Agile Project Management

Jim Highsmith

Foreword by Israel Gat

The Agile Software Development Series
Cockburn • Highsmith
Series Editors

Creating
Innovative
Products

Second Edition

Library of Congress Cataloging-in-Publication Data

Highsmith, James A., 1945-

Agile project management : creating innovative products / Jim

Highsmith.

p. cm.

Includes bibliographical references and index.

ISBN-13: 978-0-321-65839-5 (pbk. : alk. paper)

ISBN-10: 0-321-65839-6 (pbk. : alk. paper)

1. Software engineering. 2. Agile software development--Management.

I. Title.

QA76.758.H54 2010

005.1--dc22

2009019147

ISBN-13: 978-0-321-65839-5
ISBN-10: 0-321-65839-6

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

First printing July 2009

Editor-in-Chief
Karen Gettman

Executive Editor
Chris Guzikowski

Senior Development Editor
Chris Zahn

Managing Editor
Patrick Kanouse

Project Editor
Mandie Frank

Copy Editor
Margo Catts

Indexer
Tim Wright

Proofreader
Kathy Ruiz

Publishing Coordinator
Raina Chrobak

Cover Designer
Louisa Adair

Compositor
Mark Shirar

Foreword

We live in an era characterized by information overload. This statement itself is now a cliché. However, it bears repeating because in virtually every field we are still drowning in information and struggling for meaningful knowledge. Even in the relatively nascent field of agile project management and development we already need to filter "noise," connect the dots, and integrate diverse disciplines. Jim Highsmith's *Agile Project Management* is a timely book for the drowning agilist in need of insights that make Agile principles actionable.

The first challenge of the struggle with information is to keep up with and not become overwhelmed by the state of the art. The coming of age of agile methods has led to exponential growth of publications on the subject—books, essays, conference proceedings, blogs, wikis, and, yes, tweets. This fast growth is compounded by the accelerating pace of new practices that get introduced by enthusiastic communities of agilists. Putting our hands around the Agile topic nowadays requires a significant investment of time and effort. Agile Project Management fully covers the whole agile landscape, in depth, in a single, easy-to-read volume.

The second level of information explosion is the day-to-day struggle of the agilist to keep the artifacts produced by the software engineering process at the just-enough level. This is not merely a matter of keeping the template zombies in check. As agile methods get mainstream traction, it's hard to resist the temptation to add "just one more artifact" to fit in with established portfolio management and life cycle processes. Blind adherence to additional process and documentation requirements can often be problematic, but is especially troublesome when such processes fail to take into account how elegant and powerful agile principles can be in sticking to the bare minimum. Jim Highsmith's book demonstrates how to maintain the effectiveness of software development without sacrificing efficiency of ideation, design, coding, and testing. It guides the reader to the point at which the code starts speaking for itself louder and clearer than any verbose reports about the code.

Agile project management will appeal to both novice and expert in just about any discipline that touches agile methods. It reaches a rare level of broad applicability by emphasizing principles over practices. The book does not dogmatically tell the reader what to do in a "one size fits all" manner. Rather, it gives the reader the tools to observe, characterize, and analyze his or her particular circumstances and then to determine the appropriate method to practice. It teaches the practitioner how to implement agile as a process platform, how to tailor his or her agile work to business needs, corporate cultures, and project imperatives in a way that prescriptive advice can never accomplish. Moreover, it explains how the process platform could and should be used by both customer and vendor.

If I were to choose a subtitle for the book, I would pick the phrase "The Thinking Man's Guide to Agile Project Management." The book challenges the reader to look at what he or she is practicing through fresh eyes. In the course of so doing, the reader will assimilate the critical role agile plays in fostering innovation and creating value through affordable experimentation. He or she will develop the ability to facilitate the application of agile at the enterprise level by balancing empowerment, performance management, and governance considerations on a large scale.

How appropriate it is that Agile Project Management is being published at a time when the macro-economic situation poses so many formidable challenges. The book clearly articulates the "secret sauce" for corporate success amidst the crisis: relentless innovation through impassioned employees. Applied in this manner, agile can become a core discipline, revolutionizing the product life cycle all the way from inception through evolution to eventual retirement, and transforming the company all the way from the R&D lab through its value chain partners to the customer.

