

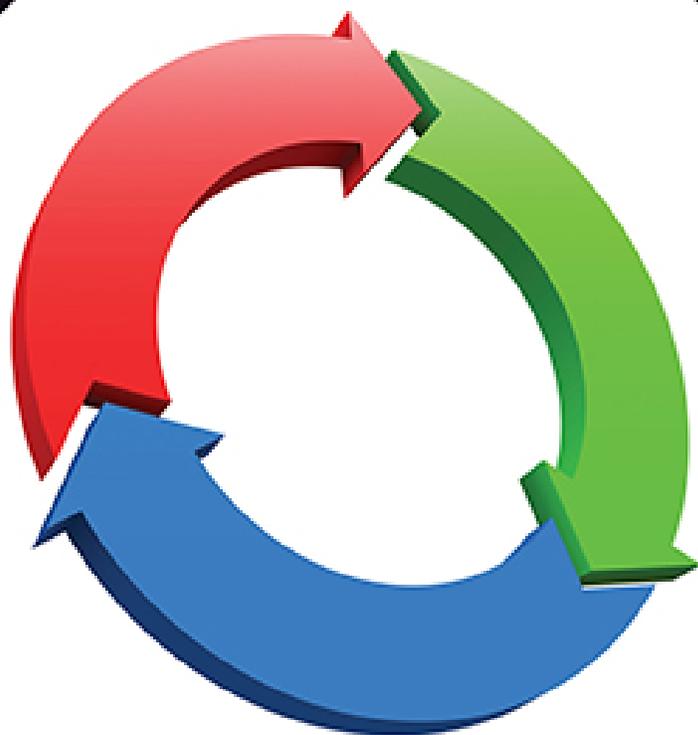
LEARNING MADE EASY



3rd Edition

# Agile Project Management

for  
**dummies**<sup>®</sup>  
A Wiley Brand



Make performance  
be progress

Deliver customer value  
in weeks, not months

Turn agile principles  
into practice

**Mark C. Layton**

CST, PMP, MBA<sup>2</sup>

**Steven J Ostermiller**

CST, PMP, ICP-ACC

**Dean J. Kynaston**

CSP-SM, CSP-PO, MBA



# Agile Project Management

3rd Edition

by Mark C. Layton, Steven J Ostermiller,  
and Dean J. Kynaston

for  
**dummies**  
A Wiley Brand

## **Agile Project Management For Dummies® , 3rd Edition**

Published by: **John Wiley & Sons, Inc.**, 111 River Street, Hoboken, NJ 07030-5774, [www.wiley.com](http://www.wiley.com)

Copyright © 2020 by John Wiley & Sons, Inc., Hoboken, New Jersey

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

**Trademarks:** Wiley, For Dummies, the Dummies Man logo, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and may not be used without written permission. SAFe and Scaled Agile Framework are registered trademarks of Scaled Agile, Inc. Certified Scrum Developer, Certified Scrum Product Owner, Certified Scrum Professional, Certified Scrum Trainer, and Certified ScrumMaster are registered trademarks of Scrum Alliance. PMI Agile Certified Practitioner and PMI-ACP are registered trademarks of Project Management Institute, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY:  
THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-

572-3993, or fax 317-572-4002. For technical support, please visit <https://hub.wiley.com/community/support/dummies>.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit [www.wiley.com](http://www.wiley.com).

Library of Congress Control Number: 2020942690

ISBN 978-1-119-67699-7 (pbk); ISBN 978-1-119-67706-2 (ebk); ISBN 978-1-119-67705-5 (ebk)

# Agile Project Management For Dummies®

To view this book's Cheat Sheet, simply go to [www.dummies.com](http://www.dummies.com) and search for “Agile Project Management For Dummies Cheat Sheet” in the Search box.

