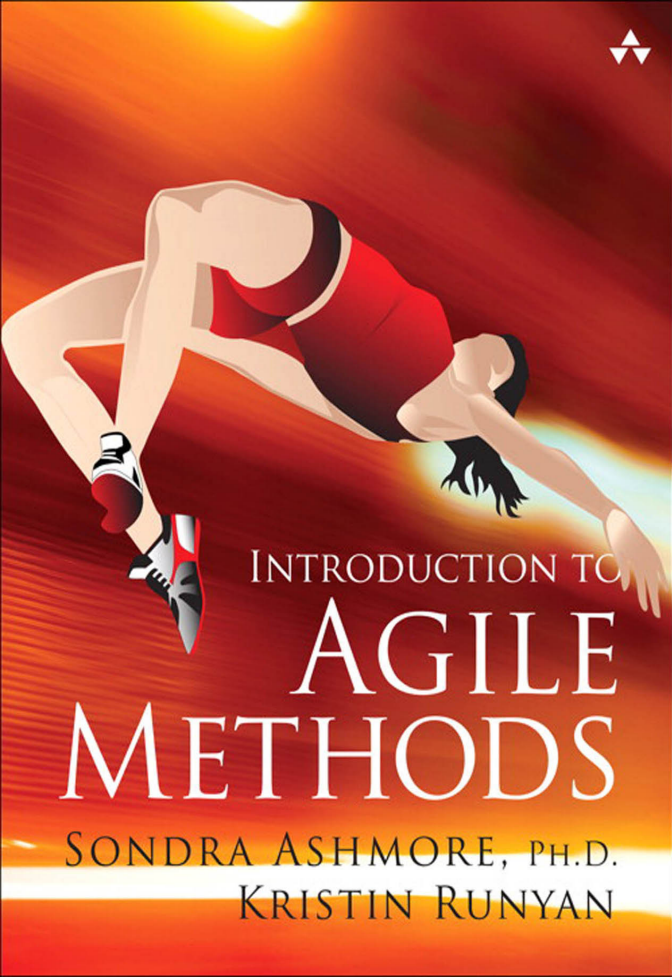

Introduction to Agile Methods

This page intentionally left blank

Introduction to Agile Methods

Sondra Ashmore, Ph.D.
Kristin Runyan

Addison-Wesley

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

Library of Congress Cataloging-in-Publication Data
Ashmore, Sondra.
Introduction to agile methods / Sondra Ashmore, Ph.D., Kristin Runyan.
pages cm
Includes index.
ISBN 978-0-321-92956-3 (pbk. : alk. paper)
1. Agile software development. 2. Open source software. I. Runyan, Kristin. II. Title.
QA76.76.D47A83 2014
005.3—dc23

2014014568

ISBN-13: 978-0-321-92956-3
ISBN-10: 0-321-92956-X

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.
First printing, July 2014

For my sons, Drake and Dane, who encourage me to follow my dreams, and for my husband, Brian, who helps make them all possible.

—Sondra Ashmore

For my family and friends, from whom I am continuously learning.

—Kristin Runyan

This page intentionally left blank

Contents

Preface

What is Agile software development, and what does it mean when someone says, "Our team used Agile to develop software"? In our experiences, we have encountered many people who can tell you about some of the Agile tools they have heard about or experienced, such as Scrum meetings or paired programming, but few touch on the fact that it is a fundamentally different approach to creating software.

Over the last several years, we were both teaching entry-level Agile software development courses—Sondra was teaching graduate students at Iowa State University, and Kristin was training her staff at her company—and we struggled to find a basic book on Agile software development that we could use in a classroom setting. When we met while doing volunteer work for a local technology nonprofit organization, we discovered that we were facing very similar challenges in finding the right book. It was that conversation that sparked the idea of writing our own book on Agile methods.

Our introductions to Agile were similar: We both were managing software development teams that were accustomed to using the more traditional method of software development called "Waterfall," and we took on the challenge of implementing some of the Agile software development tools into our projects, with the ultimate goal of becoming Agile software development organizations. Although we both jumped in and learned all we could about Agile software development, we quickly realized that Agile is a lot more than training people on new tools and methodologies. It did not take long to understand that our organizations had to make a cultural shift to become Agile, which introduced many more challenges than we had anticipated. The transition to Agile was not a destination, as our executive teams had predicted, but rather a journey that is constantly evolving based on the lessons we have learned, or *retrospectives* in Agile terminology.

We firmly believe that Agile methods offer very real benefits for the world of software engineering. Our goal is not to provide you with a comprehensive collection of everything there is to know about Agile software development, but rather to provide you with the basics you need to know to get started. Alistair Cockburn, who is featured in Chapter 3, applies the Aikido term "Shu-HaRi," meaning "to learn a skill or technique," to the process of learning Agile

methods. First you are in the "Shu" phase, when you must mimic your teacher precisely to master the basics. Next is "Ha," where you start to learn from other teachers that help you build on your skills, and you begin to gain knowledge about the history and theoretical basis for the technique. Finally, you reach the "Ri" phase, where you become the teacher and make new contributions to the technique. It is our hope that after reading this book, you are well on your way to the "Ha" stage.

We start by introducing you to the history of Agile software development, then build on the tools and techniques that are common in most Agile organizations. We end by discussing how the process comes together to launch new and exciting products to the marketplace. We have included interviews with practitioners to give you a sense of what happens in organizations that have adopted Agile methods. Each chapter concludes with a summary, suggestions for additional reading, and review questions.

The following is a brief introduction to the information you will find in each chapter:

- **Chapter 1—The History and Value of Agile Software Development**
  This chapter provides the background of the Agile movement and compares Agile to the more traditional Waterfall methodology. We explore the use cases for both Waterfall and Agile and explain the pros and cons of each. The chapter introduces the Agile Manifesto, its values, and its authors. We review the 12 key principles of Agile and introduce a fictitious company, Cayman Design, that is used for example purposes throughout the book. We include an interview with Robert (Uncle Bob) Martin.
- **Chapter 2—Organizational Culture Considerations with Agile**
  Moving from Waterfall to Agile requires a cultural transformation, and this chapter delves into the impacts, benefits, and pitfalls. We explore the Agile transformation from the viewpoint of a team member, manager, and executive to help readers understand how roles and decision-making processes will change. Our interview with Scott Ambler brings the concepts together.
- **Chapter 3—Understanding the Different Types of Agile**
  This chapter describes the different Agile methodologies: Scrum, Kanban, Extreme Programming (XP), Crystal, feature-driven development (FDD), Lean software development, and the Dynamic Systems Development Method (DSDM). We provide descriptions and examples of when each methodology will work best. We outline the available certifications for

each methodology. An insightful interview with Alistair Cockburn is also included.

- **Chapter 4—Describing the Different Roles**
  This chapter provides roles and responsibilities for the various titles prominent in the different Agile methodologies. We start with a deep dive into Scrum and explore the nuances of roles such as product owner, Scrum master, and Scrum team. We then compare and contrast those roles with each of the methodologies described in Chapter 3 and specifically reference the commonalities. In addition to the standard descriptions, we explore how the different roles are deployed within organizations. This chapter includes compelling interviews from Roman Pichler and Lyssa Adkins.
- **Chapter 5—The New Way to Collect and Document Requirements**
  This chapter focuses on the front end of the Agile process, where customer and market feedback is incorporated into meaningful requirements. It addresses terms and ideas such as user stories, epics, acceptance criteria, understanding and measuring business value, prioritization, roadmaps, and burn-up charts. We also show how requirements can be enhanced with elements such as personas and usability. We dive into communication strategies and, finally, explore how Lean software development and the Lean start-up movement affect requirements. Our interview with Ellen Gottesdiener and Mary Gorman includes great insights.
- **Chapter 6—Grooming and Planning**
  As the development process progresses, requirements/user stories are "groomed" into usable input for the development teams. The requirements are prioritized, and we present several strategies for prioritization. The size of the user stories is estimated using a variety of techniques, including story points. Once the estimating is complete, the Sprint planning or XP planning game begins, incorporating the velocity of the team, current business conditions, and the amount of things such as technical debt and bugs. We explore the theory of triple constraints (scope vs. time vs. resources) and how the management of these can affect the development progress. Our interview with Mike Cohn showcases his breadth of experience in the practical application of Scrum.
- **Chapter 7—Testing, Quality, and Integration**
  This chapter explains how quality can be maintained and even enhanced using Agile tools. One of the key tenets of Agile is "frequent verification and validation" of working software, so we devote this chapter to different testing approaches, such as test-driven development, acceptance test-driven development, integrated testing, regression testing, and unit testing.

We provide a complete example of test-driven development, including code references. Our interview with Tim Ottinger rounds out all of the testing information.

