourage the Scrum Master to practice self-reflection. It might be helpful for them to question why they need to involve themselves in the implementation and how they can work toward overcoming this inclination.

THE DAILY SCRUM

12. Not Supporting Struggling Developers

Observation: A Developer experiences difficulties accomplishing an issue over several consecutive days, and nobody on the Scrum Team offers help. Moreover, the Scrum Master fails to facilitate the necessary discussion.

Background: There may be many reasons for a Developer's struggle. Here are just a few:

- **Task complexity:** The task may be more complex or challenging than initially anticipated, for example, due to technical debt.
- **Lack of experience or skills:** The Developer might not have the necessary skills or experience to complete the task.
- **Health or personal issues:** Personal or health issues may impact the Developer's performance.
- **Fear of criticism or failure:** The Developer might be reluctant to admit difficulties due to fear of judgment or criticism.
- **Lack of team collaboration:** The Developer workload has reached such an unproductive level that they no longer can support each other.

The worrying issue is less the struggle than the lack of support by the teammates and, more important, the Scrum Master.

Remedy: There are many ways to address the preceding issues, from improving craftsmanship to employing an effective Definition of Done to regular skill-sharing sessions to creating a safe space where no one is judged but is supported.

However, regarding the failure of the Scrum Master to detect the necessity to support a team member, I suggest using a simple visualization technique from Kanban: the “work item age.” It is about tracking and displaying the time a Product Backlog item has spent in a specific state, such as “in development.” This time is often represented visually on the Kanban board, for example, by adding a marker for each day the item did not move. Once an item accumulates a predefined number of markers, the team offers support.⁴

It is helpful to agree on this practice as a part of the team working agreement to create a shared understanding among all team members that they need to support each other.

13. Not Preventing Stakeholders from Attendance

Observation: Although the Developers decided to limit attendance to the Daily Scrum to the Scrum Team members, stakeholders show up uninvited, and the Scrum Master fails to address the issue.

Background: It is the Developers’ prerogative to decide how to run the Daily Scrum. If they choose to keep stakeholders away from it, this decision needs to be respected by everyone else.

Following are possible reasons that Scrum Masters fail to address the inappropriate behavior of the stakeholders attending the Daily Scrum uninvited:

- **Inadequate support from the organization:** Without enough backing from the organization, the Scrum Master might find it challenging to address the issue, particularly in organizations that consider their role to be team-focused.

4. Yuval Yeret, “4 Key Flow Metrics and How to Use Them in Scrum’s Events,” *AgileSparks*, May 10, 2018, <https://medium.com/agilesparks/4-key-flow-metrics-and-how-to-use-them-in-scrums-events-c63a5a2e1bcf>.

- **Conflict avoidance:** They may want to avoid conflicts or uncomfortable stakeholder conversations.
- **Lack of confidence:** The Scrum Master may hesitate to confront stakeholders higher up in the organization. (Ask yourself: Would you show Elon Musk the door?)
- **Ineffective communication skills:** The Scrum Master may lack the communication skills necessary to deal with high-level stakeholders.
- **Stakeholder pressure:** There could be high pressure from stakeholders on the team to perform better, making it difficult for the Scrum Master to intervene.
- **Fear of repercussions:** The Scrum Master might fear negative repercussions, like job loss or other political consequences within the organization.

Remedy: There are several ways a Scrum Master can educate stakeholders who show up uninvited to the Daily Scrum and explain that they need to stop this behavior unless the Developers invite them:

- **Set clear boundaries:** Communicate the rules and expectations for the Daily Scrum, emphasizing that it is an event for and by the Developers and the need to respect their autonomy.
- **Provide education:** Teach stakeholders about the purpose and structure of the Daily Scrum, explaining why their presence could disrupt the event.
- **Stakeholder meetings:** If necessary, arrange separate meetings for stakeholders to voice their concerns and queries, ensuring they feel heard without interrupting the Daily Scrum.
- **Use visual aids:** Use signs or visual cues indicating that the Daily Scrum is in session and interruptions are not welcome unless invited. (An “On Air” sign works for Scrum Teams, too.)
- **Encourage self-management:** Foster the Developer’s self-esteem, enabling them to address the issue directly with the stakeholders.
- **Involve management:** If necessary, involve higher management to support Scrum’s rules and the Scrum Master’s efforts.

THE RETROSPECTIVE

The final set of Scrum Master anti-patterns addresses the Sprint Retrospective.

14. Groundhog Day

Observation: The Sprint Retrospective never changes in composition, venue, or length.

Background: In this case, the Scrum Team tends to revisit the same issues repeatedly—it's Groundhog Day without the happy ending. It has become a boring exercise, a ritualized formality—what was good during the last Sprint, what was bad, let's have some action items—that proves of little and, in time, diminishing value to the Scrum Team.

Remedy: Try to vary the format of the Retrospective; for example:

- Go meta and have a Retrospective on your Retrospective.
- Have a themed Retrospective.⁵
- Change the venue or the time.
- Disguise the Retrospective as a team breakfast.

There are so many things a Scrum Master can do to make Retrospectives great again and reduce the absence rate. Be creative with every single Retrospective and aim not to repeat a single format or theme. Liberating Structures and many other websites dedicated to organizing Retrospectives make this an affordable and worthwhile investment.

Toolbox Tip

Retromat.org is a free website by Corinna Baldauf dedicated to helping you tailor Retrospectives to your Scrum Team's needs: "Planning your next agile retrospective? Start with a random plan, change it to fit the team's situation, print it and share the URL. Or browse around for new ideas!"

15. Let's Have the Retro Next Sprint

Observation: The Scrum Team postpones the Retrospective to the next Sprint.

See Chapter 9, anti-pattern 4, for a detailed description.

5. Use themed Retrospectives to be creative as a team. For many Westerners, for example, Halloween may come to mind. Other topics may include your favorite video game, song, movie, opera, or book. What about travel, gardening, or cooking? There is no limit to your creativity. (Pro tip: Organize your themed Retrospective during a team-building event.)

16. #NoRetro

Observation: There is no Retrospective, as the team believes there is nothing to improve, and the Scrum Master accepts this notion.

See Chapter 9, anti-pattern 1, for a detailed description.

17. #NoDocumentation

Observation: No one is taking minutes for later use.

See Chapter 9, anti-pattern 10, for a detailed description.

18. The Blame Game

Observation: The Retrospective is an endless cycle of blame and finger-pointing.

See Chapter 9, anti-pattern 14, for a detailed description.

19. No Psychological Safety

Observation: One or two team members dominate the Retrospective.

Background: This communication behavior is often a sign of a compromised environment that is no longer psychologically safe.

Retrospectives support a Scrum Team's path to excellence only when everyone can address issues and provide their feedback free from third-party influence. If some team members dominate the conversation and perhaps even bully or intimidate other teammates, the Retrospective will fail to provide such a safe place. This failure will result in participants dropping out of the Retrospective, rendering the results less valuable.

Remedy: It is the primary responsibility of the Scrum Master, in their role as a chief facilitator,⁶ to ensure that everyone is heard and has an opportunity to voice their thoughts.

6. I believe that every Scrum Team member needs to be able to run any Scrum event, Retrospectives included. Of course, the Scrum Master will likely be the most skilled facilitator. However, that does not mean the Scrum Master must be the default facilitator.