## Table of Contents

### Cover

### Introduction

[About This Book](#)

[Foolish Assumptions](#)

[Icons Used in This Book](#)

[Beyond the Book](#)

[Where to Go from Here](#)

### Part 1: Understanding Agility

#### Chapter 1: Modernizing Project Management

[Project Management Needed a Makeover](#)

[Introducing Agile Project Management](#)

[Agile Project Management Is Becoming Agile Product Management](#)

#### Chapter 2: Applying the Agile Manifesto and Principles

[Understanding the Agile Manifesto](#)

[Outlining the Four Values of the Agile Manifesto](#)

[Defining the 12 Agile Principles](#)

[Adding the Platinum Principles](#)

[Changes as a Result of Agile Values](#)

[The Agile Litmus Test](#)

## **Chapter 3: Why Being Agile Works Better**

[Evaluating Agile Benefits](#)

[How Agile Approaches Beat Historical Approaches](#)

[Why People Like Being Agile](#)

## **Chapter 4: Agility Is about Being Customer Focused**

[Knowing Your Customers](#)

[Figuring Out the Problem Your Customer Needs to Solve](#)

[Understanding Root Cause Analysis](#)

## **Part 2: Being Agile**

### **Chapter 5: Agile Approaches**

[Diving under the Umbrella of Agile Approaches](#)

[Reviewing the Big Three: Lean, Scrum, and Extreme Programming](#)

[Putting It All Together](#)

### **Chapter 6: Agile Environments in Action**

[Creating the Physical Environment](#)

[Low-Tech Communicating](#)

[High-Tech Communicating](#)

[Choosing Tools](#)

### **Chapter 7: Agile Behaviors in Action**

[Establishing Agile Roles](#)

[Establishing New Values](#)

[Changing Team Philosophy](#)

### **Chapter 8: The Permanent Team**

[Enabling Long-Lived Product Development Teams](#)

[Enabling Autonomy, Mastery, and Purpose](#)  
[Building Team Knowledge and Capability](#)

## **Part 3: Agile Planning and Execution**

### **Chapter 9: Defining the Product Vision and Product Roadmap**

[Agile Planning](#)  
[Defining the Product Vision](#)  
[Creating a Product Roadmap](#)  
[Completing the Product Backlog](#)

### **Chapter 10: Planning Releases and Sprints**

[Refining Requirements and Estimates](#)  
[Release Planning](#)  
[Preparing for Release](#)  
[Sprint Planning](#)

### **Chapter 11: Working throughout the Day**

[Planning Your Day: The Daily Scrum](#)  
[Tracking Progress](#)  
[Agile Roles in the Sprint](#)  
[Creating Shippable Functionality](#)  
[Information Radiators](#)  
[The End of the Day](#)

### **Chapter 12: Showcasing Work, Inspecting, and Adapting**

[The Sprint Review](#)  
[The Sprint Retrospective](#)

## **Part 4: Agility Management**

### **Chapter 13: Managing a Portfolio: Pursuing Value over Requirements**

[Understanding the Differences in Agile Portfolio Management](#)  
[Managing Agile Product Portfolios](#)

## **Chapter 14: Managing Scope and Procurement**

[What's Different about Agile Scope Management?](#)

[Managing Agile Scope](#)

[What's Different about Agile Procurement?](#)

[Managing Agile Procurement](#)

## **Chapter 15: Managing Time and Cost**

[What's Different about Agile Time Management?](#)

[Managing Agile Schedules](#)

[What's Different about Agile Cost Management?](#)

[Managing Agile Budgets](#)

## **Chapter 16: Managing Team Dynamics and Communication**

[What's Different about Agile Team Dynamics?](#)

[Managing Team Dynamics](#)

[What's Different about Agile Communication?](#)

[Managing Agile Communication](#)

## **Chapter 17: Managing Quality and Risk**

[What's Different about Agile Quality?](#)

[Managing Agile Quality](#)

[What's Different about Agile Risk Management?](#)

[Managing Agile Risk](#)

## **Part 5: Ensuring Success**

### **Chapter 18: Building a Foundation**

[Organizational and Individual Commitment](#)

[Choosing the Right Pilot Team Members](#)

[Creating an Environment That Enables Agility](#)

[Support Agility Initially and Over Time](#)

### **Chapter 19: De-Scaling across Teams**

[Multi-Team Agile Development](#)

[Making Work Digestible through Vertical Slicing](#)

[Multi-Team Coordination with LeSS](#)

[Aligning through Roles with Scrum@Scale](#)

[Joint Program Planning with SAFe](#)

[Disciplined Agile Toolkit](#)

## **Chapter 20: Being a Change Agent**

[Becoming Agile Requires Change](#)

[Why Change Doesn't Happen on Its Own](#)

[Strategic Approaches to Implementing and Managing Change](#)

[Platinum Edge's Change Roadmap](#)

[Leading by Example](#)

[Avoiding Transformation Pitfalls](#)

[Signs Your Changes Are Slipping](#)