- **Chapter 8—Tracking and Reporting**
  This chapter emphasizes the importance of tracking and reporting progress in the Agile process. To understand the tracking process, we define the necessary meetings such as the daily stand-up, Sprint review/demo, and Sprint retrospectives. We also dive into Kanban, because the tracking of a Kanban project is quite different. We demonstrate tools such as burn-up and burn-down charts and parking lots used in feature-driven development. We discuss how to measure success within Agile, including the ever-important metric of customer satisfaction. We conclude with an interview with Agile coach Kent McDonald.
- **Chapter 9—Agile beyond IT**
  This chapter explains how the implementation of Agile affects far more than the IT organization; the entire launch process for new products or enhanced features is now different and applying the four Agile values enhances the ability to deliver to the marketplace. We also showcase how the Agile principles can be applied to other organizations outside IT. Some in Marketing have adopted Agile quite thoroughly, even creating the Agile Marketing Manifesto. Our interview with Travis Arnold, one of the creators of the Agile Marketing Manifesto, completes the chapter.
- **Appendix—John Deere Case Study**
  The appendix presents an interview we conducted with three leaders at John Deere's Intelligent Solutions Group, who have helped lead their journey to become a more Agile organization.

We sincerely hope you enjoy exploring the world of Agile software development. We welcome your feedback and encourage you to visit our web site or follow us on Twitter for more information on Agile.

Sondra Ashmore, Ph.D., PMI-PMP/ACP
@Sondra1130

Kristin Runyan, PMI-PMP, CSPO, CSM
http://www.runyanconsulting.com
@KristinRunyan

Acknowledgments

We would like to extend our sincere appreciation to the following people who have given generously of their time and talents to make important contributions to this book.

- **Interviewees**—Robert (Uncle Bob) Martin, Scott Ambler, Alistair Cockburn, Roman Pichler, Lyssa Adkins, Ellen Gottesdiener, Mary Gorman, Mike Cohn, Tim Ottinger, Kent McDonald, and Travis Arnold
- John Deere, for their interviews and photos
- **Reviewers**—Brad Rasmussen, Brandon Carlson, Hastia Sartika, Steve Sieverding, and Robert Gilbert
- **Subjects in photos**—Anna Runyan, Kristin Runyan, Jacob Runyan, Scott Clarke, Brian Ashmore, and Sondra Ashmore
- **Photographer**—Tim Runyan
- **Blackjack code author**—Brian Ashmore

This page intentionally left blank

About the Authors

Sondra Ashmore, Ph.D., is an IT leader who specializes in Fortune 500 corporations. Her areas of expertise include product management, project management, and new product development for IT offerings. She received her graduate education at Rensselaer Polytechnic Institute in technical communication and management, and at Iowa State in human–computer interaction. Her research focuses on the software development process, both Waterfall and Agile, and explores strategies to optimize the user experience. In 2012, she was recognized as a "Forty under 40" business leader by the *Business Record* and won the Women of Innovation award from the Technology Association of Iowa for business innovation and leadership for her work at IBM. Sondra is certified as a Project Management Professional (PMP), Project Management Institute Agile Certified Professional (PMI-ACP), and Stanford Certified Project Manager (SCPM).

Photo reprinted with permission from the Des Moines Business Record.

Kristin Runyan is a product delivery expert in Des Moines, Iowa, specializing in product management, Agile coaching and training, and leadership. She is certified as a Scrum Master (CSM), Scrum Product Owner (CSPO), Pragmatic Marketing Product Manager, and Project Management Professional (PMP). She is also a 2011 winner of the Women of Innovation award from the Technology Association of Iowa. Kristin got her undergraduate degree at Texas Christian University and her MBA at Saint Louis University. She is an avid blogger at www.runyanconsulting.com, and her Twitter handle is @KristinRunyan. Kristin enjoys living in the Midwest with her husband, two daughters, and black lab.

Photo courtesy of Businessolver.

This page intentionally left blank

This page intentionally left blank

Chapter 2

Organizational Culture Considerations with Agile

Learning Objectives

- Understand organizational culture and why it matters in an Agile implementation
- Dive into ways things might be different in an Agile organization from a developer, manager, and executive viewpoint
- Look at successes and failures in behaviors to see the cultural impacts
- Understand how the Agile principles drive different behaviors in an organization
- Investigate the healthy team dynamics of self-organization teams, continuous improvement, frequent delivery, effective seating arrangements, incorporating virtual resources, and adapting to the changing environment
- Explore how an Agile workplace differs for managers and the ways that they must change with regard to teamwork, trust, and transparency
- Review the role of executives and how their behavior can position an Agile transformation for success with executive alignment, respecting priorities, creating supportive environments for the teams, and driving the right behaviors with metrics

Moving from Waterfall to Agile requires an organizational culture transformation in most companies. This chapter delves into the impacts as well as examples of both successful and unsuccessful ways to handle the culture change. First we understand what organizational culture is and how it influences an organization. Then we look through the eyes of three positions in a company to see how Agile changes the way common situations and challenges are addressed.

What Is Organizational Culture, and Why Does It Matter?

Before we dive into the specific impacts of Agile, let's discuss corporate culture. According to Philip Atkinson, corporate culture "is the infrastructure, the glue that binds together people and processes to generate results . . . The culture should become the major force that propels the organization onward" (Atkinson 2012).

How important is the organizational culture in a company's ability to adopt Agile? Critical. In fact, in the VersionOne annual survey of "The State of Agile Development," they found that culture change is the primary barrier to further Agile adoption at companies (VersionOne 2013, p. 9).

Why is it so hard to change a culture? An organization's culture was not built overnight; it is the accumulation of years of interactions and experiences that have formed into a belief system of how work progresses and how decisions are made. To shift the culture, one must create new experiences and reward those who dare to embrace the new and unfamiliar. The organization must be prepared to examine its old practices with a critical eye and try a new way of doing things. When the new experiences are difficult or uncomfortable, it is common to revert to old habits. Driving true culture change, which Agile requires, needs commitment and nurturing from all levels of the organization.

The Team Members' Viewpoint

The first perspective we want to explore concerns the individual team members that are being placed on an Agile team. These people are typically in development and quality assurance (QA) roles but could include others in the organization. The team role is described in detail in Chapter 4, "Describing the Different Roles." Why would team members want to embrace Agile for their projects? Several key principles within Agile are highly desirable to most individual team members.

What Is Different?

To understand the benefits and cultural impacts of Agile, first we should explore what is actually different with the Agile practices and the typical work environments of pre-Agile organizations.

Self-Organizing Teams

Agile advocates for self-organizing teams, which is a big change for many organizations. Many workplaces hire individuals to work on a specific thing, and even though they are part of an organizational structure, they really act alone. They are responsible for their own work, and their performance is measured based on their individual achievements. That all changes with Agile: Employees come together as a team—not just a set of individuals—and they establish themselves as an entity. They become a group of people driving toward a common goal. One of the first manifestations of this newly created team takes the form of a **working agreement**, described in Chapter 4. The team gets to establish their own norms and rules of engagement without management oversight; this offers developers and testers a great deal of autonomy that may not have existed before Agile.

Another desirable outcome of self-organizing teams is that the team members get to actually **select the tasks** they want to work on during the iteration. Before Agile, many organizations distributed work from the manager to the developer, without the developer getting much say at all on the assigned tasks. The pre-Agile assumption was that managers understood the work and how best to deliver it, so they would assign the necessary tasks to individual developers who would execute on their task. There is a high cost of missed opportunities with this method that Agile addresses: The developers might have a better solution for the business problem, and Agile gives them the opportunity to voice their suggestions and alternatives. Also, developers may have skills in a variety of areas, and to be assigned tasks by someone else does not allow them to choose to work on different items. The added variety to their workload is often appealing. Another benefit to self-organizing teams is the ability to organically cross-train. This happens when one developer wants to learn something new and acts on that desire. When selecting the tasks to work on, developers can request those tasks under the guidance of a more experienced developer, or they may ask to shadow an experienced developer so they can learn. The opportunity to choose their own tasks is a dramatic improvement over the pre-Agile environment.

This concept can be demonstrated in university settings as well. If you are put on a team to deliver a class project with no predetermined task assignments, the team can decide how best to distribute the work: One student might be the

best researcher, one might be the fastest typist, and another might be best at the oral presentation. By allowing the team to determine how work is distributed, you can play to the strengths of each individual and facilitate new learning, which is the point of the next section.

