Mastering Work Intake

From Chaos to Predictable Delivery

Thomas M. Cagley, Jr.

SAFe® Practice Consultant (SPC)

Certified Scrum Master (CSM)

and

Jeremy Willets

SAFe® Practice Consultant (SPC)

J.ROSS PUBLISHING

ISBN-13: 978-1-60427-200-0

Printed and bound in the U.S.A. Printed on acid-free paper.

10 9 8 7 6 5 4 3 2 1

The Library of Congress Cataloging-in-Publication Data can be found on the WAV section of the publisher's website at www.jrosspub.com/wav.

Direct all inquiries to J. Ross Publishing, Inc., 151 N. Nob Hill Rd., Suite 476, Plantation, FL 33324.

Phone: (954) 727-9333
Fax: (561) 892-0700
Web: www.jrosspub.com

To my wife, Barb:

Thank you for putting up with me and my obsession with trying to make the world a better place. Oh, and the dad jokes.

—Tom

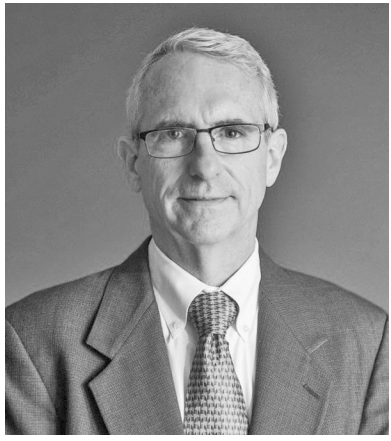
To Corinne, Cecilia, and Cordelia:

Thank you for your never-ending support on this journey.

—Jeremy (Dad)

CONTENTS

ABOUT THE AUTHORS



Thomas M. Cagley, Jr., is a consultant, speaker, author, coach, and agile guide who leads organizations and teams to unlock their inherent greatness. He has developed estimation models and has supported organizations developing classic and agile estimates. Tom helps teams and organizations improve cycle time, productivity, quality, morale, and customer satisfaction, and then prove it.

Tom is an internationally respected blogger and podcaster for over 11 years, focusing on software processes and measurement. His blog entries and podcasts have been listened to or read over a million times. He coauthored *Mastering Software Project Management: Best Practices, Tools and Techniques* with Murali K. Chemuturi. Tom penned the "Agile Estimation Using Functional Metrics" chapter in *The IFPUG Guide to IT and Software Measurement*. His certifications include SAFe® Practice Consultant (SPC) and Certified Scrum Master (CSM).

More information about Tom can be found at tomcagley.com.

Jeremy Willets is a coach, speaker, and author who has spent the last decade working with people and teams to achieve greatness in the workplace. He started out as a technical writer on a Scrum team and quickly fell in love with Scrum and the Agile Manifesto values and principles. Since then, he's served thriving organizations as a Scrum Master, agile coach, senior agile coach, release train engineer, people manager, and mentor.

Jeremy has spoken at conferences throughout the midwestern United States. He's an avid Substack blogger and music maker. He holds the SAFe® Practice Consultant (SPC) certification.

More information about Jeremy can be found at jeremywillets.com.

PREFACE

WHY WORK INTAKE?

Regardless of whether you're creating, enhancing, or maintaining products, work intake is a challenge you will deal with. Those who choose to ignore it run the risk of a semipermanent state of thrashing. This is the same as chasing the proverbial new shiny object instead of doing the thing that customers will actually pay you for. Mastering work intake could very well make or break your career and your company. It is the only path from chaos to predictability.

When we thought about the idea of writing a book on this topic, we did our research. It was shocking that such a small number of texts deal with this topic. Many focus on what to do when the work is in progress or the mechanics of work prioritization. But very few focus on the full pipeline that work follows as it enters and exits your system. They also ignore the fact that different types of work enter at different levels and times. The unwillingness to tackle work intake is one of our industry's dirty little secrets.

We have worked with great engineering teams and organizations that experienced work intake challenges. They ended up either building the wrong thing or building the right thing, but at the wrong time. In all cases, there were significant ramifications—it was chaos. Across the industry, work intake is still a problem. The reasons for this are many, and we'll explore many of those antipatterns in this book.