By the way, equally distributed speaking time is, according to Google, also a sign of a high-performing team. Learn more: [What Google Learned from Its Quest to Build the Perfect Team.](#)⁷

If a Scrum Master needs to restore psychological safety, several angles may prove successful:

- Set clear expectations for Retrospectives by pointing to Scrum Values, emphasizing the importance of respectful communication, active listening, and equal participation. Encourage the team to hold each other accountable to these rules.
- Seek to speak with the dominant individuals privately, addressing their behavior and its impact on the team. Help them understand the importance of psychological safety, and request their support in creating a more inclusive environment.
- During Retrospectives, watch the dynamics and intervene when necessary to maintain a balanced and inclusive discussion. Redirect the conversation if it becomes unproductive or if specific individuals start to dominate or bully others.
- Utilize structured discussion formats, such as round robin or Conversation Café, where each team member takes turns sharing their thoughts. These practices can help ensure that everyone has an opportunity to speak and that the conversation remains balanced.
- Actively involve introverted or less vocal team members by asking for their opinions, prompting them to share their thoughts, and acknowledging their contributions. Offer support and reassurance when necessary. However, please note that not all of your less vocal teammates may feel comfortable with this approach.
- Offer different ways for team members to share their thoughts and feedback, such as written input or anonymous submissions and surveys. Anonymity encourages everyone to share their perspectives, even if they are uncomfortable speaking up during the Retrospective.
- Inspect the effectiveness of Retrospectives and adapt as needed. Gather feedback from the team on how the Retrospectives work, and implement changes to improve psychological safety.

7. Charles Duhigg, "What Google Learned from Its Quest to Build the Perfect Team," *The New York Times*, February 28, 2016.

(See also the Remedy section of anti-pattern 14, The Blame Game, in Chapter 9.)

20. Excluding Stakeholders All the Time

Observation: The Scrum Team categorically rejects having Retrospectives with their stakeholders.

See Chapter 9, anti-pattern 5, for a detailed description.

FOOD FOR THOUGHT

Agile Coaching by Scrum Master? How come Scrum Masters are often stuck in a tactical role at the team level when we all know that a successful Scrum Master increasingly shifts their focus from the Scrum Team to the organization? Or is this effect just collateral damage of a well-known scaling framework's classification of the Scrum Master role?

Scrum at the team level is not rocket science; however, the serious impediments to becoming agile often reside at the organizational level. So why does it seem necessary to create a new role—the agile coach or organizational coach—to address this need?

Redundancy: Is a Scrum Master of a successful team redundant at the team level? Can they consequently move to a pool of Scrum Masters from which a team needing support can request assistance?

CONCLUSION

There are plenty of possibilities to fail as a Scrum Master. Sometimes, it is the lack of organizational support. Some people are not suited for the job. Others put themselves above their teams for questionable reasons. Moreover, some Scrum Masters lack feedback from their Scrum Teams and stakeholders. Whatever the case, try to lend your Scrum Master in need a hand to overcome the misery. Scrum is a team sport.

Toolbox

Regarding the following tips and exercises, you will find detailed descriptions in Appendix B:

- **“20 Questions from a New Scrum Master to the Scrum Team”** comprises 20 questions that fit into a 60-minute timebox, from Product Backlog forensics to metrics to team challenges and technical debt.
- **“20 Questions from a New Scrum Master to the Product Owner”** comprises 20 questions addressing the future collaboration between the two individuals and the rest of the Scrum Team.
- **“20 Questions from a New Scrum Master to the Developers”** comprises 20 questions addressing the foundations of a Scrum Team’s capability to build valuable products: from technical excellence and what it takes to achieve this high proficiency level to technical debt.
- **“Agile Metrics—The Good, the Bad, and the Ugly”** points to suitable agile metrics reflecting either a Scrum Team’s progress in becoming agile or an organization’s progress in becoming a learning organization.
- **“Building Communities of Practice”** helps win hearts and minds within the organization. It provides authenticity to the agile transformation—signaling that the effort is not merely another management fad.
- **“How to Create a Definition of Done”** details a collaborative exercise of the whole Scrum Team, probably including some of the stakeholders at a later stage, to create, inspect, and adapt a Definition of Done.
- **“Meta-Retrospectives—How to Get Customers and Stakeholders Onboard”** details an excellent Retrospective format to foster collaboration within the extended team, including your team’s stakeholders, to build a shared understanding of the big picture and create valuable action items.
- **“Peer Recruiting”** is a guide on how to hire Scrum Masters (and other agile practitioners) by including self-managing teams in the decision process.
- **“Retrospectives with Distributed Teams”** describes a string of Liberating Structures microstructures that distributed Scrum Teams can use to organize meaningful Retrospectives.
- **“Stakeholder Communication Tactics”** comprise eleven proven stakeholder communications tactics to help you not just do Agile but *become* agile.

This page intentionally left blank

INDEX

Numbers

- 1–2–4-All, 333
- 15% solutions, 352
- 20 questions
 - from a new Scrum Master to the Developers, 312–315
 - from a new Scrum Master to the Product Owner, 308–312
 - from a new Scrum Master to the Scrum Team, 305–308
- 25/10 crowdsourcing, 352
- 100% in advance anti-pattern, 35, 203
- 360-degree feedback system, 7

A

- absent Product Owner anti-pattern, 42–43
- “acceptance” by the product owner anti-pattern, 51
- acceptance criteria, Product Backlog, 206
- additional work anti-pattern, 225–226
- administrative assistant anti-pattern, 8–10
- agile, 6–7
 - anti-patterns, 5–6
 - CoP (community of practice), 325

- metrics, 316
 - bad, 322–323
 - ugly, 323
 - practices, cherry-picking, 88–89
 - Scrum Master, 2
- Agile Fluency game, 89
- anonymous poll, 318–319
- anti-pattern/s. *See also* remedy
 - Definition of Done, 277
- Developer
 - board out of date, 59–60
 - cutting corners to meet an increment’s deadline, 62–63
 - daily status report, 70–71
 - death by PowerPoint, 75, 165
 - dispensable buffer, 76–77
 - excessive feedback, 72
 - fixed scope, 221–222
 - gold-plating, 61–62, 113
 - ignoring the Definition of Done, 278
 - lack of professionalism, 58–59, 112
 - lost Scrum Team, 68–69
 - monologuing, 73, 147–149
 - no capacity check, 66, 127–128