Continuous Improvement

Another big organizational culture impact of Agile from a team member's viewpoint is the ownership of continuous improvement. Within Agile, each team member is responsible for ensuring that problems from a past or current iteration are not carried into the next one; this creates a higher level of engagement from employees, because they have to determine how best to solve any problems or issues existing within the team. This type of continuous improvement applies not just to their code and the products they produce but also to their sense of teamwork. The best teams actively practice reflection, usually through a meeting called a *retrospective*, which is covered in Chapter 8, "Tracking and Reporting." This meeting is sometimes referred to as a postmortem or "Lessons Learned," and it allows for the team to discuss what went well, what did not go well, and what they need to improve. It is important for teams to step away from the day-to-day activities and spend time discussing the team dynamics. Allowing the teams time to reflect on their actions and progress is important within the fast-paced cycle of iterative development (Cohn 2010, p. 213).

The reason why this is a compelling difference from other methodologies is because individuals are no longer waiting to have their problems solved for them—they are actively engaged in the solutions. In the past, a developer or tester might have relied on the manager or someone else in the organization to resolve issues, improve processes, and relay information. Within Agile, the teams are empowered and encouraged to seek out answers and improvements on their own.

To apply this to a university setting, consider teams that are assigned for the entire semester. Within a team, there may be highly organized, highly driven participants as well as others who are not willing or able to put forth as much time and effort. If one of the team members goes to the professor seeking a solution and the professor rearranges the team assignments, that is certainly one way to solve the problem, but it is not the Agile way. Within Agile, the team members would be asked to resolve their differences themselves first. Of course, if the issue is a true performance problem, then the professor may need to intervene, but the first course of resolution within Agile is always within the team.

Frequent Delivery

When developers move into Agile teams, they may experience a profound shift with the expectations of working software and frequent delivery. In many pre-Agile organizations, developers and testers could work on a project for months and months with very little feedback. This is no longer true in an Agile workplace: Teams are asked to deliver tested code very quickly so that stakeholders can review progress and make adjustments if necessary. This shift in expectations can be significant to some developers and testers—particularly perfectionists—who want everything to be precise and thorough before anyone sees it.

For Alistair Cockburn and the Crystal methodology, this principle is the number one priority: In his 2010 blog, titled "Seven Properties of Highly Successful Projects in Crystal Clear," he states:

> The single most important property of any project, large or small, agile or not, is that of delivering running, tested code to real users every few months. The advantages are so numerous that it is astonishing that any team doesn't do it:
>
> - The sponsors get critical feedback on the rate of progress of the team.
> - Users get a chance to discover whether their original request was for what they actually need and to get their discoveries fed back into development.
> - Developers keep their focus, breaking deadlocks of indecision.
> - The team gets to debug their development and deployment processes, and gets a morale boost through accomplishments.
>
> All of these advantages come from one single property, **Frequent Delivery.** (Cockburn 2010)

Obtaining feedback is critical to an organization's ability to course-correct if something is not quite right or does not meet the needs of the customer. Within Waterfall, because working software was not delivered at regular intervals but rather as a "big bang" after months or years of development, gathering this type of interim feedback was difficult. Becoming comfortable with frequent delivery is important for developers and testers because they must truly desire the feedback and be willing to act on it.

Frequent delivery is not always part of the university experience, because many classes have a team project that is due at the end of the semester; however, projects that allow for feedback throughout the semester are easier to manage. For example, if a critical paper is due after month 1, the team can incorporate the feedback from that paper into future deliverables. If the written presentation is due after month 2, then the team can learn if their proposal resonates and practical ways to improve it. By the time they make their oral presentation at the end of the semester, they are more confident in their work because they

have had the opportunity to solicit and incorporate feedback throughout the semester.

Removing the "Us versus Them" Scenarios

In a pre-Agile world, some development teams handed off their code to a testing group, where the QA activities took place; this handoff frequently created an "us versus them" environment, where the developers could criticize the testers based on the types and quality of tests they were running, and the testers could lament the poorly written code that they were expected to debug. By changing the definition of "done"—a concept described in detail in Chapter 6, "Grooming and Planning"—to include testing, the team has a new appreciation for the testing effort. Developers are now collaborating with their testing teammates to create the highest quality software. If an iteration cannot be completed on time because of testing problems, the entire team is responsible.

Testing is one of the most visible manifestations of the Agile concept of teamwork, but many other groups are affected as well. Agile is about working together toward a common goal, and it creates a structure or framework to facilitate collaboration and break down organizational silos. Another area where this is demonstrated is between the product owner and the team. In the past, a lack of clear requirements was often the reason that software was late or inadequate. Now, the product owner, with responsibility for bringing clarity and priority to the requirements, is part of the delivery team. If a developer is unsure of how to proceed, it is his or her responsibility to seek clarity from the product owner, and it is the product owner's responsibility to provide an educated response.

Central to this culture shift is the idea that the team succeeds or fails together. Within Agile, it is impossible for a tester to succeed but the developer to fail, or for the product owner to succeed but the Scrum master to fail. Either the team, the whole team, delivered working software at the end of the iteration, or they did not. If the iteration did not produce working software, it is up to the team to diagnose the problems and work to correct them, previously described as continuous improvement.

In a university setting, this is often demonstrated in group projects where the entire group will receive a single grade. The professor is not likely to entertain conversations about who did what and why the project is difficult—he or she is interested in results, and part of delivering the necessary results is figuring out ways to maximize the talents and motivation of everyone on the team.

Physical Workspace

The collaboration aspect of Agile allows team members to work together and solve problems quickly. When the entire development team is colocated, which

is ideal, then how the seating arrangements are managed can greatly contribute to their effectiveness. The best scenario is for the team to sit together in a type of pod arrangement that easily facilitates spontaneous conversations. Cubicle walls (or even offices) can be torn down so everyone can sit together and see one another (see Figure 2.1). This invites collaboration. The founders of Scrum were very direct on this point: "Use open working environments. Such environments allow people to communicate more easily, make it easier to get together, and facilitate self-organization" (Schwaber and Beedle 2002, p. 39).

By allowing the developers and testers to have easy access to one another, questions or issues that may have taken several e-mails or meetings to resolve can be addressed face-to-face for immediate resolution. The speed of clarification and problem solving that comes from being collaborative provides teams with an excellent opportunity to improve their deliverables.

The idea of colocation works in a university setting as well. Many colleges offer remote degree plans today, which is a great alternative for students who live in rural areas or who want to participate in a program that is not offered locally. The challenge with these sorts of arrangements is on the team assignments, where classmates do not have the opportunity to get together regularly to discuss the project progress and roadblocks. These challenges can be overcome, however, using some of the methods described later in the chapter.

Figure 2.1 *Agile teams working together*

Successes

Now that we understand the differences that Agile brings to individual team members—self-organizing teams, the opportunity for continuous improvement, embracing frequent delivery without organizational silos, and having a collaborative physical workspace—next we examine what organizations do to ensure success or avoid paths to failure.

Team Dynamics

The best self-organizing teams place a high priority on members' knowing each other and creating an environment that plays to the individuals' strengths. What best practices do strong teams use?

First, members get to know one another. Each person has his or her own personality, family situation, areas of expertise, and temperament; by knowing teammates as human beings first and workmates second, team members see themselves as a cohesive unit, poised for success (Adkins 2010, loc. 4841). The most successful team members are the ones who maximize their environment around both personal and professional preferences. For example, if one team member needs to take children to school before work and cannot arrive before 8:30 a.m., then having the mandatory daily stand-up meeting (described in Chapter 8) at 8:15 a.m. would put that team member at a disadvantage. From a professional perspective, if one team member is unskilled at writing documentation, it would be unwise to have that person assume all of the documentation responsibilities. An effective team will address both the personal and professional nuances of their group to maximize everyone's effectiveness and create a desirable work environment; the working agreement described earlier is a great starting place for that clarity. A project and team that are structured around the individuals' personal goals and unique talents will generate the desired commitment (Cohn 2010, p. 216).

The second indicator of success on high-performing teams is their ability to adjust and course-correct. When things do not go as planned, the successful teams find a way to address the conflict in a productive manner.

The most effective teams are the ones that tackle concerns early on, in a respectful and resolution-oriented way. The least effective teams get angry but do not share their frustrations constructively. Ideally, teams operate in a manner that is based on honesty and proactively addresses situations the instant they are problematic. If a team member says or does something that seems disrespectful or too negative, it is best to address that right away. Members of strong teams are honest with one another and if something is not working, the whole team works together for a better alternative.

Incorporating Virtual Resources

Another defining characteristic of high-performing Agile teams is their ability to include virtual and perhaps even offshore (international) resources on their teams. Having team members that are not in your physical location introduces new challenges, and the teams that are able to adapt and incorporate the skills of the virtual team member will enjoy success.