Here is a final word as to why we chose this topic: poor work intake frustrates everyone—including customers and coworkers. We hope this book will help everyone in the world of product development feel a lot less frustration and at least a little more satisfaction with their work.

THIS BOOK IS FOR YOU

At one time or another, work intake has been a common challenge for every software team that we have worked with. Here are our thoughts on how this work will be useful to you:

- **Agile coach/Scrum Master:** We have spent most of our careers in these roles, and this material is definitely written from that perspective. Scrum Masters and agile coaches help teams as they face decisions concerning what to work on. These are often difficult questions. Expect to see examples of antipatterns throughout, as well as the steps we took to resolve them.
- **Product owner/product manager:** Prioritization of work is an important work intake concept. It is fundamental to the product owner and product manager roles. We will explore many strategies that you can use when prioritizing. These strategies will be especially useful when everyone wants a different thing or if you are working in an organization without the capacity to execute the main business needs.
- **Portfolio epic sponsor:** If this is your role, you're making large commitments on behalf of your company. We will explore how to make these commitments consistent and repeatable.
- **Portfolio manager:** This role facilitates the refinement and prioritization of higher level backlog items. The goal is to link them up to business objectives. There must be a relationship between organizational goals and each team's stories. If there is no relationship, strategy and innovation will suffer.
- **Project manager/program manager:** These roles care about whether teams or programs are working on the right things

at the right time. Executing work in flight is part of mastering work intake.

- **Team member:** The team that is creating the product plays a critical role in the development of the product. These roles are more than coming in and writing or testing code for eight hours every weekday. Being a team member demands that you remain cognizant of priorities as you are working. You must also exercise discipline when work enters the team. How can team members influence the trajectory of the work they do, as well as the organization? Mastering work intake involves recognizing that it's easy to say "yes" and much harder to say "no."
- **Manager/team lead:** People leaders at the team level have a part to play in the work intake equation. You need to know what your team is executing on, what work is next, and the skill sets required to do the work. The work in front of your team might be a perfect match for their skill sets right now, but there are always opportunities for professional development.
- **Director/senior manager:** People leaders at a higher level have an economy of scale to worry about when it comes to work intake. Engineering organization leaders want to understand how the business is using engineering capacity. These leaders want their people to execute a company-wide strategy and product vision. Visualization and measurement are crucial work intake concepts to understand and practice.
- **Executive/sponsor (C-level, VP-level):** People at this level are making large commitments on behalf of their organizations. For example, a conversation with a customer at a trade show may lead to an idea for a product enhancement. It's great to have the organizational clout to be able to fast-track requests. But what's the actual cost of expediting something? There are ramifications for accepting new work into any system. Just because you have a big stick doesn't mean you should swing it all the time.
- **Customer-facing internal stakeholder:** People in sales, tech support, etc., often spend time engaging with customers. These roles have first-hand experience and visibility into customer

challenges and requests. They have better insight into how the products are being used than the teams who are developing them. When it is time to prioritize work, how can you use this knowledge to make sure that the rest of the company listens to you?

- **Customer:** Last, but never least, we all use software products daily. We are all customers and we all have ideas on how to improve the products that we use. What happens, though, when product development happens without customer input? We'll discuss strategies you can use as a customer to get your voice heard.

This list of roles isn't comprehensive, but we are hopeful that it gives you a sense of the value of this material, regardless of which type of role you occupy. We had a great time putting this book together. But we know that this book won't be successful unless it provides significant value to readers and practitioners.

IS THIS AN AGILE BOOK?

We've spent many years working with agile teams and organizations. We've also spent a lot of time working in contexts that aren't using agile. While we wrote with an agile lens, this book applies to most business operating systems. Regardless of how a business creates and supports products, work is going to enter the system. If you don't get this right, you won't be in business for very long.

Traditional project management methods are still thriving in many firms. If this is your company, congratulations! You have a huge opportunity to impact how your company creates and supports products. We're confident that you'll find ideas in this book to try out. And remember, *being agile* is not the goal, per se; the goal is flexibility in how you bring products to market. Mastering work intake is a main tenet in reclaiming flexibility for your company.