Israel Gat
Senior Consultant, Cutter Consortium
July 2009
Austin, Texas

Preface

When the Manifesto for Agile Software Development (www.agilealliance.org) was written in spring 2001, it launched a movement—a movement that raced through the software development community; generated controversy and debate; connected with related movements in manufacturing, construction, and aerospace; and been extended into project management.

The impetus for this second edition of *Agile Project Management* comes from four sources: the continuing pace of business change, the maturing of the agile movement over the last five years, the trend to large and distributed agile projects, and the formation of a project management organization for agile leaders (the Agile Project Leadership Network).

An article titled "There Is No More Normal" introduces a series of Business Week articles on game changing ideas. It quotes John Chambers, CEO of Cisco Systems, "Without exception, all of my biggest mistakes occurred because I moved too slowly."[1] If there is no more normal then adapting to change, quickly becomes a requisite skill for improving organizational performance. Agile project management enhances a team and an organization's ability to deal with change and the abnormal.

The essence of this agile movement, whether in new product development, new service offerings, software applications, or project management, rests on two foundational goals: delivering valuable products to customers and creating working environments in which people look forward to coming to work each day.

Innovation continues to drive economic success for countries, industries, and individual companies. Although the rates of innovation in information technology in the last decade may have declined from prodigious to merely lofty, innovation in areas such as biotechnology and nanotechnology are picking up any slack.

[1] McGregor, Jena. "There Is No More Normal," *Business Week* (March 23 and 30, 2009).

New technologies such as combinatorial chemistry and sophisticated computer simulation are fundamentally altering the innovation process itself. When these technologies are applied, the cost of iteration can be driven down dramatically, enabling exploratory and experimental processes to be both more effective and less costly than serial, specification-based processes. This dynamic is at work in the automotive, integrated circuit, software, and pharmaceutical industries. It will soon be at work in your industry.

But taking advantage of these new innovation technologies has proven tricky. When exploration processes replace prescriptive processes, people have to change. For the chemist who now manages the experimental compounding process rather than design compounds himself, and the manager who has to deal with hundreds of experiments rather than a detailed, prescriptive plan, new project management processes are required. Even when these technologies and processes offer lower cost and higher performance than their predecessors, the transformation often proves difficult.

Project management needs to be transformed to move faster, be more flexible, and be aggressively customer responsive. Agile Project Management (APM) answers this transformational need. It brings together a set of principles, practices, and performance measures that enable project managers to catch up with the realities of modern product development.

The target audience for this book is leaders, those hardy individuals who shepherd teams through the exciting but often messy process of turning visions into products—be they software, cell phones, or medical instruments. Leaders arise at many levels—project, team, executive, management—and APM addresses each of these, although the target audience continues to be project leaders. APM rejects the view of project leaders as functionaries who merely comply with the bureaucratic demands of schedules and budgets and replaces it with one in which they are intimately involved in helping teams deliver products.

Four broad topics are covered in Agile Project Management: opportunity, values, frameworks, and practices. The opportunity lies in creating innovative products and services—things that are new, different, and creative. These are products that can't be defined completely in the beginning but evolve over time through experimentation, exploration, and adaptation.

The APM values focus helps create products that deliver customer value today and are responsive to future customer needs. The frameworks, both enterprise and project, assist teams in delivering results reliably, even in the face of constant change, uncertainty, and ambiguity. Finally, the practices—from developing a product vision box to participatory decision making—provide actionable ways in which teams deliver results.

In this second edition of *Agile Project Management*, the five major new or updated topics are agile values, scaling agile projects, advanced release planning, project governance, and performance measurement. Chapters 2–4 have been rewritten around three summarizing value statements: delivering value over meeting constraints, leading the team over managing tasks, adapting to change over conforming to plans. The "Scaling Agile Projects" chapter has been completely revised to reflect the growth and development of the agile movement over the past five years. A new chapter has been added to encourage teams to place more attention on release planning. Finally, chapters on the organizational topics of project governance and changing performance measurement have been added.

In the long run, probably the most important addition is the new perspective on performance measurement. We ask teams to be agile, and then measure their performance by strict adherence to the Iron Triangle: scope, schedule, budget. This edition of APM proposes a new triangle—an Agile Triangle, that consists of value, quality, and constraints. If we want to grow agile organizations, then our performance measurement system must encourage agility.

Finally, although Agile Project Management can be applied to a wide range of product development efforts—and examples of this range are included, the book's primary product emphasis is software development.