- no impediments, 73–74, 149
- No Routine, 68
- no slack time, 64–66
- no time for refinement, 77–78
- no WiP limit, 57–58, 112
- not supporting struggling developers, 73
- planning meeting, 146–147
- planning too detailed, 66, 220
- problem-solving session, 71–72
- same faces again, 75, 166–167
- shadow Sprint Backlog, 224–225
- showing off undone work, 75–76
- side gigs, 112
- submissive engineers, 79–81
- technical debt, 63–64
- too little planning, 66–67
- too much estimating, 129, 220–221
- unpreparedness, 69–70
- incentivized stakeholder
 - financial incentives to innovate, 103–104
 - “My Budget” syndrome, 98–99
 - selling non-existent features, 101–103
 - “We know what to build”, 100–101
- IT management
 - all hands to the pumps without Scrum, 96–98
 - reassigning team members, 119–120
- organization-level
 - line managers attend Retrospective, 192–193
 - no suitable venue, 191–192
 - Sprint success defined by output, not outcome, 267–268
- Product Backlog
 - 100% in advance, 203
 - everything is detailed and estimated, 205
 - INVEST?, 205–206
 - involving the Scrum team, 211–212
 - missing acceptance criteria, 206
 - outdated items, 204
 - oversized, 202–203
 - prioritization by proxy, 200–201
 - too detailed acceptance criteria, 206–207
- Product Goal
 - Product Goal establishes how to accomplish the goal, 254–255
 - Product Goal is not transparent, 251
 - Product Owner forces the impossible upon the Scrum Team, 249–250
 - Product Owner singlehandedly creates Product Goals, 249
 - pursuing multiple, 248
 - there is no Product Goal, 247–248
- Product Owner
 - 100% in advance, 35
 - absent, 42–43
 - additional work, 225–226
 - assigning tasks to team members, 46–48
 - changing work items from the Sprint
 - Backlog during the Sprint, 111
 - copy & paste, 32–33
 - crowding-out collaboration, 36–37
 - delaying, 44–45
 - “I know it all”, 37, 163–164
 - introducing additional work, 48–50, 150
 - involving the Scrum team, 39–40
 - last minute changes, 133–134
 - last-minute Product Backlog, 39
 - no business objective, 271
 - no research, 38–39, 207
 - no Sprint cancellation, 45–46
 - output focus, 40–42
 - overreaching, 34–35
 - part-time owner, 31–32
 - prioritization by proxy, 35
 - Product Goal exclusionism, 251–252
 - Scrum team has no Sprint goal, 40
 - Sprint cancellations without consultation, 45
 - storage for ideas, 207–208
 - technical debt, 209–211
 - unfinished business, 132–133
 - what are we fighting for, 132
- Scrum Master
 - acting as an agile manager, 5–6
 - administrative assistant, 8–10
 - Blame Game, 24, 189–191

- Daily Scrum enforcer, 150–151
- defining technical solutions, 18–19
- excluding stakeholders all the time, 26
- flow disruption, 17–18, 113
- Groundhog Day, 22–23
- no improvements, 17, 139
- No Psychological Safety, 24–26
- no slack time, 15–17
- #NoDocumenatation, 24
- #NoRetro, 24, 113
- not preventing stakeholders from Daily Scrum attendance, 21–22
- not supporting struggling developers, 20–21, 113
- prisoners, 187–189
- Retrospective postponement, 23
- running Scrum events by allowing someone to speak, 7–8
- spying for management, 11–13
- team harmony over conflict resolution, 13–14
- Scrum Team
 - changing Sprint Goals in the middle of a Sprint, 265
 - daily status report, 143
 - delivering X instead of Y, 237–238
 - disrespecting fellow team members, 144
 - dogmatism, 238
 - excessive feedback, 144–146
 - extensive complaining, 186–187
 - failure to achieve Sprint Goal on a regular basis, 264–265
 - forecast imposed, 137–138
 - handing the Increment over to QA, 234–235
 - “hardening” Sprint, 114–115
 - ignoring the purpose of the Sprint Review, 160–161
 - immediate user value is required in all Increments, 239–240
 - inability to accommodate work unrelated to Sprint Goal, 266–267
 - irregular Sprint lengths, 134–135
 - lost Scrum Team, 143
 - no Definition of Done, 281–282
 - no routine, 143–144
 - no Sprint Goal, 262
 - no Sprint Review, 159
 - no standard for “ready”, 136–137
 - no visualization, 218–220
 - #NoAccountability, 184–185
 - #NoDocumentation, 186
 - not respecting Sprint boundaries, 265–266
 - overcrowding, 146
 - product Increment fails to achieve desired level of quality, 236–237
 - reintroducing waterfall planning through the backdoor, 253–254
 - rushed Retrospective, 180
 - skipping the Sprint Goal, 217
 - someone sings, 182–183
 - Sprint Goal imposed on Scrum Team, 262–263
 - Sprint Goal is confidential, 267
 - Sprint Goal is overly ambitious, 263–264
 - Sprint Zero, 115–117
 - sticking with obsolete Product Goals, 252–253
 - UNSMART action items, 184
 - variable Sprint length, 118
 - what improvement?, 185
- stakeholder
 - cherry-picking individual practices, 88–89
 - command & control by line managers, 151–152
 - communicating via body language, 154
 - emergency work, 104
 - flow disruption, 105
 - internal stakeholders do not attend, 169–170
 - lack of leadership support, 86–87
 - lack of transparency, 84–86
 - no customers present, 170–172
 - Product Backlog and refinement, 105
 - root causes, 84
 - Scrum on a tight budget, 90–92
 - Sprint Review as approval gate, 167–169
 - starting over again, 172–173
 - steering committee meetings, 94

- talkative, 152–154
 - telling people how to do things, 92–93
 - arbitrary deadlines, 62–63
 - ask-me-anything session, 366–367
 - autonomy, team, 98, 120–121, 188
- B**
- bad agile metrics, 322–323
 - Baldauf, C., 23
 - bias
 - confirmation, 100
 - hindsight, 100
 - self-serving, 100
 - Blame Game anti-pattern, 24, 189–191
 - board out of date anti-pattern, 59–60, 112
 - body language, 106, 154
 - Bonaker, Z., WADE-matrix, 336
 - budget, 98–99
 - bypassing the product owner anti-pattern, 104
- C**
- Celebrity Interview, 359
 - change, resistance to, 100–101. *See also* transformation
 - cherry-picking
 - individual practices anti-pattern, 88–89
 - Product Backlog items, 269–270
 - coaching
 - Developer, 74
 - one-to-one, 93
 - Scrum Master, 6–7, 14, 20
 - stakeholder, 169
 - collaboration, 93
 - in creating the Definition of Done, 283
 - between sales and Scrum teams, 102
 - command & control
 - culture, 117
 - by line managers anti-pattern, 151–152
 - communication, stakeholder
 - aggregate information in dashboards, 365
 - Daily Scrum, 364
 - Institute ambassadors, 366
 - release notes, 365–366
 - Sprint Reviews, 363–364
 - stakeholder Retrospective, 364
 - training classes, 366
 - confidentiality
 - Retrospective, 183
 - Sprint Goal, 267
 - confirmation bias, 100
 - conflict resolution, 13–14
 - continuous learning, Scrum Master, 6–7
 - Conversation Café, 352
 - CoP (community of practice), 324
 - overcoming resistance, 328
 - serving the community, 326
 - serving the organization, 327–328
 - copy & paste Product Owner, 32–33
 - creating
 - Definition of Done, 275–276, 334–335
 - Sprint Goals, 260–261
 - crowding-out collaboration anti-pattern, 36–37
 - customer/s
 - communication with Scrum Team, 95–96
 - feedback, 101, 229
 - cycle time, 57, 319–321
- D**
- DAD (Discovery and Action Dialog (DAD)), 333
 - daily newsletter, 367–368
 - Daily Scrum
 - communicating with stakeholders, 364
 - Developer anti-patterns
 - daily status report, 70–71
 - excessive feedback, 72
 - lost Scrum Team, 68–69
 - monologuing, 73, 147–149
 - no impediments, 73–74, 149
 - No Routine, 68
 - not supporting struggling developers, 73
 - planning meeting, 71, 146–147
 - problem-solving session, 71–72
 - unpreparedness, 69–70
 - Liberating Structures
 - 1–2-4-All, 333
 - Lean Coffee, 333
 - Product Owner anti-patterns, introducing additional work, 150