## **Part 6: The Part of Tens**

### **Chapter 21: Ten Key Benefits of Agile Product Development**

[Higher Customer Satisfaction](#)

[Better Product Quality](#)

[Reduced Risk](#)

[Increased Collaboration and Ownership](#)

[More Relevant Metrics](#)

[Improved Performance Visibility](#)

[Increased Investment Control](#)

[Improved Predictability](#)

[Optimized Team Structures](#)

[Higher Team Morale](#)

### **Chapter 22: Ten Key Factors for Agile Product Development Success**

[Dedicated Team Members](#)

[Collocation](#)

[Done Means Shippable](#)

[Address What Scrum Exposes](#)

[Clear Product Vision and Roadmap](#)

[Product Owner Empowerment](#)

[Developer Versatility](#)  
[Scrum Master Clout](#)  
[Leadership Support for Learning](#)  
[Transition Support](#)

## **Chapter 23: Ten Signs That You're Not Agile**

[A Non-Shippable Sprint Product Increment](#)  
[Long Release Cycles](#)  
[Disengaged Stakeholders](#)  
[Lack of Customer Contact](#)  
[Lack of Skill Versatility](#)  
[Automatable Processes Remain Manual](#)  
[Prioritizing Tools over the Work](#)  
[High Manager-to-Creator Ratio](#)  
[Working around What Scrum Exposes](#)  
[Practicing Faux Agile](#)

## **Chapter 24: Ten Valuable Resources for Agile Professionals**

[Agile Project Management For Dummies Online Cheat Sheet](#)  
[Scrum For Dummies](#)  
[The Scrum Alliance](#)  
[The Agile Alliance](#)  
[International Consortium for Agile \(ICAgile\)](#)  
[Mind the Product and ProductTank](#)  
[Lean Enterprise Institute](#)  
[Extreme Programming](#)  
[The Project Management Institute Agile Community](#)  
[Platinum Edge](#)

**[Index](#)**

**[About the Authors](#)**

**[Advertisement Page](#)**

**[Connect with Dummies](#)**

## End User License Agreement

# List of Tables

## **Chapter 2**

[TABLE 2-1 Individuals and Interactions versus Processes and Tools](#)

[TABLE 2-2 Identifying Useful Documentation](#)

[TABLE 2-3 Customer Dissatisfaction and How Agile Might Help](#)

[TABLE 2-4 Contrasting Historical Project Management with Agile Product Managemen...](#)

## **Chapter 4**

[TABLE 4-1 Customer Interview Do's and Don'ts](#)

## **Chapter 5**

[TABLE 5-1 Key Practices of Extreme Programming](#)

[TABLE 5-2 Similarities between Lean, Scrum, and Extreme Programming](#)

## **Chapter 6**

[TABLE 6-1 Common Distractions](#)

## **Chapter 7**

[TABLE 7-1 Characteristics of a Good Product Owner](#)

[TABLE 7-2 Characteristics of a Good Development Team Member](#)

[TABLE 7-3 Characteristics of a Good Scrum Master](#)

## **Chapter 10**

[TABLE 10-1 Decomposing a Requirement](#)

## **Chapter 11**

[TABLE 11-1 Common Roadblocks and Solutions](#)

## **Chapter 13**

[TABLE 13-1 Keys for Effective Agile Portfolio Management](#)

## **Chapter 14**

[TABLE 14-1 Traditional versus Agile Scope Management](#)

[TABLE 14-2 Agile Artifacts and Scope Management Roles](#)

[TABLE 14-3 Traditional versus Agile Procurement Management](#)

## **Chapter 15**

[TABLE 15-1 Traditional versus Agile Time Management](#)

[TABLE 15-2 Agile Artifacts and Time Management](#)

[TABLE 15-3 Traditional versus Agile Cost Management](#)

[TABLE 15-4 Sample Scrum Team Budget for a Two-Week Sprint](#)

[TABLE 15-5 Income from a Traditional Project with a Final Release after Six Mont...](#)

[TABLE 15-6 Income with Monthly Releases and a Final Release after Six Months](#)

## **Chapter 16**

[TABLE 16-1 Traditional versus Agile Team Dynamics](#)

[TABLE 16-2 Product Management and Self-Managing Teams](#)

[TABLE 16-3 Success of Collocated and Dislocated Scrum Teams](#)

[TABLE 16-4 Traditional versus Agile Communication](#)

