From Chaos to Successful Distributed Agile Teams

Collaborate to Deliver

Johanna Rothman

Mark Kilby

From Chaos to Successful Distributed Agile Teams

Collaborate to Deliver

Johanna Rothman and Mark Kilby

This book is for sale at
http://leanpub.com/geographicallydistributedagileteams

This version was published on 2019-06-22

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2019 Johanna Rothman and Mark Kilby

From Mark *- For Chris, Rosalie, Colin and Justin - You taught me much about working remotely and indomitable spirit.*

From Johanna *- For Mark, Shaina, and Naomi. My cave was different this time.*

Contents

Praise Quotes

What Readers Are Saying About

From Chaos to Successful Distributed Agile Teams: Collaborate to Deliver

Anyone who manages or works on some sort of distributed team should read this book. It is a goldmine. Reading this book opened my eyes to a whole level of nuance and complexity I was missing. — Mike Lowery, agile coach

Everyone should read the leadership chapter of this book, take it to heart, and then pass the book to your teams. Work together to become better, more agile teams, anywhere and everywhere. — Michael Herman, Principal Consultant, Michael Herman Associates.

This isn't just a great book for distributed agile teams; it's a great book for any agile team — Ryan Dorrell, Chief Solutions Officer, AgileThought

Thanks to this book, I now understand our distributed team is actually a nebula team and I found a ton of tips that will help us improve our experimentation, communication, and collaboration. A practical book like this was long overdue. — Jurgen Appelo, Author of *Management 3.0* and *Managing for Happiness*

A timely and practical book that is both pragmatic and compassionate—modern product development thinking in a context of healthy distributed teamwork. If you are an agile team member, leader, HR professional, coach, or virtual facilitator, this is your go-to guide for successful distributed teams. — Ellen Gottesdiener, Product Coach

Remote (part-time or full) is a common reality. You now have a guide—packed with years of hands-on experiences and learnings—to help you understand and make your distribution situation work for you rather than against you. You'll discover the tips in this handy companion will serve your collocated and distributed teams for years to come. I wish I had this book 3-4 years ago - it would have saved me and teams I worked with much frustration and misunderstandings. — Marcus Hammarberg, Author of *Kanban In Action* and *Salvation: The Bungsu Story*

From Chaos to Successful Distributed Agile Teams is a tour de force—the best book on teamwork I've read this decade. Two days after starting the book I was implementing small experiments with my own distributed team. And, I'll be recommending this book to all of my clients. —Christopher Avery Ph.D., author of *The Responsibility Process: Unlocking Your Natural Ability to Live and Lead with Power*

Distributed agile is not easy, but it is possible, and worth the journey. Their book emphasizes a people-centered approach to distributed agile—not just to enable a team to do its best work, but also to maintain connection, continuous experimentation, and learning. — Pilar Orti, Director of Virtual not Distant

Conventional agile wisdom assumes collocated teams. Yet, the reality of modern software development is that many teams have at least one, if not all, remote team members. You need to work with the people you have. This book offers pragmatic advice supported by real examples to support and nurture teamwork and high performance in remote teams. — Shane Hastie, Director of Agile Learning Programs, ICAgile

Acknowledgments

We thank these people who read and reviewed the book and provided feedback: Heidi Araya, Tonianne DeMaria, Jesse Fewell, George Dinwiddie, Ryan Dorrell, Dave Gordon, Mike Hansen, Shane Hastie, Michael Herman, David Horowitz, Sue Jasmin, Mike Lowery, Pilar Orti, Craig Smith.

We thank our editors, Rebecca Airmet and Nancy Groth. We thank Karen Billipp for the layout and Jean Jesensky for indexing the print book. Cover design by Brandon Swann.

from Johanna

I thank my clients and geographically distributed agile team workshop participants. You and my Pragmatic Manager readers and Managing Product Development readers helped me refine my ideas with your questions and comments.

I thank Shane Hastie for working with me to develop and lead several distributed team workshops and for writing our articles together.

from Mark

There are many who helped shape my thoughts in this book.

I have been fortunate to work with many distributed agile teams over the last 15 years who trusted me when we experimented together. You provided much of the inspiration for this book.

Through many conversations and collaborations (many online), I'm grateful for those who helped me refine my ideas: Jim Benson, Tonianne DeMaria, Michael Herman, David Horowitz, Pilar Orti, and Lisette Sutherland.

Much of our work in distributed agile teams arises from considering

the art of the possible. I hope all of you join me in remembering Jean Tabaka for that inspiration.

Introduction

Distributed agile teams have a bad reputation: too often, they have problems starting and finishing the work. The managers don't know why. The team members don't know why. People wonder, "Why can't this team just get on with the work?" In the meantime, the team struggles to work as fast and as hard as they can.

For years, if you wanted guidance about how to be a geographically distributed agile or lean team, the answer was, "Don't do that."

"Stop being distributed" or "Don't use agile" is not useful advice. That would require sweeping changes in people's expertise, location, and the organization's ability to deliver products. That would result in disruption of work or possibly loss of valuable team members and product sales.

Distributed work is not the same as collocated work. But the agile principles can be adapted and applied to distributed teams. That's what this book is about.

We wrote this book for three audiences. First, for distributed and dispersed team members, so they can see how they might create communication channels and agile practices that work.

Second, we address those who facilitate and serve distributed teams. We've seen a variety of possible team "leaders." These servant leaders might be coaches, agile project managers, or Scrum Masters. Sometimes, these servant leaders are technical leaders or technical managers. Whatever their title, they facilitate and serve the distributed team.

Our third audience is the managers, executives, and organizational coaches/facilitators who want to take advantage of global talent *and* agile approaches for frequent delivery of value to customers.

These organizational leaders create the environment—the culture for collaboration—in which the teams then work and evolve.

We assume that all of these people—team members, team leaders, and organizational leaders—want everyone on the distributed or dispersed team to work to the best of their capability. However, too often, distributed and dispersed work frustrates everyone. That's because too many teams retain their collocated mindset for their distributed or dispersed teams.