-
- purpose of, 142–143
 - remote, Liberating Structures
 - DAD (Discovery and Action Dialog (DAD)), 333
 - Min Specs, 332
 - Troika Consulting, 330–332
 - WINFY (What I Need From You), 333
 - Scrum Master anti-patterns
 - Daily Scrum enforcer, 150–151
 - not preventing stakeholders from attendance, 21–22
 - not supporting struggling developers, 20–21
 - Scrum Team anti-patterns
 - daily status report, 143
 - disrespecting fellow team members, 144
 - excessive feedback, 144–146
 - lost Scrum Team, 143
 - no routine, 143–144
 - overcrowding, 146
 - stakeholder anti-patterns, 105–106
 - command & control by line managers, 151–152
 - communicating via body language, 154
 - talkative stakeholders, 152–154
 - daily status report anti-pattern, 70–71, 143
 - deadlines, arbitrary, 62–63
 - death by PowerPoint anti-pattern, 75, 165
 - defining technical solutions anti-pattern, 18–19
 - Definition of Done, 62, 235
 - acceptance by the Product Owner, 290
 - anti-patterns, 277
 - creating, 275–276, 334–335
 - Developer anti-patterns
 - ignoring the Definition of Done, 278
 - multiple versions of the Definition of Done in a single product, 279
 - Increments, 235–236
 - layers, 275–276
 - organization-level anti-patterns
 - dogmatism, 289–290
 - no common ground in a product group, 289
 - no transparency, 290
 - purpose of, 273–275
 - Scrum Team anti-patterns
 - Definition of Done is too demanding, 287
 - Increment fails to achieve desired level of quality, 284–285
 - Increment fails to meet the Sprint Goal, 285–286
 - Increments released that fail to meet the Definition of Done, 283–284
 - lack of collaboration in creating Definition of Done, 283
 - not improving the Definition of Done, 288
 - playing it safe, 286
 - Scrum Team has no Definition of Done, 281–282
 - too much ambition, too soon, 287–288
 - Definition of Ready, 279–281
 - delaying Product owner anti-pattern, 44–45, 111
 - Developer/s
 - coaching, 74
 - Daily Scrum anti-patterns, 68
 - daily status report, 70–71
 - excessive feedback, 72
 - lost Scrum Team, 68–69
 - monologuing, 73, 147–149
 - no impediments, 73–74, 149
 - No Routine, 68
 - not supporting struggling developers, 73
 - planning meeting, 71, 146–147
 - problem-solving session, 71–72
 - unpreparedness, 69–70
 - Definition of Done anti-patterns
 - ignoring the Definition of Done, 278
 - multiple versions in a single product, 279
 - Product Backlog anti-patterns
 - no time for refinement, 77–78
 - submissive engineers, 79–81
 - Retrospective anti-pattern, dispensable buffer, 76–77
 - role and responsibilities, 56
 - self-management, 47
 - Sprint anti-patterns
 - board out of date, 59–60, 112
 - cutting corners to meet an increment’s deadline, 62–63
-

- gold-plating, 61–62, 113
 - lack of professionalism, 58–59, 112
 - no slack time, 64–66
 - no WiP limit, 57–58, 112
 - side gigs, 60–61, 112
 - technical debt, 63–64
 - Sprint Backlog anti-patterns
 - fixed scope, 221–222
 - planning too detailed, 220
 - shadow Sprint Backlog, 224–225
 - too much estimating, 220–221
 - urge to deliver the complete Sprint Backlog, 222–223
 - Sprint Goal anti-patterns
 - cherry-picking Product Backlog items, 269–270
 - no visualization of progress, 270
 - Sprint Planning anti-patterns
 - no capacity check, 66, 127–128
 - no slack time, 129
 - planning too detailed, 66, 129
 - team leads, 68, 129–132
 - technical debt, 128–129
 - too little planning, 66–67
 - too much estimating, 129
 - Sprint Review anti-patterns
 - death by PowerPoint, 75, 165
 - same faces again, 75, 166–167
 - showing off undone work, 75–76
 - side gigs, 167
 - dispensable buffer anti-pattern, 76–77
 - disrespecting fellow team members anti-pattern, 144
 - dogmatism, 3, 238, 289–290
- E**
- Ecocycle planning, 353
 - emergency work anti-pattern, 104, 121–122
 - excessive feedback anti-pattern, 72, 144–146
 - excluding stakeholders all the time anti-pattern, 26, 181–182
 - Experience Fishbowl, 359
 - extensive complaining anti-pattern, 186–187
- F**
- failure to achieve Sprint goal anti-pattern, 113
 - features, nonexistent, selling, 101–103
 - feedback
 - 360-degree, 7
 - customer, 101, 229
 - Developer, 149
 - excessive, 72
 - iterative, 104
 - late, 49
 - peer, 20, 149
 - peer recruiting team, 348–349
 - Product Owner, 44
 - sessions, 164
 - team, 72
 - fixed scope anti-pattern, 221–222
 - flawed tracking or useless metrics anti-pattern, 10–11
 - flow disruption anti-pattern, 17–18
 - Scrum Master, 113
 - stakeholder, 105
 - flow theory, 57
 - focus, 239
 - forecast imposed anti-pattern, 137–138
 - funding
 - agile transformation, 90–92
 - “My Budget” syndrome, 98–99
- G**
- gold-plating anti-pattern, 61–62, 113
 - governance, 291
 - Groundhog Day anti-pattern, 22–23
- H**
- “hardening” Sprint anti-pattern, 114–115
 - Hawthorne effect, 323
 - hindsight bias, 100
 - Hiring: 73 Scrum Master Interview Questions to Identify Suitable Candidates*, 344
- I**
- “I know it all” anti-pattern, 37
 - ignoring the Definition of Done anti-pattern, 278

-
- impromptu networking, 349
 - incentivizing innovation, 103–104
 - Increment/s, 229–230
 - Definition of Done, 283–284
 - purpose of, 230–231
 - Scrum Team anti-patterns
 - all Increments must contribute to accomplishing the goal, 238–239
 - delivering X instead of Y, 237–238
 - dogmatism, 238
 - failure to achieve desired level of quality, 284–285
 - handing it over to QA, 234–235
 - immediate user value is required in all Increments, 239–240
 - no decision on the release of increments, 231–232
 - only releasing to the complete user base, 233–234
 - postponed releases, 232–233
 - product Increment fails to achieve desired level of quality, 236–237
 - released Increments not meeting the Definition of Done, 235–236
 - stakeholder anti-patterns
 - rebuilding legacy applications, 241–242
 - roadmaps secrets, 240–241
 - inexperience, Scrum Master, 5
 - innovation, incentivizing, 103–104
 - Integrated~Autonomy, 359–360
 - internal stakeholders do not attend anti-pattern, 169–170
 - interviewing for Scrum Master position, 344–345
 - introducing additional work anti-pattern, 48–50, 150
 - INVEST principles, 205–206
 - involving the Scrum team anti-pattern, 39–40
 - irregular Sprint lengths anti-pattern, 134–135
 - IT management, Sprint anti-patterns
 - all hands to the pumps without Scrum, 96–98, 118
 - reassigning team members, 119–120
 - team autonomy undermined, 120–121
 - iterative feedback, 104
 - J-K**
 - Jeffries, R., 230
 - Jocham, R., 329
 - joint sales-Scrum charter, 102
 - Kanban board, 21, 72
 - Kniberg, H., 317
 - L**
 - lack of leadership support anti-pattern, 86–87
 - lack of professionalism anti-pattern, 58–59, 112
 - lack of transparency anti-pattern, 84–86
 - last minute changes anti-pattern, 133–134
 - last-minute Product Backlog anti-pattern, 39
 - late feedback, 49
 - layers, Definition of Done, 275–276
 - laziness, Scrum Master, 2–3
 - lead time, 319–321
 - leadership
 - Scrum Master, 2
 - senior management, 86–87
 - Lean Coffee, 333, 352, 366–367
 - learning, workshops, 91
 - legacy applications, rebuilding, 241–242
 - Liberating Structure/s, 351–352, 353–354, 360–361
 - 1–2-4-All, 333
 - DAD (Discovery and Action Dialog (DAD)), 333
 - Lean Coffee, 333
 - Min Specs, 332
 - remote Sprint Planning, 355–358
 - TRIZ, 8–23
 - Troika Consulting, 330–332
 - line managers, participation in Retrospective, 192–193
 - lost Scrum Team anti-pattern, 68–69, 143
 - M**
 - management
 - Product Backlog, 43
 - self-, 47
 - spying for, 11–13
 - media channels
 - daily newsletter, 367–368
 - product and engineering blog, 368
-