Jim Highsmith
July 2009
Flagstaff, Arizona

Introduction

Agile Project Management (APM) contains four focal points: opportunities created by the agile revolution and its impact on product development, the values and principles that drive agile project management, the specific practices that embody and amplify those principles, and practices to help entire organizations, not just project teams, embrace agility.

Chapter 1, "The Agile Revolution," introduces changes that are occurring in product development—from cell phones to software—and how these changes are driving down the cost of experimentation and fundamentally altering how new product development should be managed. The chapter outlines the business objectives of APM and how organizations need to adapt to operating in a chaotic world.

Chapters 2–4 describe the values and principles that actuate APM. These core agile values are articulated in the *Declaration of Interdependence* and the *Manifesto for Agile Software Development*. For clarity and simplicity, three summarizing core values—Delivering Value over Meeting Constraints, Leading the Team over Managing Tasks, Adapting to Change over Conforming to Plans—are introduced, and each is discussed in a chapter.

Chapters 5–10 cover APM frameworks and individual practices. Chapter 5 describes both an agile enterprise framework (Project Governance, Project Management, Iteration Management, Technical Practices) and the phases in the Agile Delivery Framework (Envision, Speculate, Explore, Adapt, Close). Chapters 6–10 identify and describe the practices in each of the phases. Chapter 8 covers advanced release planning and includes a section on value point calculation.

Chapter 11, "Scaling Agile Projects," examines how agile principles are used, together with additional practices, to scale APM to larger projects and larger teams. Scaling covers both organizational and product-related practices.

Chapter 12, “Governing Agile Projects,” begins the transition from topics about agile projects to agile organizations. This chapter addresses the key management and executive issues around project governance and the need to separate governance from delivery operations.

Chapter 13, “Beyond Scope, Schedule, and Cost: Measuring Agile Performance,” continues the emphasis on agile organizations. It raises the issue that measuring success based on scope, schedule, and cost must change. The Agile Triangle, introduced Chapter 1, is examined in detail as a new way of thinking about performance measurement to encourage agility.

Chapter 14, “Reliable Innovation,” underscores how APM helps address the changing nature of new product development, summarizes the role of the agile project leader, and reflects on the need for conviction and courage in implementing agile project management.

Conventions

One presentation technique used in this edition is highlighting key points or quotes in a box and italicizing the statement as illustrated.

> *Presented in a box and words italicized.*

The Agile Software Development Series

Out of all the people concerned with agility in software development over the last decade, Alistair Cockburn and I found so much in common that we joined efforts to bring to press an Agile Software Development Series based around relatively light, effective, human-powered software development techniques. We base the series on these two core ideas:

- Different projects need different processes or methodologies.
- Focusing on skills, communication, and community allows the project to be more effective and more agile than focusing on processes.

The series has the following main tracks:

- *Techniques to improve the effectiveness of a person who is doing a particular sort of job*. This might be a person who is designing a user interface, gathering requirements, planning a project, designing, or testing. Whoever is performing such a job will want to know how the best people in the world do their jobs. *Writing Effective Use Cases* (Cockburn 2001) and *Patterns for Effective Use Cases* (Adolph et al. 2003) are individual technique books.
- *Techniques to improve the effectiveness of a group of people*. These might include techniques for team building, project retrospectives, collaboration, decision making, and the like. *Collaboration Explained* (Tabaka 2006), *Improving Software Organizations* (Mathiassen et al. 2002) and *Surviving Object-Oriented Projects* (Cockburn 1998) are group technique books.
- *Examples of particular, successful agile methodologies*. Whoever is selecting a base methodology to tailor will want to find one that has already been used successfully in a similar situation. Modifying an existing methodology is easier than creating a new one and is more effective than using one that was designed for a different situation. *Scaling Lean & Agile Development* (Larman 2008), *Scaling Software Agility* (Leffingwell 2007), *Crystal Clear* (Cockburn 2004), *DSDM: Business Focused Development* (DSDM Consortium 2003), and *Lean Software Development: An Agile Toolkit* (Poppendieck and Poppendieck 2003) are examples of methodology books.

Three books anchor the Agile Software Development Series:

- This book, *Agile Project Management*, goes beyond software development to describe how a variety of projects can be better managed by applying agile principles and practices. It covers the business justification, principles, and practices of APM.
- *Agile Software Development Ecosystems* (Highsmith 2002) identifies the unique problems in today's software development environment, describes the common principles behind agile development as expressed in the *Agile Manifesto*, and reviews each of the six major agile approaches.