[TABLE 16-5 Agile Communication Channels](#)

## **Chapter 17**

[TABLE 17-1 Traditional versus Agile Quality](#)

[TABLE 17-2 Traditional versus Agile Risk](#)

[TABLE 17-3 Income from a Traditional Project with a Final Release after Six Mont...](#)

[TABLE 17-4 Income from Agile Development with Monthly Releases and a Final Relea...](#)

[TABLE 17-5 Cost of Failure on a Waterfall Project](#)

[TABLE 17-6 Cost of Failure with Agile Techniques](#)

[TABLE 17-7 Agile Risk Management Tools](#)

## **Chapter 20**

[TABLE 20-1 Common Agile Transition Problems and Solutions](#)

# **List of Illustrations**

## **Chapter 1**

[FIGURE 1-1: Actual use of requested software features.](#)

[FIGURE 1-2: Agile project management timeline.](#)

[FIGURE 1-3: Waterfall versus agile project.](#)

## **Chapter 2**

[FIGURE 2-1: Traditional project opportunity for change.](#)

[FIGURE 2-2: Charts and graphs for providing transparency.](#)

## **Chapter 3**

[FIGURE 3-1: A comparison of historical project management and agile concepts.](#)

[FIGURE 3-2: The waterfall project cycle is a linear methodology.](#)

[FIGURE 3-3: Agile approaches have an iterative development cycle.](#)

[FIGURE 3-4: Stability in flexibility with agile product development.](#)

[FIGURE 3-5: A risk and investment chart comparing waterfall and agile methodolo...](#)

## **Chapter 4**

[FIGURE 4-1: The sweet spot where a product is valuable, feasible, and usable.](#)

[FIGURE 4-2: The product canvas. © Shardul Mehta 2015<http://streetSMARTproductma...>](#)

[FIGURE 4-3: A customer journey map.](#)

[FIGURE 4-4: A customer empathy map.](#)

[FIGURE 4-5: The job-to-be-done timeline.](#)

[FIGURE 4-6: The scientific method.](#)

[FIGURE 4-7: A user story map.](#)

[FIGURE 4-8: The Ishikawa diagram.](#)

## **Chapter 5**

[FIGURE 5-1: Early hardware and software.](#)

[FIGURE 5-2: The origins of waterfall.](#)

[FIGURE 5-3: Iteration in waterfall.](#)

[FIGURE 5-4: The scrum approach.](#)

[FIGURE 5-5: Sprints are recurring processes.](#)

## **Chapter 6**

[FIGURE 6-1: Better communication through collocation.](#)

[FIGURE 6-2: A scrum task board on a wall or whiteboard.](#)

[FIGURE 6-3: Virtual classroom setting.](#)

[FIGURE 6-4: Virtual collaboration board.](#)

## **Chapter 7**

[FIGURE 7-1: The product team, scrum team, and development team.](#)

[FIGURE 7-2: Product owner communication cycle.](#)

[FIGURE 7-3: Development team member skill development.](#)

[FIGURE 7-4: Team communication complexity is a function of team size.](#)

## **Chapter 8**

[FIGURE 8-1: Highly aligned and autonomous team quadrants.](#)

## **Chapter 9**

[FIGURE 9-1: Traditional planning versus scrum planning.](#)

[FIGURE 9-2: Stages of agile planning and execution with the Roadmap to Value.](#)

[FIGURE 9-3: The product vision statement as part of the Roadmap to Value.](#)

[FIGURE 9-4: Expansion of Moore's template for a vision statement.](#)

[FIGURE 9-5: The product roadmap as part of the Roadmap to Value.](#)

[FIGURE 9-6: Features grouped by themes.](#)

[FIGURE 9-7: Product roadmap with ordered requirements.](#)

[FIGURE 9-8: Product backlog items sample.](#)

## **Chapter 10**

[FIGURE 10-1: Card-based user story example.](#)

[FIGURE 10-2: Sample user stories.](#)

[FIGURE 10-3: User story decomposition guidelines.](#)

[FIGURE 10-4: A deck of estimation poker cards.](#)

[FIGURE 10-5: Story sizes as T-shirt sizes and their Fibonacci numbers.](#)

[FIGURE 10-6: Release planning as part of the Roadmap to Value.](#)

[FIGURE 10-7: Sample release plan.](#)

[FIGURE 10-8: Operational support scrum team model.](#)

[FIGURE 10-9: Sprint planning as part of the Roadmap to Value.](#)