meetings, steering, 94
 mentoring. *See also* coaching
 Product Owner, 164
 Scrum Master, 6–7
 Meta-Retrospective, 335, 337–339
 how to run, 336
 Scrum Values exercise, 336–337
 metric/s, 10–11, 40, 58, 101. *See also* agile
 agile, 316
 bad, 322–323
 good, 319–322
 ugly, 323
 cycle time, 319–321
 lead time, 319–321
 qualitative, 319
 self-assessment, 317–319
 Min Specs, 332, 359
 missing acceptance criteria anti-pattern, 206
 monologuing anti-pattern, 73, 147–149
 Musk, E., “The Secret Tesla Motors Master Plan”, 85
 “My Budget” syndrome anti-pattern, 98–99

N

new kid on the block anti-pattern, 117–118
 no business objective anti-pattern, 271
 no capacity check anti-pattern, 66, 127–128
 no customers present anti-pattern, 170–172
 no impediments anti-pattern, 73–74, 149
 no improvements anti-pattern, 17, 139
 no psychological safety anti-pattern, 24–26
 no research anti-pattern, 38–39, 207
 no routine anti-pattern, 143–144
 no slack time anti-pattern, 129
 Developer, 64–66
 Scrum Master, 15
 no Sprint Review anti-pattern, 159
 no standard for “ready” anti-pattern, 136–137
 no suitable venue anti-pattern, 191–192
 no time for refinement anti-pattern, 77–78
 no WiP limit anti-pattern, 57–58, 112
 #NoAccountability anti-pattern, 184–185
 #NoDocumentation anti-pattern, 24, 186

#NoRetro anti-pattern, 24, 113, 179
 not supporting struggling developers anti-pattern, 73

O

one-to-one coaching, 93
 organization-level anti-patterns
 Definition of Done
 dogmatism, 289–290
 no common ground in a product group, 289
 no transparency, 290
 Product Goal predetermines the how, 254–255
 Retrospective
 line manager attendance, 192–193
 no suitable venue, 191–192
 reporting structures within the Scrum Team, 194
 Sprint Goal
 Scrum Team lacks focus, 268
 Sprint success defined by output, not outcome, 267–268
 stakeholders dictate the Product Goal, 255–256
 outdated Product Backlog items, 204
 output focus anti-pattern, 134
 overcrowding anti-pattern, 146
 overreaching Product Owner anti-pattern, 34–35
 oversized Product Backlog, 202–203

P

part-time owner anti-pattern, 31–32
 passive stakeholders, 173–174
 patience, 3, 7
 peer feedback, 20, 149
 peer recruiting, 342
 hiring a Scrum Master, 342–349
 People Operations and, 341–342
 peer review, 104
 People Operations, 341–342
 pilot program, 92
 planning meeting anti-pattern, 71, 146–147
 planning too detailed anti-pattern, 66, 129, 220

-
- Popper, K., 32
 - postponement
 - increment release, 232–233
 - Retrospective, 23, 180–181
 - practices. *See also* CoP (community of practice)
 - agile, cherry-picking, 88–89
 - remote Retrospective, 354
 - preparing, Sprint Planning, 126–127
 - prioritization by proxy anti-pattern, 35, 200–201
 - prisoners anti-pattern, 187–189
 - problem-solving session anti-pattern, 71–72
 - Product Backlog
 - actionable, 199
 - anti-patterns
 - 100% in advance, 203
 - everything is detailed and estimated, 205
 - INVEST?, 205–206
 - missing acceptance criteria, 206
 - outdated items, 204
 - oversized, 202–203
 - prioritization by proxy, 200–201
 - too detailed acceptance criteria, 206–207
 - Developer anti-patterns
 - no time for refinement, 77–78
 - submissive engineers, 79–81
 - management, 43
 - oversized, 35
 - Product Owner anti-patterns
 - last-minute Product Backlog, 207
 - no research, 207
 - storage for ideas, 207–208
 - technical debt, 209–211
 - purpose of, 198–200
 - refinement session, 33
 - Scrum Team anti-patterns
 - involving the Scrum team, 211–212
 - too much refinement, 212
 - stakeholder anti-patterns, 105
 - Product Goal/s, 329
 - anti-patterns
 - Product Goal is not transparent, 251
 - Product Owner forces the impossible upon the Scrum Team, 249–250
 - Product Owner singlehandedly creates Product Goals, 249
 - pursuing multiple Product Goals, 248
 - there is no Product Goal, 247–248
 - canvas, 330
 - purpose of, 246–247
 - Scrum Team anti-patterns
 - exclusionism, 251–252
 - reintroducing waterfall planning through the backdoor, 253–254
 - sticking with obsolete Product Goals, 252–253
 - stakeholder responsibility for creating and communicating, 255–256
 - Product Owner
 - Daily Scrum anti-patterns
 - assignments, 46–48
 - introducing additional work, 48–50, 150
 - Product Backlog and refinement anti-patterns
 - 100% in advance, 35
 - copy & paste Product Owner, 32–33
 - crowding-out collaboration, 36–37
 - “I know it all” Product Owner, 37
 - involving the Scrum team, 39–40
 - last-minute Product Backlog, 39, 207
 - no research, 38–39
 - overreaching owner, 34–35
 - oversized Product Backlog, 35
 - part-time owner, 31–32
 - prioritization by proxy, 35
 - storage for ideas, 207–208
 - technical debt, 209–211
 - providing feedback, 44
 - role and responsibilities, 30–31
 - Sprint anti-patterns
 - cancellations without consultation, 111
 - changing work items from the Sprint Backlog, 111
 - delaying, 111
 - no Sprint cancellation, 112
 - Sprint stuffing, 111
 - Sprint Backlog anti-patterns
 - additional work, 225–226
 - changing items during the Sprint, 226–227
-