Chapter 4 Adapting over Conforming

A traditional project manager focuses on following the plan with minimal changes, whereas an agile leader focuses on adapting successfully to inevitable changes.

Traditional managers view the plan as the goal, whereas agile leaders view customer value as the goal. If you doubt the former, just look at the definition of "success" from the Standish Group, who has published success (and failure) rates of software projects over a long period of time. Success, per the Standish Group is "the project is completed on time and on budget, with all the features and functions originally specified."[1] This is not a value-based definition but a constraint-based one. Using this definition, then, managers focus on following the plan with minimal changes. Colleague Rob Austin would classify this as a dysfunctional measurement (Austin 1996)—one that leads to the opposite behavior of what was intended.

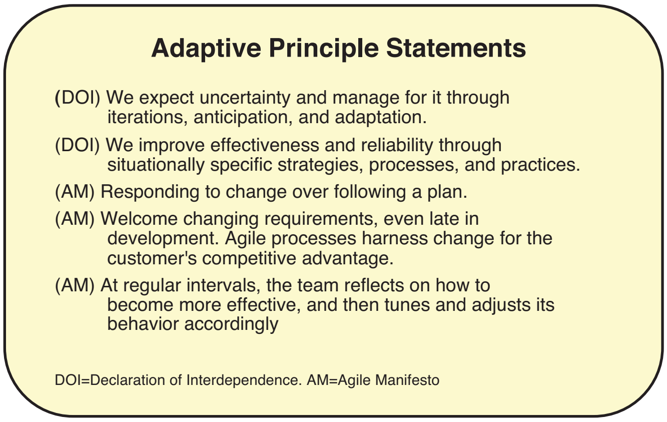
When customer value and quality are the goals, then a plan becomes a means to achieve those goals, not the goal itself. The constraints embedded in those plans are still important; they still guide the project; we still want to understand variations from the plans, but—and this is a big but—plans are not sacrosanct; they are meant to be flexible; they are meant to be guides, not straightjackets.

[1] Standish Group. Chaos Reports (http://www.standishgroup.com/chaos_resources/chronicles.php).

Both traditional and agile leaders plan, and both spend a fair amount of time planning. But they view plans in radically different ways. They both believe in plans as baselines, but traditional managers are constantly trying to "correct" actual results to that baseline. In the PMBOK[2], for example, the relevant activity is described as "corrective action" to guide the team back to the plan. In agile project management, we use "adaptive action" to describe which course of action to take (and one of those actions may be to correct to the plan).

The agile principles documents—the Agile Manifesto and the Declaration of Interdependence—contain five principle statements about adaptation, as shown in Figure 4-1.

Figure 4-1
Adaptive Principle Statements



Adaptive Principle Statements

(DOI) We expect uncertainty and manage for it through iterations, anticipation, and adaptation.

(DOI) We improve effectiveness and reliability through situationally specific strategies, processes, and practices.

(AM) Responding to change over following a plan.

(AM) Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

(AM) At regular intervals, the team reflects on how to become more effective, and then tunes and adjusts its behavior accordingly

DOI=Declaration of Interdependence. AM=Agile Manifesto

These principles could be summarized as follows:

- We expect change (uncertainty) and respond accordingly rather than follow outdated plans.
- We adapt our processes and practices as necessary.

The ability to respond to change drives competitive advantage. Think of the possibilities (not the problems) of being able to release a new version of a product weekly. Think of the competitive advantage of being able to pack-

[2] The Project Management Institute's *Project Management Body of Knowledge*, known as the PMBOK.

age features so customers feel they have software specifically customized for them (and the cost to maintain the software remains low).

Teams must adapt, but they can't lose track of the ultimate goals of the project. Teams should constantly evaluate progress, whether adapting or anticipating, by asking these four questions:

- Is value, in the form of a releasable product, being delivered?
- Is the quality goal of building a reliable, adaptable product being met?
- Is the project progressing satisfactorily within acceptable constraints?
- Is the team adapting effectively to changes imposed by management, customers, or technology?

The dictionary defines *change* as: "To cause to be different, to give a completely different form or appearance to." It defines *adapt as*: "To make suitable to or fit for a specific use or situation." Changing and adapting are not the same and the difference between them is important. There is no goal inherent in change—as the quip says, "stuff happens." Adaptation, on the other hand, is directed towards a goal (suitability). Change is mindless; adaptation is mindful.