HOW TO USE THIS BOOK

This book combines theory and experience to diagnose and solve work intake problems. All chapters include *Learning Objectives* and *End-of-Chapter Questions*. Many chapters include a variety of callouts:

- **You Asked:** answers to common questions. The explanation picks up where the theory in the text ends and adds greater context so you can apply the new ideas.
- **Experience Report:** an example of a real-world situation. Drawn from our experiences or interviews with practitioners to highlight the topic.
- **Experiment:** example of how to diagnose an issue and to help you address the topic. Experiments include proposed problem statements, hypotheses, how to validate your outcomes, and an example.

FINAL THOUGHTS ON WORK INTAKE AND AGILE

The *Agile Manifesto* was a rebellious act. It attacked the outdated methods used to create software products at the time. Waterfall was chief among them. The authors of this book have very different pedigrees. Jeremy never had the pleasure of working in a waterfall environment, while Tom has. Regardless, both can picture work intake in that context. The picture in Jeremy's mind goes something like this:

> *Your team is working on a project with different stage-gate phases. The developers finished writing the code based on a 100-page specification document. Everything is committed to a customer by a certain date. Everyone has signed off on the document, even though most only skimmed it. The testers have begun the testing phase. You expect that the developers will spend the next two months fixing the bugs that the testers find. Then, your team will prepare the project for deployment.*

> *Yesterday, a product manager asked you whether the team could do a new piece of* high-priority *work. They said it's going to be "pretty big." You have a choice to make. You can interrupt the team with this new work or wait until they are completely done. Your choice is disruption or a long lead time for a new high-priority request. You seem to have very little flexibility.*

When we discussed this scenario, Tom shook his head and laughed. Phrases like "We will get to that in phase two" and "Who's in for working all weekend?" were not jokes. They were—and are—the reality of the situation facing development teams.

Unfortunately, this reality rings true today. Work entering during a sprint or between planning events is common. When new work appears, saying "no" seems counter to continuous value delivery. Saying "yes" to new work is exactly the opposite, though. It crushes the continuous delivery of anything and is antithetical to being agile. Flexibility on behalf of customers is important, but it only matters if you actually deliver the right value when needed. Mastering work intake is the capability that will actually deliver what is needed when it is needed.

ACKNOWLEDGMENTS

This book is the result of nearly three years' worth of iterative and incremental progress. It is the result of hundreds of hours spent writing and editing in the early hours of the morning, in the lobby while waiting for kids to finish lessons, or at the end of a long day when everyone else at home was already asleep. This work is also the result of hundreds of hours of collaborative Zoom calls with each other.

We could not have produced this work on our own. We would like to thank Barb Cagley, who reformatted all the illustrations in the home stretch of creating this work. We would also like to thank the people who answered our call to review some (or all) of this work: Linda Podder, Benjamin Woznicki, Melissa Greller, Jillian Testa, Ryan Sylvester, and David Herron. We are grateful for the time they dedicated to giving us feedback. We are even more grateful to call them friends.

We would like to thank Stephen Buda at J. Ross Publishing. His expert guidance throughout the publishing process was greatly appreciated.

Finally, we'd like to acknowledge the people whose work has inspired us to write on this topic: Jim Benson, Daniel Vacanti, Donald Reinertsen, Johanna Rothman, Mik Kersten, and W. Edwards Deming. Their work has changed us—and it has changed the world.

INTRODUCTION TO TERMS

We will define lots of terminology throughout this book, but here are a few general terms that will not be explained in the text:

- *Agile*: the agile software development movement, which began in the early 2000s.
- *Agile Manifesto*: the document created by a conclave of software industry veterans at a ski resort in Utah. You can read the manifesto at https://agilemanifesto.org.
- *Scrum*: the framework for product development that was popularized by Jeff Sutherland and Ken Schwaber. You can read the Scrum Guide at https://www.scrum.org.
- *Kanban*: the lean method of visualizing work and managing work in progress. It was popularized in the world of software by people like David Anderson and Jim Benson.
- *Scaled Agile Framework* (SAFe®): the Scaled Agile Framework bills itself as the leading system for scaling agile. You can read about the framework at https://www.scaledagileframework.com/.
- *Product development*: the act of developing products. In this work, we will focus on product development in the software realm.
- *Extreme programming* (XP): Extreme programming is one of several popular agile processes. You can read more at http://www.extremeprogramming.org.