- Sprint stuffing, 227–228
 - Sprint Goal anti-patterns, no business objective, 271
 - Sprint Planning anti-patterns
 - absent Product Owner, 42–43
 - delaying Product owner, 44–45
 - last minute changes, 133–134
 - last-minute changes, 40
 - no Sprint cancellation, 45–46
 - output focus, 40–42, 134
 - Scrum team has no Sprint goal, 40
 - Sprint cancellations without consultation, 45
 - unfinished business, 132–133
 - what are we fighting for, 132
 - Sprint Review anti-patterns
 - “acceptance” by the product owner, 51
 - “know-it-all” product owner loving their ideas, 51–52, 163–164
 - selfish owner, 50–51
 - product/s
 - discovery, 198
 - and engineering blog, 368
 - features, nonexistent, selling, 101–103
 - roadmap, 102, 240–241
 - psychological safety, 188, 191, 250
 - pull system, 59
 - purpose
 - of an agile community of practice, 325
 - of the Daily Scrum, 142–143
 - of Definition of Done, 273–275
 - of Increments, 230–231
 - of the Product Backlog, 198–200
 - of the Product Goal, 246–247
 - of the Retrospective, 178
 - of the Sprint, 109–110
 - of the Sprint Review, 158
 - pursuing multiple Product Goals, 248
- Q-R**
- qualitative metrics, 319
 - quality/quality assurance, 234–235, 236–237, 283. *See also* Definition of Done Increment, 284–285
 - radical candor, 148
 - reassigning team members anti-pattern, 119–120
 - rebuilding legacy applications, 241–242
 - refinement session, Product Backlog, 33
 - release notes, 365–366
 - remedy
 - absent Product Owner anti-pattern, 43
 - Developer anti-pattern
 - cherry-picking Product Backlog items, 269–270
 - cutting corners to meet an increment’s deadline, 63
 - daily status report, 71
 - death by PowerPoint, 165
 - dispensable buffer, 76–77
 - fixed scope, 222
 - gold-plating, 62
 - ignoring the Definition of Done, 278
 - lack of professionalism, 59
 - lost Scrum Team, 69
 - monologuing, 148–149
 - no impediments, 74
 - no slack time, 65–66
 - no visualization of Sprint Goal progress, 270
 - no WiP limit, 57–58
 - planning meeting, 147
 - planning too detailed, 220
 - problem-solving session, 72
 - same faces again, 166–167
 - shadow Sprint Backlog, 225
 - showing off undone work, 76
 - side gigs, 61
 - submissive engineers, 79–81
 - team leads, 131–132
 - too little planning, 67
 - too much estimating, 221
 - unpreparedness, 70
 - urge to deliver the complete Sprint Backlog, 223
 - “hardening” Sprint anti-pattern, 115
 - incentivized stakeholder anti-pattern
 - financial incentives to innovate, 103–104
 - “My Budget” syndrome, 99

- selling non-existent features, 102–103
- “We know what to build”, 101
- IT management anti-pattern
 - reassigning team members, 119–120
 - team autonomy undermined, 120–121
- new kid on the block anti-pattern, 118
- organization-level anti-pattern
 - line managers participation in the Retrospective, 193
 - no suitable venue, 191–192
 - Sprint success defined by output, not outcome, 267–268
- Product Backlog anti-pattern
 - 100% in advance, 203
 - missing acceptance criteria, 206
 - outdated items, 204
 - oversized, 202–203
 - prioritization by proxy, 201
- Product Goal anti-pattern
 - Product Goal is not transparent, 251
 - Product Owner forcing the impossible upon the Scrum Team, 250
 - Product Owner singlehandedly creates Product Goals, 249
 - pursuing multiple Product Goals, 248
 - there is no Product Goal, 247
- Product Owner anti-pattern
 - “acceptance”, 51
 - additional work, 226
 - assigning tasks to team members, 47–48
 - changing items during the Sprint, 227
 - copy & paste owner, 32–33
 - crowding-out collaboration, 36–37
 - delaying, 44
 - “I know it all”, 37, 164
 - introducing additional work, 49–50
 - involving the Scrum team, 39–40
 - last minute changes, 134
 - last-minute Product Backlog, 39
 - no business objective, 271
 - no research, 38–39
 - no Sprint cancellation, 46
 - output focus, 42
 - overreaching owner, 34–35
 - part-time owner, 31–32
 - selfish owner, 50–51
 - storage for ideas, 207–208
 - technical debt, 211
 - unfinished business, 132–133
- Scrum Master anti-pattern
 - administrative assistant, 9–10
 - “agile manager” behavior, 6–7
 - Blame Game, 190–191
 - Daily Scrum enforcer, 150–151
 - defining technical solutions, 19–20
 - flawed tracking or useless metrics, 11
 - flow disruption, 18
 - Groundhog Day, 23
 - irregular Sprint lengths, 135
 - no improvements, 17
 - No Psychological Safety, 24–26
 - no slack time, 16–17
 - not preventing stakeholders from Daily Scrum attendance, 22
 - not supporting struggling developers, 21
 - overly controlling Scrum Master, 8
 - prisoners, 188–189
 - Retrospective postponement, 23
 - spying for management, 12–13
 - team harmony over conflict resolution, 13–14
- Scrum Team anti-pattern
 - Definition of Done is too demanding, 287
 - delivering X instead of Y, 237–238
 - disrespecting fellow team members, 144
 - excessive feedback, 144–146
 - excluding stakeholders all the time, 181–182
 - extensive complaining, 187
 - failure to achieve Sprint Goal on a regular basis, 264–265
 - forecast imposed, 137–138
 - ignoring the purpose of the Sprint Review, 160–161
 - inability to accommodate work unrelated to Sprint Goal, 266–267
 - Increments released that fail to meet the Definition of Done, 284
 - new kid on the block, 118

- no decision on release of Increments, 232
- no routine, 143–144
- no Sprint Goal, 262
- no Sprint Review, 159
- no standard for “ready”, 136–137
- no visualization, 219–220
- #NoAccountability, 185
- #NoDocumentation, 186
- #NoRetro anti-pattern, 179
- not improving the Definition of Done, 288
- only releasing Increments to the complete user base, 233–234
- overcrowding, 146
- overly ambitious Sprint Goal, 264
- planning decisions made by definition of ready anti-pattern, 135
- postponed Increment release, 233
- Product Goal exclusionism, 252
- released Increments not meeting the Definition of Done, 236
- Retrospective postponement, 181
- rushed Retrospective, 180
- skipping the Sprint Goal, 217
- someone sings, 183
- Sprint Goal imposed on Scrum Team, 263
- Sprint task accounting, 162
- sticking with obsolete Product Goals, 253
- unauthorized Sprint Backlog additions, 218
- UNSMART action items, 184
- variable Sprint length anti-pattern, 118
- what improvement?, 185
- Sprint stuffing, 228
- stakeholder anti-pattern
 - command & control by line managers, 152
 - communicating via body language, 154
 - emergency work, 122
 - internal stakeholders do not attend, 169–170
 - lack of transparency, 85–86
 - no customers present, 171–172
 - passivity, 174
 - rebuilding legacy applications, 242
 - Scrum on a tight budget, 91–92
 - Sprint Review as approval gate, 168–169
 - starting over again, 172–173
 - talkative stakeholders, 153–154
 - talking to customers is off limits, 96
 - telling people how to do things, 93
- remote Daily Scrum, Liberating Structures 1–2-4-All, 333
- DAD (Discovery and Action Dialog (DAD)), 333
- Lean Coffee, 333
- Min Specs, 332
- Troika Consulting, 330–332
- WINFY (What I Need From You), 333
- remote Retrospective
 - applications for running, 354
 - closing, 353
 - deciding what to do, 352–353
 - gathering data, 350–351
 - generate insight from collected data, 351–352
 - good practices, 354
 - Liberating Structures, 353–354
 - setting the stage, 349–350
- remote Sprint Planning, Liberating Structures, 355–358
- remote Sprint Review
 - closing, 360
 - discussing accomplishments, 358–359
 - Liberating Structures, 358–361
- reporting
 - self, 12
 - structure within a Scrum Team, 194
- Retromat.org, 23
- Retrospective
 - Developer anti-patterns, dispensable buffer, 76–77
 - Meta-, 337–339
 - how to run, 336
 - Scrum Values exercise, 336–337
 - organization-level anti-patterns
 - line manager attendance, 192–193
 - no suitable venue, 191–192
 - reporting structures within the Scrum Team, 194
 - purpose of, 178
 - remote
 - applications for running, 354