When team members are not colocated, the lack of face-to-face communication is the first hurdle that must be overcome, and this can be accomplished using video tools. Ideally, this means that every meeting is conducted with a video connection so the virtual team members are included in the discussion. Honestly, the structured meetings are the easy part; where virtual team members might miss out is in the organic conversations that happen in the hallway or in the lunchroom. Human beings are always thinking and learning. Sometimes walking away from your desk or leaving the meeting room or bumping into a particular person on the way to the vending machine can spark an idea or help solve a problem. When those moments of inspiration occur, it is vitally important to contact the virtual team members and bring them in on the innovation. Otherwise, the team could advance but the virtual team members would be inadvertently left behind.

How can you create an environment where virtual team members feel as if they are part of the team? The first best practice is to have those people sit with the team at the start of the project for a significant period of time; ideally, this is at least two iterations. It is important for the team to feel as if they "know" the virtual employees, and vice versa; once the initial relationship is established, the virtual team members do not feel so far away. Inside jokes and team banter can happen naturally, with everyone feeling equally involved.

Travel budgets are tight at many companies these days, but having virtual employees come to town for key meetings, such as project kick-offs or quarterly reviews, can help the team bond and can increase the trust between the team members. In fact, to ensure success with distributed teams, companies will likely need to increase—not decrease—their travel budgets (Demarco et al. 2008).

Optimizing the Workspace

Companies and teams that enjoy success in an Agile environment take their workspace considerations seriously and look for options to optimize. Here is a summary of the three key spaces that are required:

- **Individual workstations** should be arranged to facilitate collaboration. Some teams prefer to have distance or walls separating them from other teams to keep noise and distractions to a minimum; other companies

place teams with or near the business units that they support so the sense of a common goal is shared in the space.

- An **Agile (Scrum) room** is a discussion area with white boards for times when something needs to be debated or brainstorming of ideas is required. Ideally, this space is not shared with other teams or the rest of the company, because it should be immediately available when something comes up and the team wants to be able to leave their documentation and drawings on white boards for future reference. One innovative company turned old management offices into Scrum rooms so teams had their own dedicated space.
- Access to **company conference rooms** for the larger meetings, such as Sprint demos and backlog grooming sessions, is also necessary (see Figure 2.2). These often have to be scheduled because they are shared spaces, and each conference room needs access to a projector or Smart Board to display the working software at the end of the sprint.

In some companies, moving the physical space will make a powerful statement about the seriousness of the Agile implementation. If you say you are Agile but you stay in cubicles, how Agile are you? By rearranging the office

Figure 2.2 *Team workspace*

space, you visibly demonstrate your commitment to change. It is worth noting that the nature of Agile means that the physical requirements will change over time. As teams grow and shrink and traffic patterns change and interdependencies between teams morph, it will be necessary to rearrange the office space again.

Exercise

Consider the most productive team that you have worked on. What were the success factors that contributed? What skill sets did your teammates possess? How did your abilities complement each other? What lessons will you try to incorporate on future teams?

Failures/Risks

What traps or mistakes do team members make that can have a negative impact on the Agile implementation? In many ways, they are similar to the traits of successful teams, but they somehow miss the essence and turn into a liability instead of an asset. Let's explore some common mistakes.

Unhealthy Teams

Unlike the high-performing teams, unsuccessful teams typically fall into one or more of the following situations.

First, they fail to self-organize. There are truly people in the workforce today who are more comfortable being told what to do and simply execute on that order. When we ask those people to become more engaged and be part of the solution process, they are reluctant, or incapable of doing so. Some people are conditioned to this behavior, but over time in a supportive and learning culture, they can transform to being key contributors. For others, it is simply in their DNA to take direction from others and not rise to the level of accountability that Agile requires; these people will not be successful on Agile teams, so the organization needs to either find them a different role or allow them to move on.

Another example of the inability to self-organize is where teams demonstrate hostility, bullying, or demeaning behavior toward one or more of their team members. This aggressive behavior cannot be tolerated, and if the team cannot resolve it on their own, then management needs to get involved. Agile workplaces are safe environments where people are encouraged to learn and grow and take chances and deliver great results; but if there is an unhealthy team dynamic, that can be difficult—maybe impossible—to achieve.

Many people have served on unhealthy teams, and the same dynamics that exist in sports or academia can exist in the workplace. Teammates on unhealthy teams

- are unwilling to help or support one another
- refuse to broaden their role by saying things such as "it is not my job to test" or "he owns that piece of code, so it is his problem"
- withhold helpful information or training because they want to be seen as the expert

These unhealthy team dynamics can sabotage an Agile implementation, and it will take a strong Scrum master, coach, or possibly even manager to break down these bad habits.

Inability to Adapt

Agile requires team members to change, and that is uncomfortable for some people. They need to change how they interact, how they do their work, where they sit, and much more. An inability to adapt is a key reason why some team members fail in an Agile transformation. Looking at a specific example, as already mentioned, having a geographically distributed team introduces challenges, and some teams fail to adapt and accommodate. If the team does not make the effort to include the virtual resources in key conversations and meetings, then their ability to contribute is compromised; this can especially be true with international resources. Having sensitivity to time differences when scheduling meetings, creating a work schedule that allows for overlapping hours, and clarifying requirements so no language barriers impede progress are all efforts high-performing teams make. The ones who fail to have the appropriate sensitivity create a work environment that is polarized between those in the building and those outside.

Another example involves making the transition from cubicles to an open concept. When introducing the collaborative work arrangement, worries may range from a colleague who wears too much perfume, to a germophobe who fears being sneezed on, to concerns about background noise and the ability to concentrate. All are legitimate issues that need to be addressed, but none are worth abandoning the benefits of moving to the open, collaborative space. The teams or individuals that refuse to be open-minded about the seating arrangement and constantly complain are sabotaging their Agile implementation. Like all things, if a legitimate concern is affecting safety, then management needs to address it immediately. Otherwise, being adaptable in the seating arrangements demonstrates a commitment to the greater good. An inability or unwillingness

to consider new options and adapt to the evolving needs of the organization can lead to failure.

Lacking Commitment

Another area where team members can experience failure is by lacking a sense of commitment. Within Agile, the team commits to the amount of work that they intend to complete during an iteration. If a team does not feel a sense of obligation to that commitment, the level of success that can be achieved is in jeopardy. Some teams reach the end of an iteration in which they have not completed the required work, but they have an attitude of "oh well, we tried"; these are not successful teams. Failing to honor a commitment should be a disappointment. It should serve as an opportunity to assess what went wrong and how it can be corrected in the future. The teams that are lackadaisical about their commitment will never be truly Agile.

The lack of commitment can reveal itself in a number of ways—failing to complete the work is the most obvious and detrimental—but there are other manifestations as well. Not adhering to the meeting cadence within Agile by grooming, planning, tracking, and demonstrating, all described in Chapter 8, can lead to suboptimized work. It might be easy to become lazy about the daily stand-up meetings and write them off as unnecessary, but that is not in the best interest of the work, the team, or the Agile transformation. Being disciplined about participation and active engagement drives success.

Failing to address issues with team dynamics and allowing bad habits or relationships to continue without proactively confronting them are also examples of lacking commitment. An unwillingness to learn and grow can create a stagnant environment where an individual team member or the entire team stops moving forward and simply becomes complacent. Innovation does not come from places described as complacent or lazy or lacking commitment. Innovation is coupled with true agility, and this comes from dedication, discipline, and a desire for continuous improvements.

Review 1 At this point, the reader should be able to answer Review Questions 1–5.

A Manager's Viewpoint

Agile for team members is full of new opportunities and a chance to contribute in ways that may not have been available in the old environment. The same is

true for managers, but they may feel as though their role is shrinking or becoming less important. In many instances, Agile alters their span of control, which can create unease and a lack of clarity regarding expectations and performance measurements.

What Is Different?

The differences for managers center on how their role has evolved. It can be quite positive, but it is definitely different, and some managers are unable or unwilling to let go of their past responsibilities to embrace the new ways that they can help the team.

Questions, Not Solutions

Perhaps the biggest impact to most managers is that they are no longer responsible for defining the solutions—this now belongs to the team. Many managers have prided themselves on being owners of an application or architecture, so adjusting to the idea of team members making decisions about those applications can be difficult.

The manager is in the pivotal position of being able to facilitate how much a team learns and how quickly they embrace their self-organization. The most effective managers will make the shift from telling to asking. When a team member approaches management and asks, "How can we increase the response times with the database?" the manager in a pre-Agile world would provide an answer or at least a suggestion. In an Agile environment, the best managers will ask questions: "Why do you think the response time is bad?" "What is the customer expecting?" "What data is being retrieved?" "Is that the right amount of data?" "Are any business rules being applied to the query?" And so on. This allows the team to think through the situation to arrive at their own solutions. It might take a bit longer, but by enabling the team to derive the solution, the manager is truly being Agile and positioning the team for success and continuous improvements.