Adaptation can be considered a mindful response to change.

The Science of Adaptation

Former Visa International CEO Dee Hock (1999) coined the word "chaordic" to describe both the world around us and his approach to managing a far-flung enterprise—balanced on the precipice between chaos and order. Our sense of the world dictates management style. If the world is perceived as static, then production-style management practices will dominate. If the world is perceived as dynamic, however, then exploration-style management practices will come to the fore. Of course, it's not that simple—there is always a blend of static and dynamic, which means that managers must always perform a delicate balancing act.

In the last two decades a vanguard of scientists and managers have articulated a profound shift in their view about how organisms and organizations evolve, respond to change, and manage their growth. Scientists' findings about the tipping points of chemical reactions and the "swarm" behavior of ants have given organizational researchers insights into what makes successful companies and successful managers. Practitioners have studied how innovative groups work most effectively.

As quantum physics changed our notions of predictability and Darwin changed our perspective on evolution, complex adaptive systems (CAS) theory has reshaped scientific and management thinking. In an era of rapid change, we need better ways of making sense of the world around us. Just as biologists now study ecosystems as well as species, executives and managers need to understand the global economic and political ecosystems in which their companies compete.

"A complex adaptive system, be it biologic or economic below, is an ensemble of independent agents

- *Who interact to create an ecosystem*
- *Whose interaction is defined by the exchange of information*
- *Whose individual actions are based on some system of internal rules*
- *Who self-organize in nonlinear ways to produce emergent results*
- *Who exhibit characteristics of both order and chaos*
- *Who evolve over time" (Highsmith 2000).*

For an agile project, the *ensemble* includes core team members, customers, suppliers, executives, and other participants who interact with each other in various ways. It is these interactions, and the tacit and explicit information exchanges that occur within them, that project management practices need to facilitate.

The individual agent's actions are driven by a set of internal rules—the core ideology and values of APM, for example. Both scientific and management researchers have shown that a simple set of rules can generate complex behaviors and outcomes, whether in ant colonies or project teams. Complex rules, on the other hand, often become bureaucratic. How these rules are formulated has a significant impact on how the complex system operates.

Newtonian approaches predict results. CAS approaches create emergent results. "Emergence is a property of complex adaptive systems that creates some greater property of the whole (system behavior) from the interaction of the parts (self-organizing agent behavior). Emergent results cannot be predicted in the normal sense of cause and effect relationships, but they can be anticipated by creating patterns that have previously produced similar results" (Highsmith 2000). Creativity and innovation are the emergent results of well functioning agile teams.

An adaptive development process has a different character from an optimizing one. Optimizing reflects a basic prescriptive Plan-Design-Build lifecycle. Adapting reflects an organic, evolutionary Envision-Explore-Adapt lifecycle. An adaptive approach begins not with a single solution, but with multiple potential solutions (experiments). It explores and selects the best by applying a series of fitness tests (actual product features or simulations subjected to acceptance tests) and then adapting to feedback. When uncertainty is low, adaptive approaches run the risk of higher costs. When uncertainty is high, optimizing approaches run the risk of settling too early on a particular solution and stifling innovation. The salient point is that these two fundamental approaches to development are very different, and they require different processes, different management approaches, and different measurements of success.

Newtonian versus quantum, predictability versus flexibility, optimization versus adaptation, efficiency versus innovation—all these dichotomies reflect a fundamentally different way of making sense about the world and how to manage effectively within it. Because of high iteration costs, the traditional perspective was predictive and change averse, and deterministic processes arose to support this traditional viewpoint. But our viewpoint needs to change. Executives, project leaders, and development teams must embrace a different view of the new product development world, one that not only recognizes change in the business world, but also understands the power of driving down iteration costs to enable experimentation and emergent processes. Understanding these differences and how they affect product development is key to understanding APM.

Exploring

Agility is the ability to both create and respond to change in order to profit in a turbulent business environment (from Chapter 1). The ability to respond to change is good. The ability to create change for competitors is even better. When you create change you are on the competitive offensive. When you respond to competitors' changes you are on the defensive. When you can respond to change at any point in the development lifecycle, even late, then you have a distinct advantage.

Adaptation needs to exceed the rate of market changes.