- closing, 353
 - deciding what to do, 352–353
 - gathering data, 350–351
 - generate insight from collected data, 351–352
 - good practices, 354
 - Liberating Structures, 353–354
 - setting the stage, 349–350
 - Scrum Master anti-patterns
 - Blame Game, 24, 189–191
 - Groundhog Day, 22–23
 - No Psychological Safety, 24–26
 - #NoDocumentation, 24
 - postponement, 23
 - prisoners, 187–189
 - Scrum Team anti-patterns
 - excluding stakeholders all the time, 181–182
 - extensive complaining, 186–187
 - #NoAccountability, 184
 - #NoDocumentation, 186
 - #NoRetro, 179
 - postponement, 180–181
 - rushed Retrospective, 180
 - someone sings, 182–183
 - UNSMART action items, 184
 - what improvement?, 185
 - stakeholder, 364
 - stakeholder anti-patterns, 107
 - role and responsibilities
 - Developer, 56
 - Product Owner, 30–31
 - rushed Retrospective anti-pattern, 180
- S**
- same faces again anti-pattern, 75, 166–167
 - Scrum, abandoning, 96–98
 - Scrum Master
 - 43 ways to sabotage, 295–302
 - “agile manager” behavior, 6–7
 - anti-patterns
 - administrative assistant, 8–10
 - “agile manager” behavior, 5–6
 - flawed tracking or useless metrics, 10–11
 - overly-controlling, 7–8
 - spying for management, 11–13
 - team harmony over conflict resolution, 13–14
 - unrefined work items, 15–17
 - Daily Scrum anti-patterns
 - Daily Scrum enforcer, 150–151
 - not preventing stakeholders from Daily Scrum attendance, 21–22
 - not supporting struggling developers, 20–21
 - hiring, 342–349
 - mentoring and coaching, 6–7, 20
 - reasons for leaving the path, 4
 - dogmatism, 3
 - frustration, 3–4
 - impatience, 3
 - indifference, 3
 - inexperience, 5
 - laziness, 2–3
 - opportunism, 3
 - Retrospective anti-patterns
 - Blame Game, 24, 189–191
 - excluding stakeholders all the time, 26
 - Groundhog Day, 22–23
 - No Psychological Safety, 24–26
 - #NoDocumentation, 24
 - postponement, 23
 - prisoners, 187–189
 - Sprint anti-patterns
 - defining technical solutions, 18–19
 - flow disruption, 17–18, 113
 - #NoRetro, 113
 - not supporting struggling developers, 113
 - Sprint Planning anti-patterns
 - no improvements, 17, 139
 - no slack time, 15
 - Scrum on a tight budget anti-pattern, 90–92
 - Scrum Team
 - autonomy, 188
 - communicating with customers, 95–96
 - Daily Scrum anti-patterns
 - daily status report, 143
 - disrespecting fellow team members, 144

- excessive feedback, 144–146
- lost Scrum Team, 143
- no routine, 143–144
- overcrowding, 146
- Definition of Done anti-patterns
 - Definition of Done is too demanding, 287
 - Increment fails to achieve desired level of quality, 284–285
 - Increment fails to meet the Sprint Goal, 285–286
 - Increments released that fail to meet the Definition of Done, 283–284
 - lack of collaboration in creating
 - Definition of Done, 283
 - not improving the Definition of Done, 288
 - playing it safe, 286
 - Scrum Team has no Definition of Done, 281–282
 - too much ambition, too soon, 287–288
- Increment anti-patterns
 - all Increments must contribute to accomplishing the goal, 238–239
 - delivering X instead of Y, 237–238
 - handing the Increment over to QA, 234–235
 - immediate user value is required in all Increments, 239–240
 - no decision on their release, 231–232
 - only releasing to the complete user base, 233–234
 - postponed releases, 232–233
 - product Increment fails to achieve desired level of quality, 236–237
 - released Increments not meeting the Definition of Done, 235–236
- leads, 129–132
- output maximization, 40–42
- Product Goal anti-patterns
 - exclusionism, 251–252
 - reintroducing waterfall planning through the backdoor, 253–254
 - sticking with obsolete Product Goals, 252–253
- reassigning members, 119–120
- reporting structure, 194
- Retrospective anti-patterns
 - excluding stakeholders all the time, 181–182
 - extensive complaining, 186–187
 - #NoAccountability, 184–185
 - #NoDocumentation, 186
 - #NoRetro, 179
 - postponement, 180–181
 - rushed Retrospective, 180
 - someone sings, 182–183
 - UNSMART action items, 184
 - what improvement?, 185
- Sprint anti-patterns
 - failure to achieve Sprint goal, 113
 - “hardening” Sprint, 114–115
 - new kid on the block, 117–118
 - Sprint Zero, 115–117
 - variable Sprint length, 118
- Sprint Backlog anti-patterns
 - no visualization, 218–220
 - not having a Sprint Backlog by skipping the Sprint Goal, 217
 - unauthorized additions, 218
- Sprint Goal anti-patterns
 - changing Sprint Goals in the middle of a Sprint, 265
 - failure to achieve Sprint Goal on a regular basis, 264–265
 - inability to accommodate work unrelated to Sprint Goal, 266–267
 - not respecting Sprint boundaries, 265–266
 - Scrum Team has no Sprint Goal, 262
 - Scrum Team lacks focus, 268
 - Sprint Goal imposed on Scrum Team, 262–263
 - Sprint Goal is overly ambitious, 263–264
- Sprint Plan, 67
- Sprint Planning anti-patterns
 - forecast imposed, 137–138
 - irregular Sprint lengths, 134–135
 - no standard for “ready”, 136–137
 - planning decisions made by definition of ready, 135
- Sprint Review anti-patterns
 - ignoring the purpose of the Sprint Review, 160–161

- no Sprint Review, 159
- Sprint task accounting, 162
- trial day, 346–348
- self
 - assessment, 317–319
 - management, 47
 - reflection, 20
 - reporting, 12
 - serving bias, 100
- selling non-existent features anti-pattern, 101–103
- senior management, leadership support, 86–87
- shadow Sprint Backlog anti-pattern, 224–225
- showing off undone work anti-pattern, 75–76
- side gigs anti-pattern, 60–61, 112, 167
- Simple Ethnography, 358
- slack time, benefits, 65–66. *See also* no slack time anti-pattern
- SMART framework, 184
- someone sings anti-pattern, 182–183
- Sprint
 - board, 58, 60
 - Developer anti-patterns
 - board out of date, 59–60, 112
 - cutting corners to meet an increment’s deadline, 62–63
 - gold-plating, 61–62, 113
 - lack of professionalism, 58–59, 112
 - no slack time, 64–66
 - no WiP limit, 57–58, 112
 - side gigs, 60–61, 112
 - technical debt, 63–64
 - IT management anti-patterns
 - all hands to the pumps without Scrum, 118
 - reassigning team members, 119–120
 - team autonomy undermined, 120–121
 - Product Owner anti-patterns
 - cancellations without consultation, 111
 - changing work items from the Sprint Backlog, 111
 - delaying, 111
 - no Sprint cancellation, 112
 - purpose of, 109–110
 - Scrum Master anti-patterns
 - defining technical solutions, 18–19
 - flow disruption, 17–18, 113
 - #NoRetro, 113
 - not supporting struggling developers, 113
 - Scrum Team anti-patterns
 - failure to achieve Sprint goal, 113
 - “hardening” Sprint, 114–115
 - new kid on the block, 117–118
 - Sprint Zero, 115–117
 - variable Sprint length, 118
 - stakeholder anti-patterns
 - bypassing the product owner, 104
 - emergency work, 104
 - flow disruption, 105
 - stuffing, 111, 227–228
 - Sprint Backlog, 215–216
 - Developer anti-patterns
 - fixed scope, 221–222
 - planning too detailed, 220
 - shadow Sprint Backlog, 224–225
 - too much estimating, 220–221
 - urge to deliver the complete Sprint Backlog, 222–223
 - Product Owner anti-pattern
 - changing items during the Sprint, 226–227
 - Sprint stuffing, 227–228
 - Product Owner anti-patterns, additional work, 225–226
 - Scrum Team anti-patterns
 - no visualization, 218–220
 - not having a Sprint Backlog by skipping the Sprint Goal, 217
 - unauthorized additions to the Sprint Backlog, 218
 - Sprint Goal, 259–260
 - creating, 260–261
 - Developer anti-patterns
 - cherry-picking Product Backlog items, 269–270
 - no visualization of progress, 270
 - organization-level anti-patterns
 - Scrum Team lacks focus, 268
 - Sprint success defined by output, not outcome, 267–268