Applying this concept to a university environment often brings memories of favorite professors or key learning moments. When you struggle with a concept and visit the professor, the Agile-minded ones will ask you numerous questions until you arrive at the answer on your own; the less engaged professor might just give you the answer. By asking questions and allowing you to reach the answer on your own, your professor is demonstrating confidence in your intelligence and problem-solving capabilities. Taking the time to work with you to arrive at the answer—rather than just telling you—is an investment of the professor's time and will enable you to feel empowered and capable of reaching

the right conclusion in the future. Professors are often used to this teaching role because they purposely chose their profession; managers may not be inherently good teachers, and displaying this type of faith and investment in employees might feel foreign and unnecessary. The good Agile managers are the ones who adopt a professor-like mind-set focused on continuous improvement and learning.

Clearing Roadblocks

The manager's role shifts to one of clearing roadblocks for the team to enable their success. This is a different role for a manager, and although it is vitally important to the organization, it might not feel very rewarding, at least not at first. If the team has an issue with training or tools or collaboration with other departments, the manager can be a great help navigating the political environment to solve the problem. The difference is that the manager used to be deeply involved in the problem solving, but now he or she is clearing roadblocks to allow the team-led solution to come to fruition. Some managers view this negatively, as though part of their job—or even their worth to the company—has been diminished. This should not be the case: Effective managers in an Agile environment are a tremendous asset. Since they no longer have to focus on every day-to-day detail associated with the project, they can devote time to higher-level activities such as technology architecture and true employee development—areas that are often neglected in the pre-Agile environments. Managers can assist with mapping the business process flow and then make recommendations to improve performance and drive toward simplicity. They also can spend time creating career development plans for employees based on their skills and desires, and they can devote meaningful time and attention to recruiting and making sure each hire adds to the cohesiveness of the team and further position them for success. It is a new role of clearing roadblocks and focusing on higher-level activities, but when done well, it is meaningful and delivers significant value to the organization.

Trusting the Team

Some managers are predisposed not to trust those beneath them in the organization. Douglas McGregor captured this management style in his X-Y theory of management. The Theory X manager believes that employees are inherently lazy and therefore need authoritative supervision and a comprehensive set of controls to manage them (McGregor 1960). Thus, Theory X managers will have a very hard time with an Agile implementation, because they fundamentally believe that self-organizing teams cannot exist. These people tend to act as "command and control," meaning that they dictate the work to be done and even how it

will be done, and they tightly control the environment for that work. Clearly, this type of management attitude directly conflicts with the very core of Agile values. This belief system still exists in the modern workplace and must be addressed to ensure Agile success. If an organization has Theory X types of managers, they may need to move to new positions to not interfere with the Agile implementation. Departments that are very operational in nature often thrive with this type of management; self-organizing development teams do not.

Even if the manager is not a Theory X manager, he or she may still need to make adjustments in trusting the team. For some period of time (perhaps years), the manager has evaluated staff aptitudes based on a non-Agile measure. Often, managers have a hard time envisioning their employees stepping into and embracing new responsibilities and problem-solving techniques. The most effective Agile managers are those who know that the best way to truly embrace Agile is by letting go of their control and allowing the team to learn, and perhaps fail.

A nonworkplace example of this comes with parenting. When children are small, their physical, emotional, and intellectual capabilities are limited, but as they grow, these things obviously evolve. Imagine if parents kept acting as though their children were three-year-olds, even as they grew: The parents would still cut their food into tiny bites, would never dream of allowing them to ride a bike, and would certainly not allow them to bathe themselves. As the children grew, they would become increasingly frustrated with the limited expectations and would either lash out or disengage. The same is true in our workspace: Employees are constantly learning and evolving, and they can and should take on increasing levels of responsibility for their work. Agile supports this evolution, but some managers cannot move beyond their initial assessment of an employee's abilities. If a manager is struggling with an Agile transformation, this is an area to apply some self-reflection: Is the manager actually holding the employees back? If so, a little bit of trust can go a long way.

Successes

Can managers survive the changes in their roles in an Agile environment? Of course, and the successful ones display a few common traits. The best managers embrace the Agile values and principles by endorsing teamwork and trust.

Teamwork

The ideal manager is going to do everything in his or her power to ensure the success of the team. This includes staffing the team for success and ensuring that all roles are covered, assigning the team members full-time, and making sure they have the necessary tools and environment to deliver on their commitments.

One example that Cayman Design encountered needed management assistance to ensure team success. When creating the Scrum teams, there were not enough testing (QA) resources for every team to have a dedicated tester, so the decision was made for two teams to share testing resources. The first iteration went well, and both teams received the testing support that they needed. But in a subsequent iteration, one team ran into a problem, and they relied heavily on the tester to help them diagnose the issue. The other team, therefore, received no QA support during their iteration, and they were unable to deliver working software. Solving this problem was outside of the control of the individual teams because it required a reallocation of resources. The successful Agile manager took ownership of this problem and worked through budget and head count issues with senior management to allocate a dedicated QA resource to each team. The manager's proactive ownership of the situation positioned the teams for success.

A strong Agile manager also allows and encourages team members to work on their own differences. Members of a certain team at Cayman Design visited the manager, complaining of communication issues on the team: They did not know what their team members were working on, they were surprised when things were not completed, and they did not feel informed on roadblocks that were impeding their teammates' progress. Rather than addressing this in the typical management fashion, by calling the team together to discuss it or by speaking to everyone individually, the successful Agile manager pushed the issue back to the team to solve. Agile provides a structure to facilitate team success, and teams need to use those mechanisms on their own.

Trust

An effective Agile manager fundamentally trusts the team to do good work; this is evident in allowing team members to truly own issues and resolve them on their own. Trust is a bit like allowing your children to explore their own ideas, even if you are skeptical that they will work. Here is an example:

Your Scrum master comes to you and says that the team would like to work from home four days a week and be in the office only the one day that they host their Scrum meetings. You, as their manager, have serious reservations about this idea: It contradicts one of the Agile principles about the importance of face-to-face communication, and it will likely reduce the team's ability to collaborate. A non-Agile manager might immediately respond with, "That is out of the question. We would lose far too much collaboration if no one was in the office. Plus, what would our business partners think if they came into our workspace and no one was here?"

A successful Agile manager would likely respond with a question: "That is an interesting proposal. What is the business problem that you are trying to

solve with the work from home idea?" Through this response, the Scrum master can share additional facts and drivers that led the team to think that this was a good idea. The manager can then react to the additional information with more accurate guidance and may even—through asking questions—guide the team to a different alternative. The successful Agile manager assumes the team is trustworthy and that their ideas are valid.

Failures/Risks

The managers that fail to make the transition to an Agile organization typically display common characteristics.

Command and Control

There are managers who simply cannot give up their command and control demeanor; it is how they manage, and it is inherent in their style. It can be similar to asking a Marine drill sergeant to let recruits decide how far they are going to run and how clean their barracks need to be—it is just unnatural to them. When working with a company recently, we encountered this type of personality in the Project Management office. When asked what he liked about his current (non-Agile) role, this person said, "I like being able to manage people, control the environment, and manipulate the teams to get the deliverable that I want." A person with this type of philosophy will have a very hard time adjusting to an Agile environment.

How can managers overcome their command and control tendencies? First, we must accept that some are willing and able to change and some are not, and that the organization must respond accordingly (see the discussion of executive roles in the next section). Many previously controlling personalities have successfully shifted to Agile by understanding its benefits and the important role that they can play in optimizing the team. Some of these techniques have already been mentioned; here are some additional details.

- **Ask questions instead of offering solutions.** If a manager can shift into a questioning mode, he or she can help the team to self-organize and establish trust. This could be as simple as asking "What do you think we should do next?"
- **Direct others to the team.** If someone from another organization has a question or needs information, instead of immediately answering, as command and control personalities tend to do, a manager can encourage that person to ask the team directly. This will position the team as the authoritative source of information.

- **Do not talk.** When a command and control manager attends a meeting, typically the first impulse is to actively contribute to the meeting to advance the discussion. As a start to break down this tendency, a manager can try not to speak during an entire meeting. The silence might be profound, particularly if the team is conditioned to receiving direction. The manager can embrace the silence and let the team members fill it with their ideas and suggestions.

It is difficult to change a way of thinking that likely served managers well throughout their careers, and that is why Agile asks people to stretch outside of their comfort zones. The results can speak for themselves when empowered and self-organizing teams deliver amazing business results to the organization.

Territorial

Some managers truly believe that they "own" a process or an application and that no one should make decisions on that process or application without their involvement and consent. Agile is very collaborative and customer-focused, and if a solution is presented that crosses systems or changes workflows to enhance the experience for the customer, that effort needs to be allowed to proceed without deference to artificial organizational boundaries.