We've seen three necessary mindset changes for successful distributed agile teams. The first mindset change is the agile mindset of encouraging and managing for change. When the team encourages experimentation for *everything*, the team manages how and when it decides to change.

The second mindset change when moving to distributed agile teams is the emphasis on communication and collaboration. When the team creates communication and collaboration norms for *everyone*, the team can eliminate many of their impediments to delivering value.

The third mindset is to use agile principles—not common practices—to create a distributed agile team. When teams use principles to create *their* practices, they adapt their agile approach to fit their context.

With this mindset of experimentation, communication and collaboration, and using principles over practices, distributed agile teams can succeed. Without that mindset, teams work too "slowly" and everyone—from the team members to the executives—becomes frustrated.

That's when people say, "Agile doesn't work for distributed or dispersed teams. It doesn't work for us."

You can create high-functioning, high-performance geographically distributed agile teams. Your teams might change—you may decide that the current team makeup doesn't fit anyone's needs. You might

decide to recreate teams with more hours of overlap. But, you can succeed with geographically distributed agile teams.

Agile and lean approaches will make your problems transparent. Because they do, you may decide that agile geographically distributed teams reveal other problems in your organization. With this transparency, you can make better decisions.

We assume you, our readers, are somewhat familiar with many of the agile terms and practices. We are not going to explain them all in this book. Instead, we offer references to other books you might want to read to gain deeper understanding. We *will* explain how distributed agile teams may adapt specific practices to be successful.

As you read the book, you might notice we use words such as, "We have found…" That phrasing refers to our combined 55+ years of experience with distributed and dispersed teams. We have worked in distributed and dispersed teams in various roles: developer, tester, project manager, program manager, manager, consultant, coach, and workshop leader. Your experience might be different from ours.

Let's start.

1. Distributed Agile Teams Are Here to Stay

Many pundits, via podcasts, articles, and books, have declared remote work the wave of the future.

If you've always commuted to the office and worked with other people in person, this idea of remote work might seem strange. You might even think, "This can't possibly work." How can you learn about your colleagues? How can you collaborate? If you work from home, how can you structure your day?

Agile approaches can answer these questions for geographically distributed teams.

Back when the signatories of the Agile Manifesto released those values and principles, we had insufficient technology to manage remote communication and collaboration. Now, technology allows us to connect and collaborate when we are not face-to-face.

Agile principles amplify the need to connect and collaborate on a frequent basis to deliver value. So while a person may be "remote" from their colleagues, they can now be very connected with those same colleagues through a number of rich and natural communication channels. More importantly, they have even more flexibility than in a traditional office environment to decide when to engage in this intense collaboration with their remote colleagues.

However, using "standard" agile practices does not guarantee success for distributed teams. To make distributed agile teams work, everyone shifts their mindset to a culture of experimentation, communication and collaboration, and principles over practices.

1.1 Understand Agile Teams

We use the word "team" in this book. Based on the work of Katzenbach KAT99 and our experience, an agile team is a cross-functional group of people who:

- Have the necessary skills and capabilities their team requires to deliver on their objectives
- Are committed to a common purpose or goal
- Are interdependent and therefore make commitments about the work to each other
- Learn to understand each other: their strengths, weaknesses, and preferences
- Plan and deliver the work in a collaborative fashion, which can include co-designing, co-creating, pair reviewing, or mob-bing on the work
- Reflect together, reviewing their work and their process in a collaborative fashion
- Are committed to one team and one team only.

Often, collocated team members depend on physical connection. They can work face-to-face. They can mob with the entire team in one room. They can go to lunch as a team to learn more about each other.

A distributed or dispersed team has team members apart from each other. No one has a physical connection to all the other members of the team. However, for successful distributed agile teams to work, each team member must build social relationships with each other member.

Collocated, Distributed, or Dis-persed?

> What is "close enough" for collocation? It's close enough for easy collaboration. The *optimal* distance for communication frequency is less than eight meters. Once team members are separated by 30 meters—and this includes separation by stairs or elevators—their chance of off-the-cuff communication declines dramatically. If your team members are not within 30 meters of each other, you have a distributed team of some type.
>
> Distributed teams have people in several locations, too far away to be collocated with each other. Some people might be close enough to actually walk to each other. "Several" locations can range from two (most people are collocated and one or two are remote, which we we refer to as a satellite team) up to the the number of team members minus one. When at least two people are collocated with each other in multiple places, we refer to it as a cluster team.
>
> Dispersed teams have all people remote from each other. No one is close enough to walk to see each other. The entire team works virtually. We refer to this as a nebula team because too many people confuse dispersed with distributed.

We'll talk more about the kind of team you have and the characteristics of the team types in *Identify Your Distributed Agile Team Type*.

You might not realize that a team with all members on the same campus can be a distributed or dispersed team. However, if floors or buildings separate your people by at least 30 meters, the team is distributed.

However, agile approaches still may not be for your organization or team. So let us explore why you might want to work in a distributed team, why agile principles may be an important aspect of that work,

and how you can ask some critical questions to determine if this is the right type of work for you, your teams, and your organization.

1.2 Why Distributed Teams?

Why do you want to use distributed teams?

We know of several organizations that are completely distributed or dispersed by choice. We suspect there are more organizations who would like to be fully dispersed.

We've seen at least three good reasons to create distributed teams:

- Companies want the ability to hire talented people anywhere in the world or retain people who move.
- Companies need a resilient workforce that can continue despite road closures, bad weather, or other physical challenges in commuting to an office.
- People want the ability to avoid commute time (and possibly energy cost) for their personal benefit.

You may have other reasons for your distributed team.

We've also seen reasons that don't satisfy the organization's needs. Here are several:

- A leader believes low salaries will save project money (see *Trap: Save Money with Lower Salaries*).
- A leader wants to support colleagues in another country (see *Trap: We Can Hire Experts Anywhere*).
- A leader thinks about people as resources instead of as people, and they think they can split the work to make people more efficient (see *Think in Flow Efficiency*).

Once you clarify why you need a distributed team, next consider the decision to use agile approaches for distributed and dispersed agile teams.