[FIGURE 10-10: Sprint backlog example.](#)

[FIGURE 10-11: Ratio of sprint planning meeting to sprint length.](#)

## **Chapter 11**

[FIGURE 11-1: The sprint and the daily scrum in the Roadmap to Value.](#)

[FIGURE 11-2: Sample sprint backlog.](#)

[FIGURE 11-3: A burndown chart.](#)

[FIGURE 11-4: Profiles of burndown charts.](#)

[FIGURE 11-5: Sample task board.](#)

[FIGURE 11-6: User story verification.](#)

## **Chapter 12**

[FIGURE 12-1: The sprint review in the Roadmap to Value.](#)

[FIGURE 12-2: Agile project feedback loops.](#)

[FIGURE 12-3: Ratio of sprint review meeting to sprint length.](#)

[FIGURE 12-4: The sprint retrospective in the Roadmap to Value.](#)

[FIGURE 12-5: Ratio of sprint retrospective meeting to sprint length.](#)

## **Chapter 13**

[FIGURE 13-1: Risk versus value quadrant.](#)

[FIGURE 13-2: Prioritized portfolio backlog of investment opportunities.](#)

[FIGURE 13-3: Financial cost of delay due to thrashed parallel product developme...](#)

[FIGURE 13-4: The law of diminishing returns.](#)

## **Chapter 14**

[FIGURE 14-1: The Roadmap to Value.](#)

[FIGURE 14-2: Adding a new requirement to the product backlog.](#)

## **Chapter 16**

[FIGURE 16-1: Email versus face-to-face conversation.](#)

[FIGURE 16-2: Comparison of communication types.](#)

[FIGURE 16-3: Profiles of burndown charts.](#)

## **Chapter 17**

[FIGURE 17-1: Quality feedback cycles.](#)

[FIGURE 17-2: Testing within sprints.](#)

[FIGURE 17-3: A user story and acceptance criteria.](#)

[FIGURE 17-4: Standish Group's "2015 Chaos Report."](#)

[FIGURE 17-5: Agile product development's declining risk model.](#)

[FIGURE 17-6: Sample definition of done.](#)

## **Chapter 18**

[FIGURE 18-1: Alignment of the agile transition team and the pilot scrum team ca...](#)

[FIGURE 18-2: The Roadmap to Value.](#)

## **Chapter 19**

[FIGURE 19-1: Vertical slices of product features implemented by multiple scrum ...](#)

[FIGURE 19-2: Scrum of scrums for coordinating between scrum teams.](#)

[FIGURE 19-3: Scrum of scrums task board.](#)

[FIGURE 19-4: Basic LeSS framework.](#)

[FIGURE 19-5: The LeSS Huge framework.](#)

[FIGURE 19-6: Scrum@Scale scrum of scrums model.](#)

[FIGURE 19-7: Scrum@Scale scrum of scrums of scrums model.](#)

[FIGURE 19-8: Scrum@Scale executive action team \(EAT\).](#)

[FIGURE 19-9: Scrum@Scale product owner team.](#)

[FIGURE 19-10: Scrum@Scale executive metascrum \(EMS\).](#)

[FIGURE 19-11: SAFe for Lean Enterprises 5.0.](#)

[FIGURE 19-12: Essential SAFe configuration.](#)

## **Chapter 20**

[FIGURE 20-1: Lewin's unfreeze, change, refreeze change philosophy.](#)

[FIGURE 20-2: Platinum Edge agile transition roadmap.](#)

[FIGURE 20-3: Product development efforts that can benefit from agile techniques...](#)

[FIGURE 20-4: Satir's Curve.](#)

# Introduction

---

Welcome to *Agile Project Management For Dummies*, 3rd Edition. Agile project management has grown to be as common as any management technique for product development — and not only software product development. For nearly two decades, we have trained and coached companies big and small, all over the world, about how to become more nimble, adaptive, and responsive in both the development of their products and their organizations — in other words, how to become more agile. Through this work, we found there was a need to write a digestible guide that anyone, regardless of experience, could understand.

## ***About This Book***

*Agile Project Management For Dummies*, 3rd Edition is more than just an introduction to agile practices and approaches; you also discover the steps to become more agile in mindset and behavior. The material here goes beyond theory and is meant to be a field guide for all experience levels, giving you the tools and information you need to be successful with agile techniques in the trenches of product development.