- *Program*: the Project Management Institute (PMI) defines a program as "a group of related projects managed in a coordinated way to obtain benefits and control not available from managing them individually."* We will use this definition throughout this text. For more information about how PMI defines a *program* and distinguishes it from a *project*, see https://www.pmi.org/learning/library/understanding-difference-programs-versus-projects-6896.

* "Program Management." Project Management Institute, 7/22/2023, www.pmi.org/learning/featured-topics/program.

This book has free material available for download from the Web Added Value™ resource center at *www.jrosspub.com*

At J. Ross Publishing we are committed to providing today's professional with practical, hands-on tools that enhance the learning experience and give readers an opportunity to apply what they have learned. That is why we offer free ancillary materials available for download on this book and all participating Web Added Value™ publications. These online resources may include interactive versions of the material that appears in the book or supplemental templates, worksheets, models, plans, case studies, proposals, spreadsheets and assessment tools, among other things. Whenever you see the WAV™ symbol in any of our publications, it means bonus materials accompany the book and are available from the Web Added Value Download Resource Center at www.jrosspub.com.

Downloads for *Mastering Work Intake* include facilitated exercises to help you formulate ideas to address work intake challenges in your context and instructional material for classroom use (lecture slides, exercise solutions, etc.).

Section One

Section One

Work Intake

Section Contents

SECTION INTRODUCTION

The way in which work gets to an organization or team isn't often discussed in polite company. Have you ever heard of a conference on work intake? Or even a track at a conference? The topic is either viewed as "boring," "the way it has to be," or so far "out of control" that it's not worth discussing. Those are excuses and rationalizations. When work intake comes up, it's generally couched in soft terms. These terms give the work intake process a nice, dull appearance. They obscure the importance of mastering work intake. It's the largest determinant of whether an organization or team can get work done.

We previewed the content of this book to a group of trusted colleagues and friends. At the time, we were using *work entry* to describe work intake. One of the feedback comments we received was, "Don't you mean work intake?" The short answer was, "No." The path to that simple answer is a bit longer.

Understanding why we thought *entry* was the correct term for this book is important. *How work gets done* is a building block toward understanding agile, lean, or waterfall principles. True agile and lean teams pull prioritized work based on capacity and capabilities. They do this rather than the alternative, which is having work pushed on them. More traditional approaches, such as those embodying the Capability Maturity Model Integration best practices, are more push-oriented, although they do consider team and organizational capacity and capabilities. Regardless of approach, process and practice are only somewhat related.

Intake refers to both *how* the work is acquired and the *amount of work* taken into a team or organization. The word *take* implies that the team or organization has the power to accept (or pull) the work. The idea of intake requires teams to have a real choice. Pulling work based on capacity that is determined via planning is an example of choice. Other examples of choice include planning work based on a Monte Carlo simulation or yesterday's weather. When teams live by lean and agile principles, their process for getting work is *intake*.

Embracing the more abrasive term *work entry* was a journey. It crystallized while Tom was re-reading the book *Coaching Agile Teams* by Lyssa Adkins. In Chapter 3 of that seminal work, Adkins discusses the impact of violent language. For coaches, language correlates to impact. In our estimation, *intake* is a softer, more neutral word than *entry*. Our intent has never been to write about this topic in a neutral manner. This is because that tone rarely matches our experiences with the topic.

Entry is forceful and aggressive; it implies pushed work. Regardless of professed philosophy, when work is *pushed*, the term *intake* sugarcoats problems. The word *entry* does not. Pushed or uncontrolled acceptance of work is not agile. If a team or organization cannot control how they get work, then how they get work is *entry*.

The goal of this book is to change how organizations and teams get work so that *work intake* makes more sense. We ended up using the word *intake* because that's what we aspire to see in the organizations we work with. But every time we see bad stuff happen, we can't help but think of it as *work entry*.

1

WHAT IS WORK INTAKE?