1.3 Agile Approaches Focus Distributed Teams

Agile approaches can help distributed teams and team members respond to change. For example, while writing this book, we encountered numerous hurricanes and snowstorms. If we had only been able to work as a collocated pair, we would have had to move to be close to one another *and* we would have lost days of work because of the weather. Instead, we were able to adapt our work and create a resilient project because we worked as a distributed agile team.

We applied the three mindsets necessary for successful agile teams that we mentioned in the Introduction:

> Mindset 1. Manage for change. This means experiment.

We experimented with everything on our project—from writing approaches, tool selection, writing hours, gathering reviewer feedback—and more as we worked through this project. We didn't select one approach for our project. We used the idea of small safe-to-fail experiments with double-loop learning to help us progress through the work.

You might discover that experimentation also applies to your agile approach for your team.

> Mindset 2. Emphasize communication and collaboration.

We selected tools that we could both use that were readily available, inexpensive, and met our security needs. (See *Appendix A* for our toolset, which might not be your toolset.)

For your team, make sure that everyone has access to readily available, inexpensive, and secure technology that allows distributed

teams to collaborate. Make sure the tools aren't a barrier to collaboration.

Mindset 3. Use agile principles, not practices.

We see too many teams assume that focus on a particular practice, such as a standup or backlog refinement, will translate directly to a distributed or dispersed team. Practices don't always translate directly. In this book, we'll define and discuss eight agile principles.

1. Establish acceptable hours of overlap.
2. Create transparency at all levels.
3. Create a culture of continuous improvement with experiments.
4. Practice pervasive communication at all levels.
5. Assume good intention.
6. Create a project rhythm.
7. Create a culture of resilience.
8. Default to collaborative work.

Distributed and dispersed agile teams are here to stay. Distributed team members can experiment, communicate, and collaborate. Distributed agile approaches can help team members and projects become more resilient. And, everyone can benefit if people are not tied to a corporate office.

Questions to Ask First

Before you make the decision to use agile approaches for your distributed team, ask these questions:

Are we willing to let our team experiment with different practices over short durations to see what works best for everyone on the team?

Are we willing to default to collaboration over solo work so

the team can collectively find the best solutions?

Are we willing to challenge what's in the product and how it works, *and* challenge our assumptions of how the team works, all to discover and deliver a successful product?

If the answers to any of these questions are "No," don't despair. Please do continue with the principles in this chapter and the next, and then read *When Agile Approaches Are Not Right for You*.

Now, let's examine the three mindsets in more detail.

1.4 Create a Culture of Experimentation

The first mindset shift is to create a culture of experimentation.

Collocated agile teams tend to experiment with practices. These teams have many resources available in the form of books, courses, and tools to support their experiments. Distributed agile teams—and their leaders—often need to experiment even more. Distributed and dispersed teams now have a variety of technologies to support their experimentation, even though there might not be books or courses to guide experiments.

Consider this story of a team who learned how to create small stories (agile requirements) and deliver those stories every day.

One Team Learned To Experiment as They Worked

The Search team is distributed across three time zones. Stefan, a developer, lives in Romania (UTC+2) . He works with Pierre (the tester) and Madeleine (the product owner) in Paris (UTC +1), and Tom, another developer, in Southampton, UK (UTC).

Stefan and Pierre have extensive experience creating and testing databases with an eye towards performance. They have access to other people in the company—New York (East Coast) and China. However, their team works across those three time zones in Europe.

Stefan, Pierre, and Madeleine learned about agile approaches by first reading books, and then by traveling to one city to take public agile classes together. As an entire team, they decided to try their own agile approach. They created a kanban board to see where the work was and their flow of work. They soon learned that Pierre was overloaded and that Madeleine was too optimistic about what the team could deliver and how fast.

The team created hypotheses: if they reduced their total Work in Progress (WIP), could Pierre keep up with testing? If Stefan and Pierre tested differently, would that make a difference for Pierre and Madeleine? If Madeleine created smaller stories, would the team and the customers have better outcomes?

As the team worked together on their hypotheses, they learned to create shorter feedback loops inside the team and with customers. They created another hypothesis: what if the entire team spoke with customers every so often to learn more about what the customers needed?

Madeleine was concerned at first, because she wanted to make sure the team members weren't insensitive to customers' needs. However, once the team members assured her they would be respectful, the entire team participated as a team with internal customers in the European time zones. They had sufficient hours of overlap.

This team has since worked together for a couple of years choosing other experiments with short feedback loops, and

> they have a fairly dependable throughput. They finish at least one story a day. They have great relationships with their European customers. The team doesn't yet know how to work well with the people across more time zones, but they are working on that.

The Search team embraced experimentation with everything they do.

Experiments can come from the leadership or from the team itself. We'll talk about this more when we discuss the agile principles.

Running too many experiments at once can lead to "change fatigue." Instead, run experiments with a clear hypothesis and a short duration.

Instead of an "all-in" total change to using one specific agile framework, consider trying small experiments to apply agile principles in your teams. We have found that using agile approaches in existing distributed teams has actually improved the work, the distributed working environment, and the morale of team members.

1.5 Shift to a Mindset of Collaboration

The second mindset shift is toward collaboration and communication inside the team. Many distributed and dispersed teams find this shift the most difficult to build and maintain.

In fact, one of the most important elements of successful agile teams is having a sufficient number of hours of overlap. Without

sufficient overlap, communication becomes asynchronous instead of synchronous, and collaboration becomes very difficult.

When a "Team" Isn't a Team

The Payments team belongs to the same organization as the Search team. Payments is dispersed across many time zones: China (UTC +8), Sydney (UTC +10), London (UTC +0), Boston (UTC -5), and San Jose (UTC -8).

The Payments "team" doesn't feel like much of a team. They do not have sufficient hours of overlap to collaborate. However, they have the will to experiment. Here is what they tried over two years:

- They got together for two weeks to learn how to work together.
- They used a kanban board with a planning and reflection cadence to see their workflow.
- They tried to create very small stories and use handoffs, like those described later in *Follow the Sun* to finish work.