As an example, we had a situation where a manager believed that he owned a gateway service, and to maximize the speed of the gateway application, no business rules could reside in the gateway; all business rule logic needed to be performed by other systems. This is actually a wise architectural guideline to drive performance, but if a team has a need where putting business rules into the gateway might make the most sense, then that discussion needs to be able to proceed without territorialism. The decision may not change, but the ability to have a productive and value-focused conversation is critical to the organization.

Ownership is a difficult challenge within Agile because we want people to feel accountable for their work. Some could view territorialism as ownership, which implies accountability. The difference that Agile presents is that territorialism for the sake of ownership is bad; accountability for the sake of delivering business value to the organization is good. Whenever managers find themselves feeling uncomfortable about what is being suggested for "their" application, the best course of action is to step away from the technical details of the situation and dive into the business problem that they are trying to solve. The better we understand the end users' pain point, the better we can devise solutions, without undue deference to a particular system or workflow. Looking through the eyes of the customer can tear down territorial boundaries quickly.

Team Oversight

Another failure that comes up from time to time involves managers who simply cannot let go of the team: They continue to distribute work to team members, thwarting their ability to self-organize. Managers who attend and actively participate in the daily stand-up are not allowing the team to self-organize, and those who dictate roles on the team are not helping the Agile environment.

How can managers overcome their need to get involved in day-to-day decisions and team oversight? One of the fastest cures to this problem is to stop attending the meetings. This happened at Cayman Design, where a manager chose to stop attending the meetings and would get the necessary information from the transparency Agile afforded. At first, the manager's absence slowed the team down because they believed they needed her to make key decisions. Over time, the team realized that the manager was not going to attend the meetings, so they needed to solve their own problems and come up with their own recommendations. It can be a difficult transition, but there are simple ways to become more Agile. It is important to note that an Agile transformation does not happen overnight: Managers do not move in one motion from being controlling and territorial to being collaborative and enabling, nor do teams accept the accountability of becoming self-organizing and decisive in short order. Organizations must be thoughtful and deliberate with the Agile adoption, always striving to embrace the Agile principles and continuously improve.

Review 2 At this point, the reader should be able to answer Review Questions 6–10.

An Executive's Viewpoint

Executives play a critical role in an Agile transformation, because they hold the power in terms of budget and personnel resources to either back the Agile transformation or reinforce the ways of the past. They also have the opportunity (and obligation) to lead by example, because others in the organization are surely watching to see if situations are handled differently, now that they are an "Agile" organization.

What Is Different?

How an executive's role changes within Agile is interesting. Many executives are the sponsors of the Agile transformation without a deep realization that

their role will also be altered. Outlined next are several examples of how executives are asked to stretch and evolve in an Agile environment.

Embracing Evolving Requirements

Executives are tasked with budgets, milestones, and reporting progress to the board of directors, shareholders, and others who have invested in the success of the company. The executives need to be wise stewards of the financial resources that they control to ensure company success; that is a heavy responsibility, which makes many executives want to chart a clear path before a dime is spent. This type of thinking works well in a Waterfall environment, where we complete our requirements before we utilize our often expensive development resources. However, the ever-increasing pace of change in the marketplace makes it increasingly difficult to map out an entire project before we even begin. Agile asks executives to shift from operating capital budgets with robust (though often inaccurate) business cases to an environment of rapid delivery and inspection. For example, a company wants to replace its mainframe system with a more modern, scalable architecture. In the past, a project would be kicked off to assess every application on the mainframe and what it would take to move that application elsewhere. What are the integration points? the risk factors? the expense? the required resources? the schedule? And much more. The project management team would pore over documentation, create massive spreadsheets, and make thousands of assumptions to present a plan to the executive team. It would likely be a multimillion-dollar project with a lengthy timeline. This type of activity provides security to the executives because they can see the entire project laid out, and they believe they can anticipate the cost to completion. The problem with this approach is that assumptions have been layered onto assumptions within the business plan because there is no way to answer all of the complex questions before beginning an effort of this size.

Agile advocates for a different approach. Let's just try to replace one minor application in the mainframe. We will keep the scope very small, we will deliver frequently, and we will inspect our results often, allowing us to course-correct if necessary. With the learning gained from the first effort, we will embark on the next effort, and so on. The organization gains much more predictability in the incremental deliverables, and the wise Agile executives are courageous enough to present this new philosophy and information flow to their investors.

Respecting the Priorities

Executives in all organizations participate in different conversations than the developers and managers, and this leads to slightly different perspectives. If an

executive has just been grilled by an existing customer over a problem in the product, or by a high-margin prospect who will sign the contract only if a specific feature is added, it is common for the executive to place high priority on that immediate feedback. Keeping customers happy and gaining new business are critical to the organization and its longevity, so executives are justified in their sense of urgency. This becomes detrimental in an Agile organization when the executive priority disrupts the current work that has been fully groomed, as described in Chapter 6, and has been committed to by the development team. Stopping their work in progress is hazardous and often unnecessary. What an experienced Agile executive will do is understand and respect the current priority projects being delivered and will send the request to the appropriate product owner to begin exploring all of the details of the new request. The executive that says "stop everything and work on this" when "this" is likely ill-defined and not fully understood creates a chaotic environment where the teams are not able to deliver on their commitments.

Executives came to be in their positions because they are decisive, action-oriented and results-driven, so their first impulse is to mandate an immediate response. However, respecting the existing priorities, providing the team with an opportunity to clarify the details and examine the best solution, and allowing the team to finish the work currently in flight will lead to much more consistent and predictable results.

Staying the Course

An Agile transformation is challenging for most organizations. Some command and control managers will fight the change, offering dire predictions of failed projects as examples of why this is a bad idea. Developers may not embrace the increased accountability and transparency, and some may choose to leave the organization. There are new expenses in the form of seating arrangements, training, and Agile tools that may stress the budget. Agile transformations also have a history of bringing chronic issues that the organization has ignored for years to the surface where they must be confronted. All of these are reasons why an executive might abandon the effort and simply revert to what is comfortable (but ineffective). Any change worth making is going to require effort, and Agile is no different. The strong Agile executive will work through these issues without wavering on the commitment to Agile.

Let's examine these instances individually to identify the best reaction by a truly Agile executive.

- **The command and control manager is uncomfortable.** This often takes the form of bringing up the multitude of ways that Agile could fail—the teams cannot self-organize, our clients are too demanding, our software

is too complex, we are impeded by government regulations, our business partners are too inexperienced, and so on. Each of these hurdles, and many others, has been overcome by organizations with Agile. The Agile executive needs to resist the urge to slow down the implementation and carefully think through every possible scenario. Instead, he or she should embrace the "inspect and adapt" mantra of Agile by moving rapidly and carefully forward and learn from every decision. Constantly inspect the implementation and make course corrections as necessary. Do not let fear of what *could* happen delay the positive outcomes of what *will* happen.

- **Developers self-select out of the organization.** There are certainly people in every organization that we believe we cannot live without; their domain expertise or years of experience make them assets to the organization. Agile supports these experts and provides them with opportunities to contribute in ways they may never have before. The employees that see and desire the collaboration and accountability associated with Agile are critical to the longevity of the organization. Those that are threatened by Agile and want to withhold their expertise or refuse to try new alternatives will ultimately hold the organization back. Certainly, their departure is disruptive, but it can and should be managed by an Agile executive. Creating an environment of continuous learning involves identifying and rewarding those who want to join the transformation.
- **New expenses.** Agile does not break a budget, nor does it come for free. If you are truly committed to changing an organization's culture, then spending money on training and tools and possibly even new hires is an investment—not an expense. Asking an organization to convert to Agile without any additional budget is not demonstrating the necessary commitment to the change. Agile executives need to "put their money where their mouth is" and authorize the appropriate expenditures to ensure success.

An Agile executive will stay the course when confronted with issues and concerns. Making an Agile transformation cannot be optional to the organization: Either the executives are committed or they are not. There can be no wavering, because the minute an executive indicates that it is acceptable to continue with the old ways of doing business, many people will fall back into their old, comfortable patterns that will not deliver the desired results.

Successes

Agile executives that are committed to the transformation can chart a meaningful path for their teams by focusing on the right things that will drive true culture change.

Focus on Sustainability

One of the 12 Agile principles states that "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely." Promoting this is a critical success factor for Agile executives.