## ***Foolish Assumptions***

This book was written as a reference guide for anyone wanting to learn more about business agility. If you strive to be more agile in responding to customer needs and problems — whether or not you're an organizational leader, a project manager, a member of a product team,

an agile enthusiast, or a product stakeholder — this book will help you on your journey.

Regardless of your experience or level of familiarity, this book provides insights you may find helpful. We hope it brings clarity to any confusion or myths regarding agile product development you may have encountered.

## *Icons Used in This Book*

Throughout this book, you'll find the following icons.



**TIP**

Tips are points to help you along your agile product development journey. Tips can save you time and help you quickly understand a particular topic, so when you see them, take a look!



**REMEMBER**

The Remember icon is a reminder of something you may have seen in past chapters. It also may be a reminder of a commonsense principle that is easily forgotten. These icons can help jog your memory when an important term or concept appears.



**WARNING**

The Warning icon indicates that you want to watch out for a certain action or behavior. Read these to steer clear of big problems!



TECHNICAL  
STUFF

The Technical Stuff icon indicates information that is interesting but not essential to the text. If you see a Technical Stuff icon, you don't need to read it to understand agile product development, but the information there might just pique your interest.



ON THE  
WEB

On the Web means that you can find more information on the book's website at [www.dummies.com/go/agileprojectmanagementfd3e](http://www.dummies.com/go/agileprojectmanagementfd3e).

## ***Beyond the Book***

Although this book broadly covers the agile project management spectrum, we can cover only so much in a set number of pages! If you find yourself at the end of this book thinking, “This was an amazing book! Where can I learn more about how to advance my products under an agile approach?” check out [Chapter 24](#) or head over to [www.dummies.com](http://www.dummies.com) for more resources.

We've provided a cheat sheet for tips on assessing your current product development efforts in relation to agile principles as well as free tools for managing projects using agile techniques. To get to the cheat sheet, go to [www.dummies.com](http://www.dummies.com), and then type *Agile Project Management For Dummies Cheat Sheet* in the Search box. This is also where you'll find any significant updates or changes that occur between editions of this book.

## ***Where to Go from Here***

We wrote this book so that you could read it in just about any order. Depending on your role, you may want to pay extra attention to the appropriate sections of the book. For example:

- » If you're just starting to learn about product development and agile approaches, start with [Chapter 1](#) and read the book straight through to the end.
- » If you're a member of a product team and want to know the basics of agile product development, check out the information in [Part 3 \(Chapters 9 through 12\)](#).
- » If you're a project manager transitioning to agile approaches to product development, you may be interested to learn how agile techniques improve the management of time, cost, scope, procurement, quality, and risk. Review [Part 4 \(Chapters 13 through 17\)](#).
- » If you know the basics of agile product development and are looking at bringing agile practices to your company or expanding your agile footprint across your organization, [Part 5 \(Chapters 18 through 20\)](#) will provide you with helpful information.

**Part 1**  
**Understanding Agility**

## IN THIS PART ...

Understand why project management has modernized due to the flaws and weaknesses in historical approaches to project management.

Find out why agile methods are becoming more product-focused than project-focused, and become acquainted with the foundation of agile product development: the Agile Manifesto and the 12 Agile Principles.

Discover the advantages that your products, projects, teams, customers, and organization can gain from adopting agile techniques.

Understand the importance of placing the customer's needs first and why agile techniques help to make the customer central to every decision, functionality, and problem.

# Chapter 1

## Modernizing Project Management

---

### IN THIS CHAPTER

- » Understanding why project management needs to change
  - » Seeing how agile project management is becoming agile product management
  - » Finding out about agile product development
- 

*Agile* is a descriptor of a mindset approach to project management that focuses on early delivery of business value, continuous improvement of the product being created and the processes used to create the product, scope flexibility, team input, and delivering well-tested products that reflect customer needs.

In this chapter, you find out why agile processes emerged as an approach to software development project management in the mid-1990s and why agile methodologies have caught the attention of project managers, customers who invest in the development of new products and services, and executives whose companies fund product development. While business agility is popular in software product development, agile values, principles, and techniques apply in a multitude of industries and applications — not just software. This chapter also explains the advantages of agile approaches over long-standing project management methodologies.

# ***Project Management Needed a Makeover***

A *project* is a planned program of work that requires a definitive amount of time, effort, and planning to complete. Projects have goals and objectives and often must be completed in some fixed period of time and within a certain budget.