They had marginal success as a team. In addition, they all felt pulled by their functional managers to do other work. They had trouble affiliating as a team.

As a result, the Payments team had almost 100% turnover. When the fourth person left the team, he wrote a memo to the managers. He explained that instead of asking people across many time zones to work together, they could either create several Payments teams who had more hours of overlap or create Payments teams in a couple of offices.

That memo helped the managers realize they had to stop thinking about people as cogs and start thinking of people as humans.

> The managers decided to create an East Payments team and a West Payments team. The East team refers to the UTC + time zones and the West team refers to the UTC - time zones. By dividing one "team" with too few hours of overlap into two real teams who can collaborate, the teams were able to finish their feature sets in reasonable time frames.

The experience of the Payments team is what gives distributed and dispersed teams—agile team or not—a bad name. This can change when teams consider and apply the agile principles.

1.5.1 Think in Flow Efficiency

How we think of teams is a part of that second mindset shift to communication and collaboration.

Too often, managers think of the individuals on a team, not the team itself. When managers think of people as "resources," managers think they or the teams can split work to make people more efficient. We split this work by work type, creating experts. The name for this is Resource Efficiency.

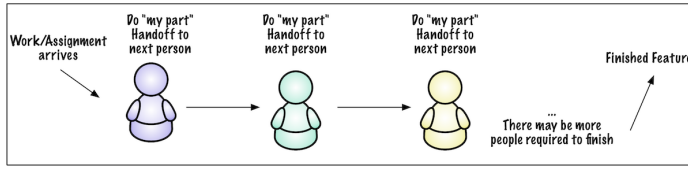

Figure 1.1: Resource Efficiency

The idea of flow efficiency comes from *This is Lean*, MOA13. When a team works as a team—not individuals—they raise their team-based throughput over what any one person can contribute alone. The name for this is Flow Efficiency.

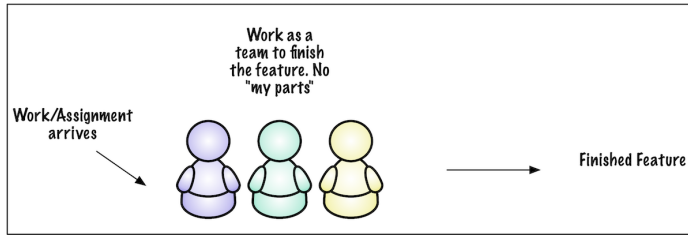

Figure 1.2: Flow Efficiency

Successful agile teams implicitly use flow efficiency. The entire team collaborates to move the work to done. The team doesn't suffer delays from handoffs. And, information doesn't become stale.

Successful *distributed* agile teams reinforce the idea of flow efficiency. The more the team works together, the faster they are. The team learns how to work together, how to learn together, and how to solve problems together. They have few delays in their team process.

That means that the cost models of moving some work to a less expensive location doesn't work if the team has insufficient hours of overlap.

A complete feature team can work anywhere in the world, assuming all the people on that team have sufficient hours of overlap. See *Measure Project Costs* for a more thorough discussion of how to keep project duration short and project costs low. As long as that team is independent, their wage cost may well be lower than other teams' wage costs.

However, if teams or people who have to collaborate together have fewer than four hours of overlap, the delays between the people or the teams may well overcome the wage cost savings [1].

If you serve a team with people who have insufficient hours of overlap, and your managers don't realize that they are creating a

[1] https://www.jrothman.com/mpd/agile/2010/03/wage-cost-and-project-labor-cost/

Cost of Delay (see *Measure Costs of Delay in Distributed Teams*) with the people, map the value stream and measure the team's cycle time.

It's possible to save money on wages with full feature teams. As long as those teams don't have interdependencies on other teams, their throughput will be high enough to compensate for any starting delays.

1.5.2 Non-Collocated Teams Deserve Face-to-Face Time

When we think in flow efficiency, we optimize for the team's deliveries, not a person's. That means non-collocated teams might need to meet on a regular basis.

People often ask us how frequently the non-collocated teams should meet in person.

We've seen teams discover better collaboration and more throughput when they meet for a minimum of a week once a quarter.

If you're not sure how often your team should meet in person, consider asking the team to *Map the Value Stream to Visualize Cycle Time* and to measure their satisfaction. When teams experience an increase in cycle time and a decrease in personal satisfaction, they tend to have systemic problems. People often need to meet to discover and fix systemic problems.

If you are starting or restarting a non-collocated team, the cheapest thing you can do is to bring the team together for a week or two.

In Week 1, create a temporary team room so people can work together, physically, in one place. The team can build mutual understanding and trust as they work together. Take this time to understand each other's strengths, work preferences, and typical work hours. (See *Build Respect With Working Agreements*). When the team works together, they can explore possibilities for: more

hours of overlap; how to plan; how to hand off work; how to offer feedback to each other; and how to reject work that does not meet the team's standards.

If the team has a Week 2, consider conducting the second week in the team's virtual workspace, even though the team remains physically together. When teams practice in their virtual workspace, they select their tools for planning, development, review, and other possible tasks. For risk management, the team might also select and practice with their secondary collaboration tools and backchannels. In this second week, the team agrees on, practices with, and manages the risks of their virtual workspace. Because the team is collocated, they can also choose to conduct a quick retrospective each day to refine how they work together. We recommend the team transition to using their virtual workspace for their retrospectives during this week. This second week creates and cements the team's working agreements.

We've seen teams learn to experiment in a one day face-to-face, when they could not understand how to experiment at all when they were dispersed. After that one day experiment, they brainstormed three ways to experiment back in their offices. They continue to experiment.

The payoff for face-to-face time might be when:

- Teams charter their project in an hour or less, and then are able to discuss what's in and out of the release with the product owner.
- Teams learn how to pair, swarm, and mob when they are in the same room and they are then able to decide how to pair, swarm, and mob when they are back in their offices.
- Teams discuss their interpersonal challenges of how to offer and receive feedback and coaching from each other. Being face-to-face allows them to learn how to talk with others and how to make sure the other person heard them.