We need to avoid burnout amongst the teams. When organizations adopt Agile, there is typically a good deal of enthusiasm and a desire to work hard and deliver compelling results. Then as the Agile implementation continues, some teams can suffer from long hours, which can create an unsustainable environment. Bob Hartman wrote about this in a July 2009 blog:

> If the pace of the team is not sustainable several undesirable effects are likely to occur:
>
> - Defects will increase. Tired teams let more defects through.
> - Work output will decrease. Tired teams do less work in more time!
> - Morale will drastically decrease. This may lead to employee turnover at a most unfortunate time in the project.
> - The blame game will become common. (Not our fault you didn't say X. I said X. Did not. Did so . . .)
> - The team starts to abandon good practices for those that "seem" faster. Sorry, but test-driven development (TDD) is actually faster than just writing the code and throwing it over the wall to QA! (Hartman 2009)

The responsibility for sustainability often lies with executives because they can control the resources and, to some extent, the expectations of the organization. If executives make commitments to customers that are unreasonable or fail to provide teams with the tools and training that they need, then the executives are contributing to an unsustainable environment where good people can become frustrated, which either affects the quality of their work or may even encourage them to leave the organization. Most employees want to work hard and accomplish significant goals, but they require an environment that facilitates their success.

Focus on Technical Excellence

Another of the 12 Agile principles focuses on technical excellence by stating: "Continuous attention to technical excellence and good design enhances agility." A strong Agile executive can have a tremendously positive impact on this idea by making sure the teams have both the time and the resources to make wise architecture decisions that will allow the application to scale and perform. Executives who want teams to rush to designs or build point solutions are not allowing them to do their best work. Technical excellence pays dividends not

just today but into the future, because a well-designed platform can handle evolving requirements and additional features. A poorly architected solution will result in convoluted designs to accomplish future goals. Jeff Sutherland, a signer of the Agile Manifesto and a creator of these principles, finds that one of the biggest remaining problems for Agile teams is that they do not demand technical excellence (Sutherland 2011).

An Agile executive has an opportunity to influence this, just as a university professor does. By forcing team members to slow down enough to think things through—and giving them the time and latitude to do so—both professors and executives encourage thoughtfulness and deliberate decision making. Rushing and setting unrealistic expectations compromise technical excellence.

Focus on Simplicity

There are many wonderful quotes about the value of simplicity and how hard it is to achieve, including one by Leonardo da Vinci: "Simplicity is the ultimate sophistication" (Goodreads 2013).

An Agile executive that understands and values simplicity can propel the organization forward. Simplicity can be utilized in an Agile implementation in several ways. The first is with the implementation itself. Being flexible in an Agile implementation means embracing the unknown and trying new things. If an organization tries to overplan an Agile transformation and think of every possible roadblock that could be encountered, they will strive for perfection and get stuck in a lengthy (and Waterfall-like) planning stage.

The Agile organization that understands simplicity will try different small steps to learn what will work best for the organization. Continuous learning by inspecting and adapting allows for a simple approach to Agile.

The second way that an Agile executive can embrace simplicity is with the corporate or product objectives. When goals are too ambitious or expectations are too varied, it leads to an organization that is fragmented and chasing too many disparate deliverables.

Steve Jobs once said, "People think focus means saying yes to the thing you've got to focus on. But that's not what it means at all. It means saying no to the hundred other good ideas that there are" (Griggs 2012).

An Agile executive knows that simplicity and precise focus enable the teams to deliver great work.

Failures/Risks

How can Agile executives fail when it comes to an Agile implementation? There are many things that even well-meaning executives do that sabotage the Agile efforts of their teams.

Not Honoring Commitments

One of the most common mistakes that executives make is not allowing the teams to honor their commitments. Looking at Scrum as our example, executives need to adhere to the sacredness of the sprint. What does this mean? It means that once a sprint goal is committed to, there should be no changes to that goal. This is especially difficult for executives because there is always some crisis or customer request or great idea that an executive wants the team to investigate immediately, and there is a tendency for executives to pull the team off their sprint commitment to work on the latest critical task. The executives that refrain from allowing their urgent requests to disrupt the team are more successful in their Agile adoptions. Truly, it is a sign of executive commitment to the process and the methodology if they honor this one aspect. Mike Cohn (2010) is very firm on this issue: "Nothing is allowed to change within the sprint. The team commits to a set of work on the first day and then expects its priorities to remain unchanged for the length of the sprint" (p. 279).

Are there instances where the sprint must be interrupted? Absolutely. If one of your production servers goes down and your developers need to restore the system, then that is clearly a priority over new development. If you are under attack from hackers (such as a DDoS attack), then the developers may need to work on an immediate patch. Those situations notwithstanding, there are many more instances where the "crises" can and should wait while the team works on their current sprint commitment.

A strong Agile executive should have the power and conviction to keep fellow executives in line so the sacredness of the sprint can remain intact. Executives who have not fully embraced Agile or do not appreciate the impacts of their decisions on the organization can create chaos and confusion by not honoring the teams' commitments.

Not Engaging the Rest of the Organization

Another common failure of an Agile executive is not enlisting the support of the other departments within the company. Often the Agile executive sponsor is the chief information officer (CIO) or another leader in the IT organization, which is where most Agile implementations start. It is imperative that the CIO or Agile sponsor capture the attention and support of the other department leaders to ensure the success of the implementation. This is particularly critical with "the business": If the Agile implementation is viewed as an IT-only endeavor, then the organizations that need to contribute clarity regarding business value and priority may not actively participate, which can doom the Agile efforts. The people in marketing or product management or various customer-facing departments have the best sense of the marketplace, competitive environment, and customer behavior, and therefore are the best people to make the prioritization decisions so the

most important and valuable efforts are worked on first. If those organizations do not participate in the Agile transformation, then the opportunity for success is greatly diminished. The Agile executive sponsor needs to illustrate to peers the importance of their involvement and the value that they will derive from participating. Gaining their commitment often requires executive alignment and usually cannot be accomplished by those lower in the organization. The time commitment and the changes to the way that work progresses through to completion will affect many organizations, and their buy-in is absolutely critical. An ineffective Agile executive may ask the team to solicit the necessary involvement from the other organizations, but this passive approach typically does not deliver the appropriate visibility and alignment. The Agile executive sponsor needs to convince his or her peers that Agile is right for the whole organization.

Valuing the Wrong Metrics

Executives play such a critical role in an Agile transformation, and often they may not realize the messages that are sent to the organization when decisions are made, even with the best intentions; metrics are a perfect example of this concept. A well-meaning executive can completely derail an Agile implementation by trying to measure the wrong things. Let's look at a few examples.

- Actual time taken to complete a task vs. estimated time
- Velocity of Team A vs. Team B
- Number of stories with acceptance criteria vs. those without

Each of these metrics may sound reasonable, but they could drive the wrong behavior. With the first example, one might wonder: What does the organization value, working software? or precise estimates? Of course, Agile would like the estimates to be relevant and close to the level of effort required, but we want our developers to spend their time writing great software, not doing extensive research to ensure accuracy in their estimates. Defining the wrong metric could force developers and testers to spend time in unproductive efforts.

In the second example, velocity, or the amount of work that a team can accomplish during a sprint (described in detail in Chapter 6), is a tool used to predict the amount of work that a team can commit to, not as a measure of productivity. First, two teams could use different measures of effort—a medium-sized task or an eight-point story (also described in Chapter 6) might mean very different things to different teams. Second, a dip in velocity might have nothing to do with productivity: If Team B has three developers at a training class, then the amount of work they can accomplish is reduced, but not because the team is less productive.

Finally, with the third example, the premise behind Agile requirements gathering is that we are learning and negotiating on the best way to deliver maximum business value. If we are measuring the performance of how stories are written, we may unknowingly reduce the conversation about that story.

Agile executives need to be very careful about the metrics they choose to ensure that incentives are provided for the correct behaviors. Measuring success in an Agile environment is different from traditional metrics, as we now place a heavy emphasis on customer satisfaction, frequent delivery of working software, and continuous learning.

Review 3 At this point, the reader should be able to answer Review Questions 11–15.

Conclusion

The organizational culture impacts to an Agile transformation are profound. Successful implementations need support from the team members, management, and executives to embrace new ways of completing work and collaborating. Every role in the organization will be affected in some way, and by understanding what is different and what drives success in each role, we are better positioned for the increase in productivity, responsiveness, and customer satisfaction that can be delivered by becoming Agile.

Summary

- Organizational culture is the accumulation of years of interactions and experiences that have formed into a belief system of how work progresses and how decisions are made. To shift the culture to Agile, the organization must be willing to embrace new roles, workflows, and definitions of success.
- Self-organizing teams have working agreements, select the work that they will own, cross-train and support one another, and create a dynamic that plays to the strengths of each individual.
- Continuous improvement on teams means that members can solve their own problems without outside intervention and they are working to improve not only the product but also team effectiveness.
- Frequent delivery of working software is central to Agile success. This creates a meaningful feedback loop for the team to incorporate.