Because you're reading this book, you're likely a project manager or someone who initiates projects, works on projects, or is affected by projects in some way.

Agile approaches are a response to the need to modernize project management. To understand how agile approaches are revolutionizing product development, it helps to know a little about the history and purpose of project management and the issues that projects face today.

## ***The origins of modern project management***

Projects have been around since ancient times. From the Great Wall of China to the Mayan pyramids at Tikal, from the invention of the printing press to the invention of the Internet, people have accomplished endeavors big and small in projects.

As a formal discipline, project management as we know it has been around only since the middle of the twentieth century. Around the time of World War II, researchers around the world were making major advances in building and programming computers, mostly for the United States military. To complete those projects, they started creating formal project management processes. The first processes were based on step-by-step

manufacturing models the United States military used during World War II.

People in the computing field adopted these step-based manufacturing processes because early computer-related projects relied heavily on hardware, with computers that filled up entire rooms. Software, by contrast, was a smaller part of computer projects. In the 1940s and 1950s, computers might have thousands of physical vacuum tubes but fewer than 30 lines of programming code. The 1940s manufacturing process used on these initial computers is the foundation of the project management methodology known as waterfall.

In 1970, a computer scientist named Winston Royce wrote “Managing the Development of Large Software Systems,” an article for the IEEE that described the phases in the waterfall methodology. The term *waterfall* was coined later, but the phases, even if they are sometimes titled differently, are essentially the same as originally defined by Royce:

1. Requirements
2. Design
3. Development
4. Integration
5. Testing
6. Deployment

On waterfall projects, you move to the next phase only when the prior one is complete — hence the name waterfall.



TECHNICAL  
STUFF

Pure waterfall project management — completing each step in full before moving to the next step — is actually a misinterpretation of Royce's suggestions. Royce identified that this approach was inherently risky and recommended developing and testing within iterations to create products — suggestions that were overlooked by many organizations that adopted the waterfall methodology.

The waterfall methodology was the most common project management approach in software development until it was surpassed by improved approaches based on agile techniques around 2008.

### ***The problem with the status quo***

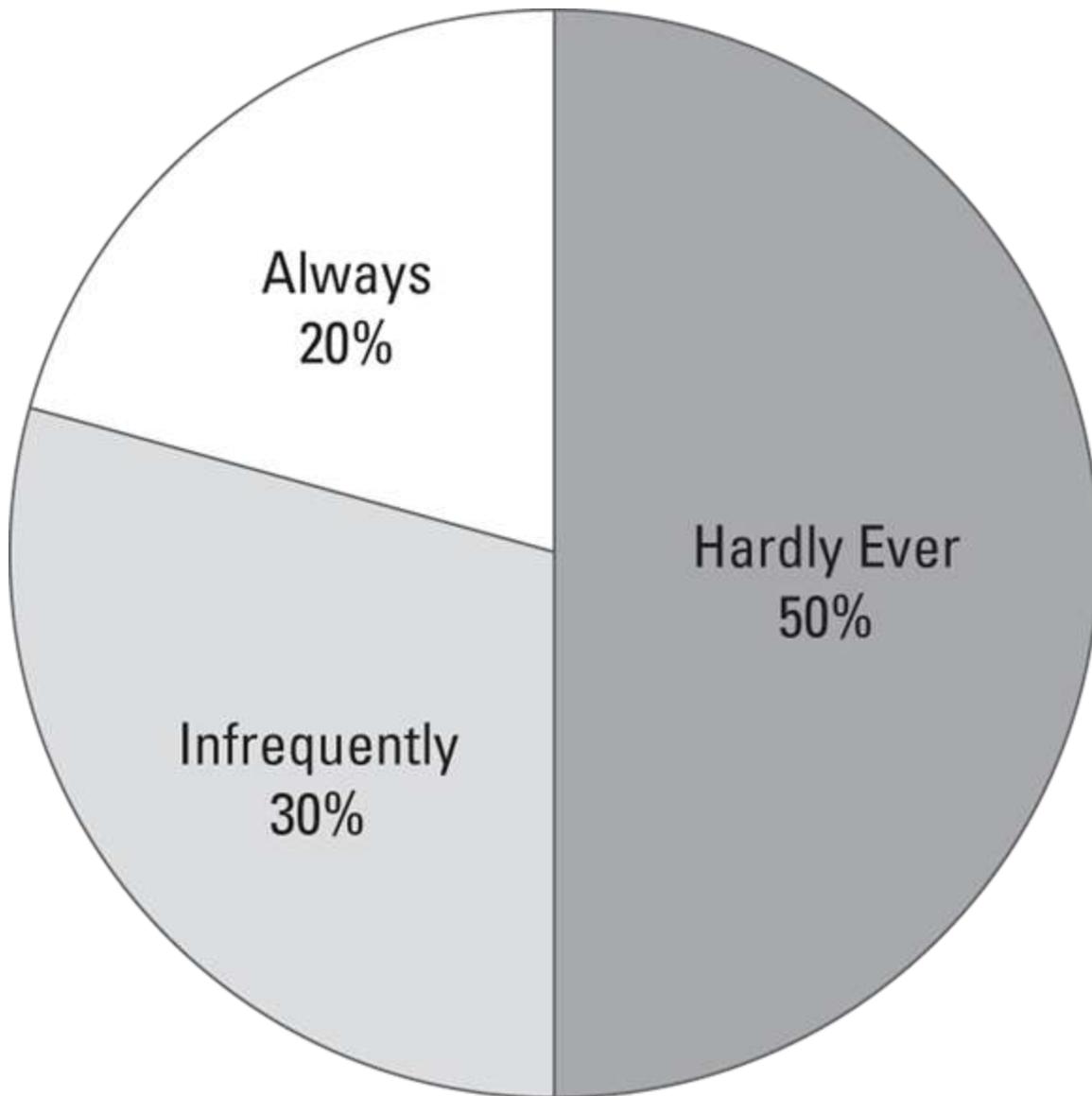
Computer technology has, of course, changed a great deal since the last century. Many people have a computer on their wrist with more power, memory, and capabilities than the largest, most expensive machine that existed when people first started using waterfall methodologies.