You might think the travel expenses are "too high." However, the team saves time and money through faster clarification of the vision, mission, goals, and working agreements. That clarification speed is worth the investment.

If you have a large team, those costs can appear prohibitive.

However, capitalizing software can offset the operational expenses. If you bring a team together for just one week, and they release a feature that you can capitalize, you might "make" all your money back in just one week. And, the team now has the skills to continue to release finished product more often.

If your team cannot collocate for at least a week, consider visualizing the team's value stream and measuring the team's *Measure Costs of Delay in Distributed Teams* to visualize and understand your team's costs when you can't bring people together.

Now that we've addressed the experimentation and collaboration mindset, let's set the stage for principles over practices.

1.6 Review the Agile and Lean Principles

We assume you are familiar with the four values of the Agile Manifesto. However, you might not be familiar with the twelve principles of agile software development [2]. This list is a paraphrase of the principles. We expect any agile team (distributed or not) to use and live by these principles.

1. Deliver early and often to satisfy the customer.
2. Welcome changing requirements.
3. Deliver working software frequently.
4. Business people and developers must work together.

[2] http://www.agilemanifesto.org/principles.html

5. Trust motivated people to do their jobs.
6. Face-to-face conversation is the most efficient and effective method of conveying information.
7. Working software is the primary measure of progress.
8. Maintain a sustainable pace.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. Reflect and adjust at regular intervals.

Changes in technology in the last decade have redefined what "face-to-face" can mean for distributed teams.

In addition, we find the two pillars of lean [3] especially helpful for distributed teams:

1. Respect for people
2. Continuous improvement

And the principles of lean software development as in *Lean Software Development: An Agile Toolkit*, POP03 provide a useful perspective when we consider how to support distributed teams:

1. Eliminate waste
2. Amplify learning
3. Decide as late as possible
4. Deliver as fast as possible
5. Empower the team
6. Build integrity in
7. See the whole

[3] http://www.leanprimer.com/downloads/lean_primer.pdf

Agile teams tend to create respect for people inside the team as a matter of course. In addition, we have entire chapters about safety and respect: *Create Your Collaborative Team Workspace* and *Build Respect With Working Agreements*.

1.7 Is a Distributed Agile Approach Right for You?

Not every organization can use a distributed agile approach. A distributed agile approach might not work for your organization if:

- Your organization wants to use a hierarchy instead of a networked approach to the work. Distributed team members need to be able to solve their problems as a team without relying on the management hierarchy.
- Your organization only tracks expenses instead of value.
- Your organization measures people on their resource efficiency, instead of a team-based flow efficiency.
- Your organization prefers to establish core hours for all employees instead of letting teams decide their optimal core hours for working collaboratively.
- Your organization prefers to center all planning and activity around "headquarters" instead of teams that may be located around the globe.
- Your managers believe that people only need to know certain things, not the entire context of a project or some work. (e.g., stakeholders, key business goals, return on investment)
- A management-designated leader—some single person—needs to be involved with every team decision, instead of letting the team decide for themselves how to work. Sometimes, that leader is actually a small cluster of people, often close to headquarters.

- The team members prefer to be told what to do, when to do it, and how to do it, instead of taking work and determining who they need on the project and how to finish it.
- Team members prefer to keep to themselves instead of sharing some personal context and goals so teammates understand where their teammates' strengths are and what motivates their best work.

Agile approaches are much more than a project change. Agile approaches change the organization's culture.

We have found that people, teams, and managers who believe in and use the agile and lean principles are able to move past behaviors and beliefs like those listed above so that they can use an agile approach. However, sometimes, an agile approach isn't right for a team.

1.8 When Agile Approaches Are Not Right for You

Adopting agile practices without thinking about the principles does not make sense for many distributed teams. (See *Adapt Practices for Distributed Agile Teams*.)

Consider whether agile approaches make sense for your teams. For instance, some teams have too few hours of overlap to take advantage of agile approaches. In that case, consider asking them to use a different project life cycle. Teams have many choices other than only waterfall or agile approaches.

Teams can use iterative and incremental approaches to the work, without requiring the collaboration and learning from an agile approach. See *Manage It! Your Guide to Modern, Pragmatic Project Management*, ROT07 and *Agile and Lean Program Management*,

ROT16A for details about choices. In addition, see *What Lifecycle? Selecting the Right Model for Your Project* .[4]

If you decide an agile approach is not right for the team, do consider organizing as a collaborative cross-functional team. Also, ask the team how they might limit their work in progress (WIP), and how they will know they are done. Teams who work via the agile principles, regardless of the project life cycle, often have better outcomes than teams that don't. Even if they don't use a recognizable agile approach.

Even if the teams might not look like they're using an agile approach, you, as a leader in the organization, can use an agile mindset to create the culture in which distributed teams can flourish.

1.9 See Traps That Prevent Successful Distributed Agile Teams

We've seen the future of successful distributed agile teams. We've also seen too many traps—often at the management or executive levels—that prevent successful distributed agile teams.

We've seen these traps prevent success in distributed agile teams:

- Assuming that lower salaries will create a lower project cost.
- People have an insufficient experimentation mindset.
- The teams were not originally created to use a culture of collaboration.
- Imposition of a specific agile approach and practices, rather than using principles for this team's context.

[4] https://www.jrothman.com/articles_/2008/01/what-lifecycle-selecting-the-right-model-for-your-project/

- Managers hold traditional expectations when distributed teams move to agile.

1.9.1 Trap: Save Money with Lower Salaries

For too long, managers have believed that software teams (or other knowledge work) is similar to factory work. Factory work often uses a divide-and-conquer approach to dividing up work to see it finish.

However, knowledge work includes innovation and learning as key parts of what a team does. Team members collaborate to discover the product as they build it.

When you separate team members by a few hours of overlap, the team has trouble working together as in Think in Flow Efficiency.

See *Avoid Chaos with Insufficient Hours of Overlap* for the details.