Learning Objectives—by the end of this chapter, you will be able to:

- Define work intake
- Give a simple example of a work intake process
- Differentiate between *work entry* and *work intake*

Every entity has a process to decide what to do, how to spend its resources, and how to deploy people—even if that process is to drop everything any time someone asks. Those processes are considered *work intake*. Work intake is not the sexiest topic. There's an old cartoon that depicts a manager standing in front of a room of coders. The manager states, "You keep coding, I'll go get some requirements." This shows how disrespected work intake is as a process. Unless a team has mature (or maturing) processes for working, it is generally not understood. In several recent conversations, we asked individuals and teams how they get the work they were assigned to do. "We triage the work with our product owner," was one end of the spectrum. "The phone rings and somebody's manager tells me what to do," was the other end. We were also accused of being killjoys for bringing up the topic during happy hour.

Most people simply come to work to deliver, so discovering how work enters an organization or team is almost always a touchy subject. Talking about work intake feels like overhead because it happens before you can do the *real* work. When asked about work intake, we are generally given a PowerPoint presentation or a process document to read. Regardless of the processes that are diagrammed, they are never right. Work intake mimics a water leak in a building; it is almost as if it were alive—or had a mind of its own. Work seeks a way to get started. Sometimes it even gets all the way to done. It can enter through the *front door*—tied to strategic plans in the enterprise portfolio. It can also find a *back door* to a team or individual and get done *off the books* (read: unsanctioned). Every level of an organization has a work intake process. This process cascades work to lower levels and informs the levels above it. The problem is that every level also has secret paths to someone who will choose to accept the work. Every one of these paths defines work intake. Every last piece of *started* work has a permanent impact on the trajectory of a firm, product, and team.

A SIMPLE EXAMPLE OF WORK INTAKE IN SCRUM

Scrum is "a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems."[1] In Scrum, work intake follows a simple flow of steps:

1. People write stories, ideas, or requirements of varying quality
2. Items get evaluated and cleaned up
3. Updated, well-formed stories get added to the backlog, becoming product backlog items (PBIs)
4. Once on the backlog, PBIs get prioritized (and reprioritized)
5. In time, PBIs get pulled into a sprint

In Scrum, the product owner owns the backlog and the prioritization process. They work with the team to determine when to do each item. If real life was this cut and dried, there would be no reason for this book.

[1] "The Scrum Guide." Scrum.org, 8/25/2023, https://scrumguides.org/scrum-guide.html.

WORK INTAKE AND AGILE

You won't find the phrase *work intake* in most (if any) agile texts, but that doesn't mean the concept isn't important. Where does all the work that the teams are doing come from? Where exactly does the work that the product owner prioritizes come from? Work is the one common ingredient that every process requires. Whether you're a startup with one Scrum team, or a thousand-person development shop using the Scaled Agile Framework (SAFe®), work has to come from somewhere—otherwise, you wouldn't have a job.

This book will discuss work intake through the lens of the *Agile Manifesto*. Without disciplined work intake, being agile will always be out of reach.

Agile Manifesto Values

Let's take a look at how the value statements from the *Agile Manifesto* pertain to the topic of work intake:

- **Individuals and interactions over processes and tools:** All companies, organizations, and teams have a process they follow when work appears. This can range from ad hoc to well-controlled. There are times when following a structured process could lead to detrimental outcomes for you and your customers. This is where *individuals and interactions* come into play. Mastering work intake requires communication with individuals up and down the value stream.
- **Working software over comprehensive documentation:** This value statement focuses on the outcomes of work intake conversations and processes. If all you do with new work is have conversations and then generate documentation, you're doing it wrong. Customers buy products that work. They won't buy a document that describes how a product should work.
- **Customer collaboration over contract negotiation:** Some of the work that enters your system will come from internal or external customers. Work requests demand conversation and collaboration to ensure transparency and alignment.

- **Responding to change over following a plan:** This value statement screams *relentless prioritization*.

Agile Manifesto Principles

Academics describe empirical approaches using the phrases "fact-based" and "experience-based." Agilists use the phrase "inspect and adapt." They observe, gather facts, and then adapt. Empirical approaches are at the heart of agile because they generate transparency. Let's take a look at how the principles tie to the work intake cycle.

Before Work Enters (Work Intake Enablers)

We call this group of principles *work intake enablers* because they enable people and organizations to master work intake (see Figure 1.1). They shape behavior before work enters an organization or team.