But change is hard. Although agile values tell us that responding to change is more important than following a plan, and that embracing rather than resisting change leads to better products, working in a high-change environment can be nerve-wracking for team members. Exploration is difficult; it raises anxiety, trepidation, and sometimes even a little fear. Agile project leaders need to encourage and inspire team members to work through the difficulties of a high-change environment. Remaining calm themselves, encouraging experimentation, learning through both successes and mistakes, and helping team members understand the vision are all part of this encouragement. Good leaders create a safe environment in which people can voice outlandish ideas, some of which turn out not to be so outlandish after all. External encouragement and inspiration help teams build internal motivation.

Great explorations flow from inspirational leaders. Cook, Magellan, Shackleton, and Columbus were inspirational leaders with vision. They persevered in the face of monumental obstacles, not the least of which was fear of the unknown. Magellan, after years of dealing with the entrenched Spanish bureaucracy trying to scuttle his plans, launched his five-ship fleet on October 3, 1519. On September 6, 1522, the Victoria, last of the ships, sailed into port without Magellan, who had died in the Philippines after completing the most treacherous part of the journey. The expedition established a route around Cape Horn and sailed across the vast Pacific Ocean for the first time (Joyner 1992).

Great explorers articulate goals that inspire people—goals that get people excited such that they inspire themselves. These goals or visions serve as a unifying focal point of effort, galvanizing people and creating an esprit de corps among the team. Inspirational goals need to be energizing, compelling, clear, and feasible, but just barely. Inspirational goals tap into a team's passion.

Encouraging leaders also know the difference between good goals and bad ones. We all know of egocentric managers who point to some mountain and say, "Let's get up there, team," when everyone else is thinking, "Who is he kidding? There's not a snowball's chance in the hot place that we can carry that off." "Bad BHAGs [Big Hairy Audacious Goals], it turns out, are set with *bravado*; good BHAGs are set with understanding," says Jim Collins (2001). Inspirational leaders know that setting a vision for the product is a team effort, one based on analysis, understanding, and realistic risk assessment, combined with a sprinkle of adventure.

Innovative product development teams are led, not managed. They allow their leaders to be inspirational. They internalize the leader's encouragement. Great new products, outstanding enhancements to existing products, and creative new business initiatives are driven by passion and inspiration. Project managers who focus on network diagrams, cost budgets, and resource histograms are dooming their teams to mediocrity.[3, 4]

Leaders help articulate the goals; teams internalize them and motivate themselves. This internal motivation enables exploration. We don't arrive at something new, better, and different without trial and error, launching off in multiple new directions to find the one that seems promising. Magellan and his ships spent 38 days covering the 334 miles of the straits that bear his name. In the vast expanse of islands and peninsulas, they explored many dead ends before finding the correct passages (Kelley 2001).

Magellan's ship Victoria sailed nearly 1,000 miles, back and forth—up estuaries that dead-ended and back out—time and time again. Magellan (his crew, actually) was the first to circumnavigate the globe. But Magellan

[3] This sentence should not be interpreted as saying these things are unimportant, because properly used, each can be useful to the project leader. It is when they become the focal point that trouble ensues.

[4] Ken Delcol observes, "Most PMs are not selected for their ability to inspire people! Leadership and business influencing skills are hard to establish in an interview. Most managers have a difficult time identifying and evaluating people with these skill sets."

would probably have driven a production-style project manager or executive a little crazy, because he surely didn't follow a plan. But then, any detailed plan would have been foolish—no one even knew whether ships could get around Cape Horn; none had found the way when Magellan launched. No one knew how large the Pacific Ocean was, and even the best guestimates turned out to be thousands of miles short. His vision never changed, but his "execution" changed every day based on new information.

Teams need a shared purpose and goal, but they also need encouragement to adapt—to experiment, explore, make mistakes, regroup, and forge ahead again.

Responding to Change

We expect change (uncertainty) and respond accordingly rather than follow outdated plans.

This statement reflects the agile viewpoint characterized further by

- Envision-Explore versus Plan-Do
- Adapting versus anticipating

In *Artful Making*, Harvard Business School professor and colleague Rob Austin and his coauthor Lee Devin (2003) discuss a $125 million IT project disaster in which the company refused to improvise and change from the detailed plan set down prior to the project's start. " 'Plan the work and work the plan' was their implicit mantra," they write. "And it led them directly to a costly and destructive course of action.... We'd all like to believe that this kind of problem is rare in business. It's not."