- Product Owner anti-pattern, no business objective, 271
- Scrum Team anti-patterns
 - changing Sprint Goals in the middle of a Sprint, 265
 - failure to achieve Sprint Goal on a regular basis, 264–265
 - not respecting Sprint boundaries, 265–266
 - Scrum Team cannot accommodate work unrelated to Sprint Goal, 266–267
 - Scrum Team has no Sprint Goal, 262
 - Sprint Goal imposed on Scrum Team, 262–263
 - Sprint Goal is confidential, 267
 - Sprint Goal is overly ambitious, 263–264
- Sprint Planning
 - Developer anti-patterns
 - no capacity check, 66, 127–128
 - no slack time, 129
 - planning too detailed, 66, 129
 - team leads, 68, 129–132
 - technical debt, 128–129
 - too little planning, 66–67
 - too much estimating, 129
 - preparing, 126–127
 - Product Owner anti-patterns
 - absent Product Owner, 42–43
 - cancellations without consultation anti-pattern, 45
 - delaying, 44–45
 - last-minute changes, 40, 133–134
 - no Sprint cancellation, 45–46
 - output focus, 40–42, 134
 - Scrum team has no Sprint goal, 40
 - unfinished business, 132–133
 - what are we fighting for, 132
 - remote, Liberating Structures, 355–358
 - Scrum Master anti-patterns
 - no improvements, 17, 139
 - no slack time, 15
 - unrefined work items, 15–17
 - Scrum Team anti-patterns
 - forecast imposed, 137–138
 - irregular Sprint lengths, 134–135
 - no standard for “ready”, 136–137
 - planning decisions made by definition of ready, 135
 - stakeholder anti-patterns, 106
 - Sprint Review
 - communicating with stakeholders, 363–364
 - Developer anti-patterns
 - death by PowerPoint, 75, 165
 - same faces again, 75, 166–167
 - showing off undone work, 75–76
 - side gigs, 167
 - Product Owner anti-patterns
 - “acceptance” by the product owner, 51
 - “know-it-all” product owner loving their ideas, 51–52, 163–164
 - selfish owner, 50–51
 - purpose of, 158
 - remote
 - closing, 360
 - discussing accomplishments, 358–359
 - Liberating Structures, 360–361
 - Scrum Team anti-patterns
 - ignoring the purpose of the Sprint Review, 160–161
 - no Sprint Review, 159
 - Sprint task accounting, 162
 - stakeholder anti-patterns, 106–107
 - bypassing the Product Owner, 122–123
 - emergency work, 121–122
 - internal stakeholders do not attend, 169–170
 - no customers present, 170–172
 - Sprint Review as approval gate, 167–169
 - starting over again, 172–173
 - Sprint Zero anti-pattern, 115–117
 - spying for management, 11–13
 - stakeholder/s
 - anti-patterns
 - cherry-picking individual practices, 88–89
 - financial incentives to innovate, 103–104
 - lack of leadership support, 86–87
 - lack of transparency, 84–86
 - “My Budget” syndrome, 98–99
 - root causes, 84
 - Scrum on a tight budget, 90–92

- selling non-existent features, 101–103
 - steering committee meetings, 94
 - talking to customers is off limits, 95–96
 - telling people how to do things, 92–93
 - “We know what to build”, 100–101
 - coaching, 169
 - collaboration with Scrum Team, 93
 - communication channels, 361–362
 - aggregate information in dashboards, 365
 - Daily Scrum, 364
 - Institute ambassadors, 366
 - release notes, 365–366
 - Sprint Review, 363–364
 - training classes, 366
 - Daily Scrum anti-patterns, 105–106
 - command & control by line managers, 151–152
 - communicating via body language, 154
 - talkative stakeholders, 152–154
 - excluding, 26, 181–182
 - Increment anti-patterns
 - rebuilding legacy applications, 241–242
 - roadmaps secrets, 240–241
 - passive, 173–174
 - Product Backlog anti-patterns, 105
 - responsibility for dictating the Product Goal, 255–256
 - Retrospective, 364
 - Retrospective anti-patterns, 107
 - Sprint anti-patterns
 - bypassing the product owner, 104
 - emergency work, 104
 - flow disruption, 105
 - Sprint Planning anti-patterns, 106
 - Sprint Review anti-patterns, 106–107
 - bypassing the Product Owner, 122–123
 - emergency work, 121–122
 - internal stakeholders do not attend, 169–170
 - no customers present, 170–172
 - Sprint Review as approval gate, 167–169
 - starting over again, 172–173
 - trust-building, 63
 - workshops, 101
 - steering committee meetings, 94
 - storage for ideas anti-pattern, 207–208
 - submissive engineers anti-pattern, 79–81
- T**
- talkative stakeholders anti-pattern, 152–154
 - talking token, 149
 - team/s
 - autonomy, 98, 120–121
 - conflict resolution, 13–14
 - feedback, 72
 - review, 104
 - trust, 65
 - technical debt, 128–129, 233
 - Developer anti-pattern, 63–64
 - Product Owner anti-pattern, 209–211
 - telling people how to do things anti-pattern, 92–93
 - too little planning anti-pattern, 66–67
 - too much estimating anti-pattern, 129, 220–221
 - too much refinement anti-pattern, 212
 - training classes, 366
 - transformation, 89–90
 - funding, 90–92
 - leadership support, 86–87
 - pilot program, 92
 - transparency, 84–85. *See also* lack of transparency anti-pattern
 - Definition of Done, 290
 - Product Goal, 251
 - Sprint Goal, 267
 - TRIZ, 8–23
 - Troika Consulting, 330–332
 - trust, 12, 118
 - building, 74
 - stakeholder, 63
 - team, 65, 93
- U**
- ugly agile metrics, 323
 - unauthorized additions to the Sprint Backlog, 218
 - unfinished business anti-pattern, 132–133
 - unpreparedness anti-pattern, 69–70

unrefined work items anti-pattern, 15–17
UNSMART action items, 184

V

value, creation, 31
variable Sprint length anti-pattern, 118
Vegas rule, 182
visualization, 86
 Sprint Backlog, 218–220
 Sprint boards, 58
 Sprint Goal progress, 270

W-X-Y-Z

Wake, B., INVEST principles, 205–206
waterfall planning, 253–254
“We know what to build” anti-pattern,
 100–101
what are we fighting for anti-pattern, 132
what improvement? anti-pattern, 185
WINFY (What I Need From You), 333
WiP (work-in-progress), 57
workshops, 91, 101