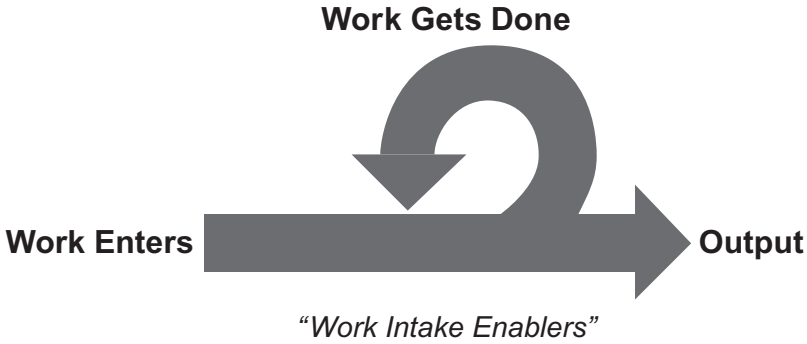


Figure 1.1 *Agile Manifesto* principles enable work intake.

- **Our highest priority is to satisfy the customer through early and continuous delivery of valuable software:** Every company serves customers. This means that you do the work they ask you

to do. It also means that you adjust your priorities because of their priorities.

- **Welcome changing requirements, even late in development; agile processes harness change for the customer's competitive advantage:** *Changing requirements* means a change to the requested work. Those changes count as work intake. This includes new, modified, or canceled backlog items.
- **Simplicity—the art of maximizing the amount of work not done—is essential:** Mastering work intake allows you to do important things. It also enables you to defer things that aren't important.

While Work Is in Progress (Don't Forget These)

We call this group of principles *don't forget these* because they help guide people and teams while work is being done (see Figure 1.2). They also shape behavior while work is being done.

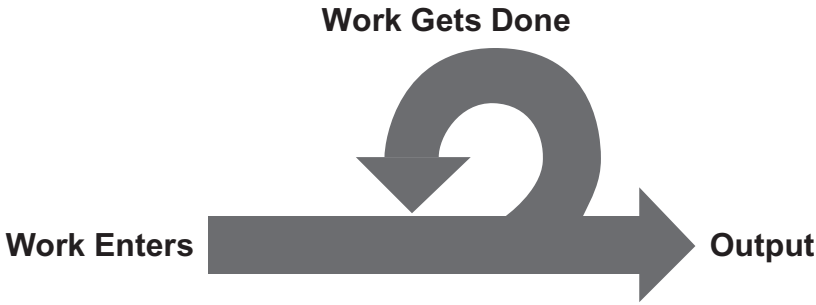


Figure 1.2 *Agile Manifesto* principles to keep in mind while doing work.

- **Business people and developers must work together daily throughout the project:** This principle encourages a better understanding of what to build—and why. Diverge from this principle and your work intake process risks misalignment.
- **Build projects around motivated individuals, give them the environment and support they need, and trust them to get the job done:** A *project* is synonymous with the current work. People do the work that enters your system. Mastering work intake ensures that your people are working on high-value items. Doing high-value work is a motivator for lots of individuals.
- **The most efficient and effective method of conveying information to and within a development team is a face-to-face conversation:** When things change—and they always do—these types of interactions are imperative.
- **Agile processes promote sustainable development; the sponsors, developers, and users should be able to maintain a constant pace indefinitely:** Mastering work intake ensures that people aren't overwhelmed by too much work.
- **Continuous attention to technical excellence and good design enhances agility:** This principle is a reminder to leave space for people and teams to do this type of work. While satisfying customer and business needs is imperative, this work is not optional. Failure to leave space for this type of work can slow down future work. It can also disrupt the flow of value coming from your organization.
- **The best architectures, requirements, and designs emerge from self-organizing teams:** Not all work will enter your system with well-understood requests. At times, some special instruction will need to occur. Asking a select group of senior people to do this work creates bottlenecks in your system. This principle acknowledges a need to push this work down to the people and teams on the front lines.

When Work Is Done (Work Intake Outcomes)

We call this group of principles *work intake outcomes* because they result from mastering work intake (see Figure 1.3).