- Agile breaks down organizational silos and creates teams where developers, testers, and requirements owners all depend on each other for shared success.
- In Agile, teams should be colocated to facilitate collaboration. This can be challenging for teams with virtual or international participants, so optimizing the physical workspace and collaboration tools is a priority.
- High-performing teams have excellent team dynamics where individuals know and respect each other and can resolve differences productively.
- Effective teams are adaptable to the changes that come with continuous improvement in Agile, and they demonstrate a commitment to completing the deliverables even when circumstances are not ideal.
- Managers in an Agile transformation have to make adjustments as well, and some managers will struggle to embrace their new roles and enable the teams.
- One area where managers can assist an Agile implementation is to refrain from offering solutions to the teams. Instead, they should ask thoughtful questions to help the teams reach their own conclusions.
- Another way that managers can accelerate an Agile deployment is to clear roadblocks for the team. This is particularly helpful when it comes to resource and staffing discussions, because the manager may be able to address impacts to the budget more effectively than the team can.
- When it comes to trusting the team, some managers follow Douglas McGregor's Theory X style of management, where they believe employees are lazy and need oversight to monitor their performance and behavior. This style of management will struggle to adopt Agile.
- Executives in organizations adopting Agile will find that their roles have changed too. Since executives often hold the budget resources and the decision-making power, their involvement in the transformation is critical.
- Executives can support Agile by understanding and embracing evolving requirements. Those that want comprehensive plans and business cases will fail to reap the benefits of iterative development, rapid feedback loops, and continuous learning.
- Executives are in a unique position with Agile to respect the commitments and priorities. They hear complaints and feedback from a wide variety of stakeholders and may be inclined to immediately act on that feedback without understanding the disruption and negative impacts to the team.
- Executives that truly want Agile to be a part of the ongoing organizational culture need to dedicate budget dollars to its implementation; it

might be with regard to tools, training, staffing, consulting, or a number of other things. A transformation of this magnitude requires investment.

Review 4 At this point, the reader should be able to answer Review Questions 16–20.

Interview with Scott Ambler

Scott W. Ambler is the Senior Consulting Partner of Scott Ambler + Associates, working with organizations around the world to help them to improve their software processes. He provides training, coaching, and mentoring in disciplined Agile and Lean strategies at both the project and organizational level. Scott is the founder of the Agile Modeling (AM), Agile Data (AD), Disciplined Agile Delivery (DAD), and Enterprise Unified Process (EUP) methodologies. He is the (co)author of several books, including *Disciplined Agile Delivery*, *Refactoring Databases*, *Agile Modeling*, *Agile Database Techniques*, *The Object Primer* (3rd ed.), and *The Enterprise Unified Process*. Scott is a senior contributing editor with *Dr. Dobb's Journal,* and he blogs about DAD at http://DisciplinedAgileDelivery.com. Scott is also a Founding Member of the Disciplined Agile Consortium (DAC), the certification body for disciplined Agile. He can be reached at scott@scottambler.com, and his web site is http://scottwambler.wordpress.com.

Kristin and Sondra: How do you know when a company is culturally ready to adopt Agile?

Scott: My initial thought was "you just know," but clearly that's not the answer you're looking for. People, at all levels of IT and at least in key positions within the business, need to recognize that they need to deliver IT solutions more effectively. They need to recognize that the old, documentation-heavy ways simply aren't working, and in many cases they need to stop lying to themselves about this. They need to recognize that greater collaboration, greater flexibility, and the need to embrace some kinds of uncertainty are the order of the day.

Kristin and Sondra: What are the most significant differences between Agile and Waterfall, from a cultural perspective?

Scott: Agile focuses on collaborative, iterative, high-value activities that focus on producing solutions that meet the needs of stakeholders. The challenge is that Agile teams need to be skilled and disciplined enough to pull this off, while the stakeholders need to be actively involved with the Agile teams. Waterfall teams often take documentation-heavy approaches in the name of risk reduction. The challenge is that Waterfall strategies prove to be higher risk in practice, but because they involve so many perceived "checks and balances," the people involved are unable to recognize the very clear risks they are taking on.

Kristin and Sondra: How important are things like the seating arrangements for a successful Agile adoption?

Scott: Communication and collaboration are key success factors on Agile projects, so how people are physically organized is important. I've explored this issue in several surveys [see http://Ambysoft.com/surveys/], and it's clear that the closer people are to one another, including to stakeholders, the higher the success rates of project teams. Even something as simple as putting people in cubes can lower the success rate.

Kristin and Sondra: How do successful teams integrate virtual (and perhaps global) team members?

Scott: The best strategy is to not take on these sorts of risks at all. If you choose to, do so with your eyes wide open. Try to have distributed subteams, not dispersed individuals. Fly key people around between sites. Fly key people together at critical times in the project, particularly at the beginning when you're making fundamental strategy decisions. Adopt communication technologies, in particular video conferencing. And finally, adopt distributed development tools, such as IBM Jazz or Microsoft TFS.

References and Further Reading

Adkins, Lyssa. (2010). *Coaching Agile teams: A companion for Scrum masters, Agile coaches, and project managers in transition*. Boston: Addison-Wesley. Kindle edition.

Ambler, Scott, and Lines, Mark. (2012). *Disciplined Agile delivery: A practitioner's guide to Agile software delivery in the enterprise*. Boston: IBM Press.

Atkinson, Philip. (2012). Creating culture change. http://www.philipatkinson.com/Philip-Atkinson-CreatingCultureChange.pdf.

Beck, Kent. (2000). *Extreme Programming explained*. Boston: Addison-Wesley.

Cockburn, Alistair. (2006). *Agile software development: A cooperative game* (2nd ed.). Boston: Addison-Wesley.

Cockburn, Alistair. (2010). Seven properties of highly successful projects in Crystal Clear. http://alistair.cockburn.us/7+Properties+of+Highly+Successful+Projects+from+Crystal+Clear.

Cohn, Mike. (2010). *Succeeding with Agile: Software development using Scrum*. Boston: Addison-Wesley.

Demarco, Tom, et al. (2008). *Adrenaline junkies and template zombies*. New York: Dorset House.

Derby, E., Larsen, D., and Schwaber, K. (2006). *Agile retrospectives: Making good teams great*. Frisco, TX: Pragmatic Bookshelf.

Goodreads. (2013). https://www.goodreads.com/author/quotes/13560.Leonardo_da_Vinci.

Gower, Bob. (2013). *Agile business: A leader's guide to harnessing complexity*. Boulder, CO: Rally Software Development.

Griggs, Brandon. (2012). Ten great quotes from Steve Jobs. http://www.cnn.com/2012/10/04/tech/innovation/steve-jobs-quotes.

Hartman, Bob. (2009). New to Agile? Work at a sustainable pace. Blog entry, July 24. http://www.agileforall.com/2009/07/new-to-agile-work-at-a-sustainable-pace.

Layton, Mark C. (2012). *Agile project management for dummies*. Indianapolis, IN: Wiley.

McGregor, Douglas. (1960). *The human side of enterprise*. New York: McGraw Hill.

Schwaber, Ken, and Beedle, Mike. (2002). *Agile software development with Scrum*. Upper Saddle River, NJ: Prentice Hall.

Schwaber, Ken, and Sutherland, Jeff. (2012). *Software in 30 days: How Agile managers beat the odds, delight their customers, and leave competitors in the dust*. Hoboken, NJ: Wiley.

Sutherland, Jeff. (2011). Ten year Agile retrospective: How we can improve in the next ten years. http://msdn.microsoft.com/en-us/library/hh350860(v=vs.100).aspx.

VersionOne. (2013). 8th annual state of Agile survey. http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf.

Review Questions

Review 1

1. In what ways does self-organization change the day-to-day life of a developer?
2. Who should the team look to first to solve their problems?
3. Why would frequent delivery of working software make a developer uncomfortable?
4. What do successful teams do to incorporate virtual team members?
5. How does altering the seating arrangements change team dynamics in Agile?

Review 2

6. Instead of offering solutions to the team, what should an Agile manager do?
7. What is an example of a type of roadblock that an Agile manager could clear?
8. What are the characteristics of a Theory X manager, according to McGregor?
9. How can an Agile manager demonstrate trust in a team?
10. What are some managerial traits that are incompatible with Agile?

Review 3

11. What are examples of items that require additional budget from an Agile executive?
12. How can executives help with sustainability on Agile teams?
13. Why might an executive want to change the priorities for an Agile team immediately?
14. Who typically serves as the executive sponsor of an Agile implementation?
15. What are examples of metrics that drive the wrong behavior, and why?

Review 4

16. What are some of the practices to ensure that international resources are effectively integrated into the team?
17. Identify several roles outside of the IT organization that are affected by an Agile transformation.
18. Why is it so important to enable teams to honor their commitments?
19. Why would someone choose to leave the organization (resign) rather than move to an Agile environment?
20. What should be done when an aspect of the Agile transformation is not working or delivering the desired results?

This page intentionally left blank

Index

Note: Page numbers followed by *f* indicate figures; those followed by *n* indicate notes.