Every project has knowns and unknowns, certainties and uncertainties, and therefore every project has to balance planning and adapting. Balancing is required because projects also run the gamut from production-style ones in which uncertainty is low, to exploration-style ones in which uncertainty is high. Exploration-style projects, similar to development of the Sketchbook Pro© product introduced in Chapter 1, require a process that emphasizes envisioning and then exploring into that vision rather than detailed planning and relatively strict execution of tasks. It's not that one is right and the other wrong, but that each style is more or less applicable to a particular project type.

Another factor that impacts project management style is the cost of an iteration; that is, the cost of experimenting. Even if the need for innovation is great, high iteration costs may dictate a process with greater anticipatory work. Low-cost iterations, like those mentioned earlier, enable an adaptive style of development in which plans, architectures, and designs evolve concurrently with the actual product.

Product, Process, People

Faced with major redirection in release 2.0 of their product, the Sketchbook Pro© team introduced in Chapter 1 delivered the revised product in 42 days. As I quip in workshops, "I know teams who would have complained for 42 days with comments such as: "They don't know what they want. They are always changing their minds." Adaptability has three components—product, process, and people. You need to have a gung-ho agile team with the right attitude about change. You need processes and practices that allow the team to adapt to circumstances. And you *need* high quality code with automated tests. You can have pristine code and a non-agile team and change will be difficult. All three are required to have an agile, adaptable environment.

The barrier to agility in many software organizations is their failure to deal with the technical debt in legacy code. The failure is understandable because the solution can be costly and time consuming. However, failure to address this significant barrier keeps many organizations from realizing their agile potential. It took years for legacy code to degenerate; it will take significant time to revitalize the code. It requires a systematic investment in refactoring and automated testing—over several release cycles—to begin to solve the problems of years of neglect.

Barriers or Opportunities

One of the constant excuses, complaints, or rationalizations about some agile practices are “They would take too much time,” or “They would cost too much.” This has been said about short iterations, frequent database updates, continuous integration, automated testing, and a host of other agile practices. All too often companies succumb to what colleague Israel Gat calls the “new toy” syndrome—placing all their emphasis on new development and ignoring legacy code. Things like messy old code then become excuses and barriers to change. Some activities certainly are cost-prohibitive, but many of these are artificial barriers that people voice. Experienced agilists turn these barriers into opportunities. They ask, “What would be the benefit if we *could* do this?”

Working with a large company—and a very large program team (multiple projects, one integrated product suite) of something over 500 people—several years ago we wanted them to do a complete multi-project code integration at the end of every couple of iterations. The reply was, “We can’t do that, it would take multiple people and several weeks of time out of development.” This was from a group who had experienced severe problems in prior releases when they integrated products very late in the release cycle. Our response was, “What would the benefit be if you could do the integration quickly and at low cost?” and, “You don’t have a choice; if you want to be agile you must integrate across the entire product suite early and often.” Grumbling, they managed the first integration with significant effort, but with far less time than they anticipated. By the time 3–4 integrations had occurred, they had figured out how to do them in a few days with limited personnel. The benefits from frequent integration were significant, eliminating many problems that previously would have lingered, unfound, until close to release date.

Most, but not all, of the time perceived barriers to change (it costs too much) really point out inefficiencies—opportunities to streamline the process and enhance the organization’s ability to adapt. Agile development demands short-cycle iterations. Doing short-cycle iterations demands finding ways to do repetitive things quickly and inexpensively. Doing things quickly and inexpensively enables teams to respond to changes in ways they never anticipated previously. Doing things quickly and inexpensively fosters innovation because it encourages teams to experiment. These innovations ripple out into other parts of the organization. Lowering the cost of change enables companies to rethink their business models.

Reliable, Not Repeatable

Note that the word “repeatable” isn’t in the agile lexicon. Implementing repeatable processes has been the goal of many companies, but in product development, repeatability turns out to be the wrong goal; in fact, it turns out to be an extremely counterproductive goal. *Repeatable* means doing the same thing in the same way to product the same results. *Reliable* means meeting targets regardless of the impediments thrown in your way—it means constantly adapting to meet a goal.