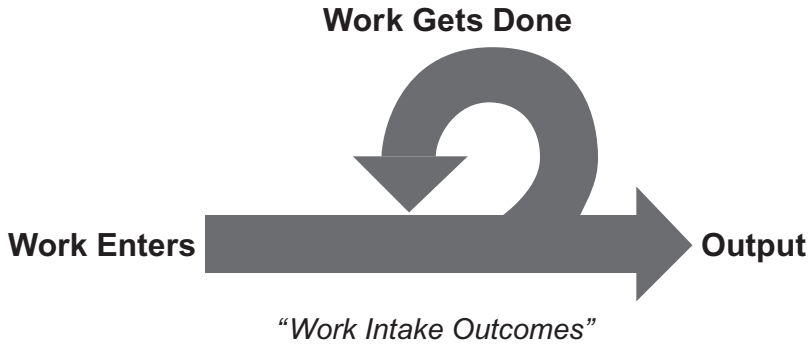


Figure 1.3 *Agile Manifesto* principles are outcomes of mastering work intake.

- **Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale:** Focus less on trying to grasp what customers want; focus more time delivering what they actually need.
- **Working software is the primary measure of progress:** If you're doing small pieces of work and there's flow in your system, a likely result is working software.
- **At regular intervals, the team reflects on becoming more effective, then tunes and adjusts its behavior accordingly:** Retrospect on the work and adjust. In other words, wash, rinse, repeat.

END-OF-CHAPTER QUESTIONS

Use the following to start a conversation about this chapter's contents:

1. Review your work intake practices. Where is there friction with the values and principles from the *Agile Manifesto*? Where are there gaps?
2. How often does unsanctioned work happen in your context? How does it happen?
3. What terms have you used to describe work intake? What terms do other people use?
4. What types of roles in your context control work intake? (For example, product owners.)

2

WHAT DOES GOOD WORK INTAKE LOOK LIKE?

Learning Objectives—by the end of this chapter, you will be able to:

- Understand that there is no single perfect way to control work intake
- Recognize the nine core principles of work intake
- Learn how to assess work intake based on the nine core work intake principles

INTRODUCTION

There is no perfect approach to bringing work into an organization or team. Since it involves people, perfection may not be something that can exist in the real world. Good work intake is independent of the framework you follow. How you achieve *good* is dependent on your context.

Some examples of good approaches include:

- Lean portfolio management in the Scaled Agile Framework (SAFe®)

- Program increment planning in (SAFe®)
- Sprint planning (Scrum and large-scale Scrum)
- Queue replenishment (kanban)

Every organization and team leverages its own version of frameworks and business practices. Implementation is a set of trade-offs that define a continuum. Value delivery and sustainable pace sit on either end of this continuum. We make trade-offs between these as work enters. As we do so, we assess whether our work intake approaches are good.

Different contexts demand never-ending trade-offs between maximizing value delivery and a sustainable pace. For example, teams will always choose to balance value delivery and sustainable pace. This is often called achieving a *work-life balance*. Teams do this when they are not under a death march's pressure and control work intake. The trade-offs change when confronted with a production or business emergency. In these cases, teams will sacrifice a sustainable pace to deliver value.

This brings us to the ultimate value delivery/sustainable pace *antipattern*. (An antipattern is a practice that appears to be useful, but in reality, it's ineffective, counterproductive, or leads to unintended negative consequences.) It's the one where everything is an emergency and teams are on a continuous death march. Unfortunately, this antipattern is not an uncommon approach to management. Teams make choices about what value means in order to protect a modicum of sustainable pace. They make these choices to save themselves from burnout. A phrase like *tech debt* obfuscates what's happening. Corners get cut for the sake of making an unsustainable pace more sustainable.

NINE CORE PRINCIPLES

It doesn't matter whether you are using Scrum, kanban, or waterfall. Good work intake requires nine core principles to be present in some form. Evan Leybourn, founder of the Business Agility Institute, calls this type of list a *Don't Forget Model*. For work intake to be good, a

team or organization has to do something to cover these principles. Forgoing any of these principles will set you on the path to the proverbial *ninth circle of (work intake) hell*.