1.9.2 Trap: Insufficient Experimentation Mindset

Agile approaches are based on collaboration and experimentation because the team receives frequent feedback. Distributed teams need to experiment even more than collocated teams, because they don't have the easy opportunity for off-the-cuff feedback.

Managers, especially, need to build their experimentation mindset, so they can consider alternatives for the product and their processes. When leaders experiment, they empower the teams to do so.

Consider these options:

- Ask the question, "What are our hypotheses, what options do we have to test them, and how will we measure the results?"

for everything: especially the team's practices and the product features.
- If the team is new to experimentation, set a timebox and ask them to choose the option where they can most easily and regularly gather data. Start with the smallest possible experiment.
- *Map the Value Stream to Visualize Cycle Time* for the team on a regular basis to understand the team's work times and wait times. Use that data to visualize potential experiments.

Small frequent experiments help distributed teams become more effective, regardless of their agile approach.

See *Focus on Principles For Your Distributed Agile Teams* to understand how to think about successful distributed agile teams. In addition, read these chapters:

- *Identify Your Distributed Agile Team Type*
- *Communicate to Collaborate*
- *Create Your Collaborative Team Workspace*
- *Cultivate Your Distributed Team's Agile Culture*
- *Build Respect With Working Agreements*
- *Adapt Practices for Distributed Agile Teams*
- *Integrate New People Into Your Distributed Agile Team*

1.9.3 Trap: Teams Were Not Originally Created for Collaboration

We described how to *Think in Flow Efficiency*, which might be a new idea for you.

The more everyone understands flow efficiency, the more they will foster collaboration in the teams. If your team was originally created for handoffs, rethink the flow of work through your team. See *Map the Value Stream to Visualize Cycle Time*.

In addition, read these chapters:

- *Create Your Collaborative Team Workspace*
- *Cultivate Your Distributed Team's Agile Culture*
- *Build Respect With Working Agreements*
- *Integrate New People Into Your Distributed Agile Team*

1.9.4 Trap: Imposition of a Specific Agile Approach

Each team needs its own rhythm, which we'll discuss in more detail in *Focus on Principles For Your Distributed Agile Teams*. Instead of focusing on specific practices, ask the teams to use the *Agile and Lean Principles* and learn to release value as often as possible. Teams might deliver at varying cadences—and the key is to keep those cadences as short as possible.

In addition, please see these chapters:

- *Create Your Collaborative Team Workspace*
- *Cultivate Your Distributed Team's Agile Culture*
- *Build Respect With Working Agreements*
- *Adapt Practices for Distributed Agile Teams*

Too many people believe specific agile approaches can work for any team. We have seen successful agile teams use the principles to adapt practices.

1.9.5 Trap: Managers Hold Traditional Mindsets

Managers hold the culture that allow distributed agile teams to succeed.

We described the three mindset shifts to: experimentation, collaboration, and principle-based agile approaches. To create a successful distributed agile culture, managers need to change their mindset, also. This might challenge the best managers, because *their* managers haven't changed the management measures.

Please do read:

- *Focus on Principles For Your Distributed Agile Teams*
- *Cultivate Your Distributed Team's Agile Culture*

Agile approaches change the culture, not just of the agile team, but for the organization. Successful agile managers work up and down the hierarchy, helping their colleagues and *their* managers change the measurements and desired outcomes. See *Lead Your Distributed Agile Teams to Success*.

1.10 Now Try This

1. Write down why you or your organization wants to use (or is using) distributed teams. If one of the reasons involves cost, review the traps in this chapter. Now, define your business reasons for a distributed agile approach.
2. Consider how you, your team(s), and organization might use experiments. Does everyone on the team have an equal voice in proposing possible experiments about product and process?
3. Consider how you, your team(s), and organization currently collaborate. Is it possible for teams to use flow efficiency, rather than resource efficiency, to collaborate on the entire project, from defining the requirements through to delivery?

Next, let's see the principles that successful distributed agile teams use to create great working team environments.

About This Sample

This sample contains the first chapter of *From Chaos to Successful Distributed Agile Teams.*

The Table of Contents is:

If you enjoyed this chapter, we hope you decide to buy the entire book. If you have questions, please email us.

If you are not sure if this book is right for you now, please consider joining our email lists:

- Johanna's email list, Pragmatic Manager
- Mark's email list, Distributed Agile Field Notes

Thanks!

Annotated Bibliography

[AMA11] Amabile, Teresa and Steven Kramer. *The Progress Principle: Using Small Wins to Ignite Joy, Engagement, and Creativity at Work*. Harvard Business Review Press, Boston, 2011. They have completed the research that says we like to finish work in small chunks so we can make progress.

[BRO95] Brooks, Frederick P. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition (2nd Edition)* Addison-Wesley, Boston, 1995. How software projects and teams really work. Learn from a master.

[COH05] Cohn, Mike. *Agile Estimating and Planning*, Prentice Hall, New Jersey, 2005. The seminal work on estimating and planning for agile projects.

[COC06] Cockburn Alistair. *Agile Software Development: The Cooperative Game, 2nd ed.* Addison-Wesley, Boston, 2006. One of the first books describing agile principles and practices and defining the importance of face-to-face communication in the cooperative game of software.

[COV89] Covey, Stephen R. *The 7 Habits of Highly Effective People: Powerful Lessons in Personal Change*. Simon and Schuster, 1989. We have found that these habits help us structure our lives and our leadership for the better.

[CSI08] Csikszentmihalyi, Mihaly. *Flow: The Psychology of Optimal Experience*. HarperCollins Publishers, 2008. Learn how people finish work, separately and together.

[DAL86] Daft, Richard L. and Robert H. Lengel. *Organizational Information Requirements, Media Richness and Structural Design* in *Management Science*, Vol. 32, No. 5, Organization Design. (May, 1986), pp. 554-571. The oft-cited article about media

richness and how the media can help or hinder resolution of uncertainty and equivocality.

[DER06] Derby, Esther and Diana Larsen. *Agile Retrospectives: Making Good Teams Great.* Pragmatic Bookshelf, Dallas, TX and Raleigh, NC, 2006. The classic work about agile retrospectives.