Repeatable processes reduce variability through measurement and constant process correction. The term originated in manufacturing, where results were well defined and repeatability meant that if a process had consistent inputs, then defined outputs would be produced. Repeatable means that the conversion of inputs to outputs can be replicated with little variation. It implies that no new information can be generated during the process because we have to know all the information in advance to predict the output results accurately. Repeatable processes are not effective for product development projects because precise results are rarely predictable, inputs vary considerably from project to project, and the input-to-output conversions themselves are highly variable.

Reliable processes focus on outputs, not inputs. Using a reliable process, team members figure out ways to consistently achieve a given goal even though the inputs vary dramatically. Because of the input variations, the team may not use the same processes or practices from one project, or even one iteration, to the next. Reliability is results driven. Repeatability is input driven. The irony is that if every project process was somehow made repeatable, the project would be extremely unstable because of input and transformation variations. Even those organizations that purport to have repeatable processes are often successful not because of those processes, but because of the adaptability of the people who are using those processes.

At best, a repeatable process can deliver only what was specified in the beginning. A reliable, emergent process can actually deliver a better result than anyone ever conceived in the beginning. An emergent process can produce what you wish you had thought about at the start if only you had been smart and prescient enough.

Herein lies a definitional issue with project scope. With production-style projects, those amenable to repeatable processes, scope is considered to be the defined requirements. But in product development, requirements evolve and change over the life of the project, so "scope" can never be precisely defined in the beginning. Therefore, the correct scope to consider for agile projects isn't defined requirements but the articulated product vision—a releasable product. Product managers may be worried about specific requirements, but executives are concerned about the product as a whole—Does it meet the vision of the customer? When management asks the ever-popular question, "Did the project meet its scope, schedule, and cost targets?" The answer should be evaluated according to the vision, value, and totality of the capabilities delivered. That is, the evaluation of success can be encapsulated in the question "Do we have a releasable product?" not on whether the set of specific features defined at the start of the project was produced.

Agile Project Management is both reliable and predictable: It can deliver products that meet the customer's evolving needs within the boundary constraints better than any other process for a given level of uncertainty. Why does this happen? Not because some project manager specified detailed tasks and micromanaged them, but because an agile leader established an environment in which people wanted to excel and meet their goals.

Although APM is reliable, it is not infallible, and it cannot eliminate the vagaries of uncertainty, nor the surprises encountered while exploring. APM can shift the odds toward success. If executives expect projects to deliver on the product vision, within the constraints of specified time and cost, every time, without fail, then they should be running an assembly line, not developing products.

Reflection and Retrospective

Adapting requires both a certain mindset and a set of skills. If we are to be adaptable, then we must be willing to seriously and critically evaluate our performance as individual contributors and as teams. Effective teams cover four key subject areas in their retrospectives: *product* from both the customer's perspective and a technical quality perspective; *process*, as in how well the processes and practices being used by the team are working; *team*, as in how well the group is working as a high-performance team; and *project*, as in how the project is progressing according to plan. Feedback in each of these areas—at the end of each iteration and at the end of the project—leads to adaptations that improve performance. The "how to" of retrospectives and reflection is covered in Chapter 10, "The Adapt and Close Phases."

Principles to Practices

We adapt our processes and practices as necessary.

Ultimately, what people *do*, how they *behave*, is what creates great products. Principles and practices are guides; they help identify and reinforce certain behaviors.

Although principles guide agile teams, specific practices are necessary to actually accomplish work. A process structure and specific practices form a minimal, flexible framework for self-organizing teams. In an agile project there must be both anticipatory and adaptive processes and practices. Release planning uses known information to "anticipate" the future. Refactoring uses information found later in the project to "adapt" code. Ron Jeffries once said, "I have more confidence in my ability to adapt than in my ability to plan." Agilists do anticipate, but they always try to understand the limits of anticipation and try to err on the side of less of it.

Final Thoughts

Developing great products requires exploration, not tracking against a plan. Exploring and adapting are two behavioral traits required to innovate—having the courage to explore into the unknown and having the humility to recognize mistakes and adapt to the situation. Magellan had a vision, a goal, and some general ideas about sailing from Spain down the coast of South America, avoiding Portuguese ships if at all possible, exploring dead end after dead end to finding a way around Cape Horn, then tracking across the Pacific to once-again known territory in the Southeast Asia archipelagoes. Great new products come from similarly audacious goals and rough plans that often include large gaps in which "miracles happen," much like the miracle of finding the Straits of Magellan.

Index