The nine core principles for good work intake are:

1. **Prioritization:** Prioritization must occur in a systematic way for all work. Approaches without solid prioritization devolve into randomness during times of stress.
2. **Control:** Work needs to follow a defined path to enter an organization or team. Everyone involved in work intake must have a modicum of discipline and self-restraint. They must also have the wherewithal to hold each other accountable.
3. **Transparency:** All interested parties can see the backlog of work. They also understand the process for maintaining the backlog. When transparency fails, people create conspiracy theories and cheat.
4. **Consistency:** Policies and processes must govern how work gets to the backlog. They must be consistent in their application. Carving policies in stone rarely makes sense. The work intake process should evolve based on context, rather than haphazard change.
5. **Frequency:** The processes governing work intake need to happen on a regular basis. The more dynamic the environment, the more often they should occur.
6. **Preparation:** Work intake is a decision or set of decisions. The work has to be ready before those involved can make a decision. Indecision or ad hoc decisions will yield random outcomes.
7. **Respect:** Respect for the process and participants is critical. Respect is not blind belief.
8. **Consequences:** There must be consequences for violating work intake processes. Jumping the queue is not a victimless crime; everyone else gets to pay the price.
9. **Ownership:** Someone is responsible and accountable for work intake. Without it, the flow of work will emulate water and seek the ground in any way possible.

Think of each of these nine principles as a light dimmer. You can adjust them based on the business context. For example, market changes might dictate that priorities get more frequent examination.

Good work intake is independent of how an organization or team manages work. We've seen agile and plan-based organizations practice disciplined work intake. We've also seen agile organizations that were out of control. The out-of-control organizations did not fare well with these nine principles.

Each of these nine principles reflects a range of behaviors. Two human reaction ranges can still cause trouble:

- **Making monolithic decisions on how the organization and all teams approach the principles:** This is akin to mandating that every team will use Scrum and have a two-week sprint starting on a Tuesday. Unless you are in a very small organization, each team will have a very different set of needs. One size may fit one team some of the time, but not every team every time.
- **The propensity to turn any of these principles to zero:** For example, not requiring preparation for planning or refinement sessions will generate frustration, storytelling, and jumping the queue.

To determine how well a team or organization is handling work intake, assess how people act. Before you do anything else, go and observe the people doing the work. Actions are the outward manifestation of the real culture of a team or organization. The rest is aspiration.

An Example—Sprint Planning in Scrum

The example of sprint planning in this section is not an endorsement of Scrum. We often use Scrum—and it is well understood—but many different approaches are useful. Your context will dictate which approach will work in each situation.

Here is an example of the nine core principles as shown via the sprint planning event from Scrum:

1. **Prioritization:** ✔ The product owner prioritizes the backlog.

2. **Control:** ✔ Work enters a Scrum team via the product backlog and sprint planning.
3. **Transparency:** ✔ Both the product and sprint backlogs are visible to the team and stakeholders.
4. **Consistency:** ✓ Events in Scrum are consistent. Scrum is mute on how to regulate work intake.
5. **Frequency:** ✔ The sprint planning event occurs on a cadence.
6. **Preparation:** ✓ Refinement prepares the product backlog for sprint planning. Backlog refinement is the responsibility of the team and is not a specific event. Conducting refinement events is not a universal convention, but is common.
7. **Respect:** ✓ The Scrum value of respect focuses on the people. Scrum is mute on respect for the process.
8. **Consequences:** ✕ Scrum is mute on this subject. The assumption is that work that doesn't go through sprint planning doesn't get done.
9. **Ownership:** ✔ The product owner manages the backlog and accepts work, therefore, the role has ownership.

(✔ = Fully addressed, ✓ = Partially addressed, ✕ = Not addressed)

Scrum fully addresses five of the nine principles. It somewhat addresses three others and misses on one. Communal convention and peer enforcement will plug the gaps. Collectively, they will keep the process moving in the right direction. This can only happen if teams are aware of the core principles and the reasons to address them.

END-OF-CHAPTER QUESTIONS

Use the following topics to start a conversation about the contents of this chapter:

1. Consider how work gets to your team(s). How does the process address the nine core principles?
2. If you found gaps between practice and principles, what is an experiment that will address one of the gaps? What needs to be in place to run this experiment?