[EOH13] Eoyang, Glenda H. and Royce J. Holladay. *Adaptive Action: Leveraging Uncertainty in Your Organization.* Stanford Business Books, 2013. Teams are complex adaptive systems. The first book to describe how containers, differences, and exchanges influence the entire adaptive system.

[EDM12] Edmondson, Amy C. *Teaming: How Organizations Learn, Innovate, and Compete in the Knowledge Economy.* Jossey-Bass, San Francisco, 2012. How self-organized teams really work, and what we need to make them work in different cultures.

[ERR11] Carmel, Erran & J. Alberto Espinosa. *I'm Working While They're Sleeping: Time Zone Separation Challenges and Solutions.* Nedder Stream Press, 2011. A terrific general-purpose read about geographically distributed people and approaches you might use to manage the time zone differences.

[ERR99] Carmel, Erran. *Global Software Teams: Collaborating across Borders and Time Zones.* Prentice Hall. 1999. The first book that discusses the ideas of how global teams might work.

[KAN14] Kaner, Sam *Facilitator's Guide to Participatory Decision-Making, 3rd Edition.* Jossey-Bass. 2014. If you need to facilitate any meetings at all, use the ideas in this book. We both find it an invaluable resource.

[KAT99] Katzenbach, Jon R. and Douglas K. Smith. *The Wisdom of Teams: Creating the High-Performance Organization.* Harper-Collins Publishers, New York, 1999. Another classic, about differentiating workgroups from teams and what creates a

team.

[KOC05] Kock, Ned. *Media Richness or Media Naturalness?* in IEEE Transactions on Professional Communication, Vol 48, No. 2, June 2005, pp. 117-130. Excellent rebuttal to communication richness.

[MOA13] Modig, Niklas and Pär Åhlström. *This is Lean: Resolving the Efficiency Paradox*. Rheologica Publishing, 2013. Possibly the best book about how managers should consider agile and lean. A wonderful discussion of resource efficiency vs. flow efficiency.

[POP03] Poppendieck, Mary and Tom Poppendieck. *Lean Software Development: An Agile Toolkit*. Addison-Wesley, Boston, 2003. The first book to provide a lean approach to software. Contains the simplified value stream mapping we show here.

[BCD05] Rothman, Johanna and Esther Derby. *Behind Closed Doors: Secrets of Great Management*. Pragmatic Bookshelf, Dallas, TX and Raleigh, NC, 2005. We describe the Rule of Three and many other management approaches and techniques in here.

[ROT07] Rothman, Johanna. *Manage It! Your Guide to Modern, Pragmatic Project Management*. Pragmatic Bookshelf, Dallas, TX and Raleigh, NC, 2007. Waterfall and agile are just two of the four major kinds of life cycles. You can select which life cycle will make the most sense for you. If you want to examine other life cycles and learn how to replan, read this book.

[RE14] Rothman, Johanna and Jutta Eckstein. *Diving for Hidden Treasures: Uncovering the Cost of Delay in Your Project Portfolio*. Practical Ink, 2014. A book about Cost of Delay and how to see how those costs affect your project portfolio.

[RHI13] Rothman, Johanna and Shane Hastie. *Lessons Learned from Leading Workshops about Geographically Distributed Agile Teams* in *IEEE Software*, March/April 2013, pp 7-10.

Shane and Johanna developed and facilitated several workshops about geographically distributed teams in a variety of locations between 2011-2014. This article captures our lessons learned from the workshops. The article is available at https://www.jrothman.com/articles/2013/03/lessons-learned-from-leading-workshops-about-geographically-distributed-agile-teams/.

[ROT12] Rothman, Johanna. *Hiring Geeks That Fit*. Practical Ink, 2012. If you want to know how to hire people, this is it, from soup to nuts. All the templates are available for free on Johanna's website. The book explains how to use them.

[ROT16A] Rothman, Johanna. *Agile and Lean Program Management: Scaling Collaboration Across the Organization*. Practical Ink, 2016. A program is a collection of projects with one business objective, often requiring several feature teams. Learn how to scale the collaboration, not the process.

[ROT17] Rothman, Johanna. *Create Your Successful Agile Project: Collaborate, Measure, Estimate, Deliver*. Pragmatic Bookshelf, Raleigh, NC, 2009. Learn the difference between iteration- and flow-based agile approaches and what practices to consider when. Includes how-tos for workgroups and managers.

[ROR98] Smith, Preston and Donald J. Reinertsen. *Developing Products in Half the Time: New Rules, New Tools*. Wiley, 1998. A classic introduction to many ideas of lean and agile for product development.

[SCH10] Schein, Edgar H. *Organizational Culture and Leadership*. Jossey-Bass. San Francisco 2010. Culture is not about the color of the walls or the foosball tables. Culture is about us, as humans. A fascinating look at what culture means.

[SUN18] Sutherland, Lisette and Kirsten Janene-Nelson. *Work Together Anywhere: A Handbook on Working Remotely—Successfully—for Individuals, Teams, and Managers*. Collaboration Superpowers, Delft, the Netherlands. 2018. Full of tips and ideas for managers and team members about how to facilitate remote

work. Includes guidance on preparing for a transition to remote work for any kind of team.

[TAB06] Tabaka, Jean. *Collaboration Explained: Facilitation Skills for Software Project Leaders.* Addison-Wesley, 2006. One of the first books to introduce skills for planning and facilitating various agile meetings such as workshops, sprint planning, release planning, retrospectives, and more.

[WEI93] Weinberg, *Gerald M. Software Quality Management, Vol 2: First-Order Measurement.* Dorset House Publishing, New York, 1993. This particular Weinberg book has excellent definitions of congruence and human interaction.

Glossary

Agile Approach: A collaborative team-based approach to finishing valuable work. The value of working in an agile way is that you have the ability to replan quickly, because the team completes work.

Backchannel: An application that allows everyone on the team to communicate and coordinate when another communication channel may be in use (audio/video). We particularly like a chat application for a team's backchannel.

Backlog: Ranked list of items that need to be completed for the product.

Bus factor: The risk of not sharing information among all team members. When a team has a bus factor of "one," that means only one team member has the necessary information.

Buddy System: People who help a remote team member maintain connection for the work.

Collocated: When team members sit side by side, or, at least when everyone is within 30 meters of each other.

Copilot: A person who assumes a similar role to assist across multiple groups or time zones. For instance, a Product Owner or a meeting facilitator in one location may have a copilot in another location who can help coordinate activities and responsibilities.

Community of Practice: A way to share knowledge among people who belong to different teams, and share the same interests or function. For example, in a program, you might have an architecture community of practice that helps any developer learn how to evolve the design of the product. A test community of practice would provide a forum for testers to discuss what and how to test.

Conway's Law: The architecture of your product reflects the location/architecture of the team who created that product. Originally described in *The Mythical Man Month.*

Cost of Delay: The revenue impact you incur when you delay a project. Aside from "missing" a desired release date, you can incur Cost of Delay with multitasking, or waiting for experts, or from one team waiting for another in the program. All of these problems—and more—lead to delay of your product release.

Digital nomad: Knowledge worker who works from anywhere as long as they have electricity and a high-speed network connection. Nomads include anyone who might travel to a new location (or country) every 2-3 months for a change of scenery and to optimize their preferred work environment.

Distributed: When some team members are separate from each other, the team is distributed.

Dispersed: When none of the team members share any collocated space, the team is dispersed.

Double Loop Learning: Assess and challenge everything about the product and the project. Because agile approaches allow the team to finish frequently, the team (and the Product Owner or customer) can reassess their product progress and assumptions, as well as the project and process assumptions.

Flow: Instead of planning for a timebox of a week or two, the team limits the number of items under consideration. The team still takes the work in rank order.

Generalizing Specialist: Someone who has one skill in depth, and is flexible enough to be able to work across the team to help move a feature to done.

Geo-fence: The teams or team members are separated by more than 30 meters, even if they are all in one zip code.

HiPPO: Highest Paid Person's Opinion. A HiPPO can derail any decision about anything.

Information Radiator: Any physical or virtual board that provides up-to-date information on the status of the work or the product.

Kanban: Literally the Japanese word for "signboard." A scheduling system for limiting the amount of work in progress at any one time.

Kaizen: A philosophy for continuous improvement.

Lean: A pull approach to managing work that looks for waste in the system, a holistic approach. The two pillars of lean are respect for the people and continuous improvement.

Lean Coffee: A meeting structure where the people attending the meeting evolve the agenda as they proceed, always finishing a conversation on the most valuable item first. See http://leancoffee.org/.

MVP: Minimum viable product. What is the minimum you can do, to create an acceptable product? This is not barely good enough quality. This is shippable product. However, this is minimal in terms of features.

Pairing: When two people work together on one task, one monitor, and one keyboard. They work only on this task together.

Parking Lot: This is a place to put issues you don't want to lose but don't necessarily want to address at this time.

Psychological Safety: Each team member has the capability and motivation to say what they perceive and feel, in an environment of respect for each other.

Spike: If you cannot estimate a story, timebox some amount of work (preferably with the entire team) to learn about it. Then you will be able to know what to do after the day or two timebox.

Servant Leadership: An approach to managing and leading where the leader creates an environment in which people can do their best work. The leader doesn't control the work; the team does. The leader trusts the team to provide the desired results.

Sprint: An iteration in Scrum.

Story or **User Story**: A requirement in the form of value to a person.

Swarming: When the team works together to move a feature to done, all together.

Technical Debt: Shortcuts a team takes to meet a deliverable. Teams might incur technical debt on purpose, as a tactical decision. Technical teams can have architectural, design, coding, and/or testing debt. Program teams might have risk or decision debt—the insufficiency of work for managing risks or making decisions.

Timebox: A specific amount of time in which the person will attempt to accomplish a specific task.

WIP or Work in Progress: Any work that is not complete. When you think in lean terms, it is waste in the system. Note that you do not get credit for partially completed work in agile approaches.

More from Johanna

I consult, speak, and train about all aspects of managing product development. I provide frank advice for your tough problems—often with a little humor.

If you liked this book, you might also like the other books I've written:

- *Create Your Successful Agile Project: Collaborate, Measure, Estimate, Deliver*
- *Manage Your Project Portfolio: Increase Your Capacity and Finish More Projects, 2nd ed*
- *Agile and Lean Program Management: Scaling Collaboration Across the Organization*
- *Diving for Hidden Treasures: Uncovering the Cost of Delay Your Project Portfolio*
- *Predicting the Unpredictable: Pragmatic Approaches to Estimating Project Cost or Schedule*
- *Project Portfolio Tips: Twelve Ideas for Focusing on the Work You Need to Start & Finish*
- *Manage Your Job Search*
- *Hiring Geeks That Fit*
- *Manage It!: Your Guide to Modern, Pragmatic Project Management*
- *Behind Closed Doors: Secrets of Great Management*

In addition, I have essays in:

- *Readings for Problem-Solving Leadership*
- *Center Enter Turn Sustain: Essays on Change Artistry*

I'd like to stay in touch with you. If you don't already subscribe, please sign up for my email newsletter, the Pragmatic Manager, on my website. Please do invite me to connect with you on LinkedIn, or follow me on Twitter, @johannarothman.

I would love to know what you think of this book. If you write a review of it somewhere, please let me know. Thanks!

Johanna

More from Mark

With over two decades of experience in agile principles and practices, Mark Kilby has cultivated more distributed and dispersed teams than collocated teams. He has consulted with organizations across many industries and coached teams, leaders and organizations internally.

Mark also co-founded a number of professional learning organizations such as Agile Orlando, Agile Florida, Virtual Team Talk, and the Agile Alliance Community Group Support Initiative among others.

His easy-going style helps teams learn to collaborate and discover their path to success and sustainability. While sharing his insights from this work in many publications, Mark shares most of his ideas and developments on https://www.markkilby.com.