At the same time, the people using computers have changed as well. Instead of creating behemoth machines with minimal programs for a few researchers and the military, people create hardware and software for the general public. In many countries, almost everyone uses a tablet or smartphone, directly or indirectly, every day. Software runs our cars, our appliances, our homes; it provides our daily information and daily entertainment. Even young children use computers — 2-year-olds are almost more adept with the iPhone than their parents. The demand for newer, better products is constant.

Somehow, during all this growth of technology, processes were not left behind. Software developers are still using project management methodologies from the 1950s, and all these approaches were derived from manufacturing processes meant for the hardware-heavy computers of the mid-twentieth century.

Today, traditional projects that do succeed often suffer from one problem: *scope bloat*, the introduction of unnecessary product features. Think about the software products you use every day. For example, the word-processing program we're typing on right now has many features and tools. Even though we write with this program every day, we use only some of the features all the time. We use other elements less frequently. And we have never used quite a few tools — and come to think of it, we don't know anyone else who has used them, either. The features that few people use are the result of scope bloat.

Scope bloat appears in all kinds of software, from complex enterprise applications to websites that everyone uses. [Figure 1-1](#) shows data from a Standish Group study that illustrates just how common scope bloat is. In the figure, you can see that 80 percent of requested features are infrequently or never used.



© Copyright 2017 Standish Group

**FIGURE 1-1:** Actual use of requested software features.

The numbers in [Figure 1-1](#) illustrate an enormous waste of time and money. That waste is a direct result of traditional project management processes that are unable to accommodate change. Project managers and stakeholders know that change is not welcome mid-project, so their best chance of getting a potentially desirable feature is at the start of a project. Therefore, they ask for

- » Everything they need
- » Everything they think they may need
- » Everything they want
- » Everything they think they may want

The result is the bloat in features that results in the statistics in [Figure 1-1](#).

## SOFTWARE PROJECT SUCCESS AND FAILURE

Stagnation in traditional project management approaches is catching up with the software industry. In 2015, a software statistical company called the Standish Group did a study on the success and failure rates of 10,000 projects in the US. The results of the study showed that

- *29 percent of traditional projects failed outright.* The projects were cancelled before they finished and did not result in any product releases. These projects delivered no value whatsoever.
- *60 percent of traditional projects were challenged.* The projects were completed but had gaps between expected and actual cost, time, quality, or a combination of these elements. The average difference between the expected and actual project results — looking at time, cost, and features not delivered — was well over 100 percent.
- *11 percent of projects succeeded.* The projects were completed and delivered the expected product in the originally expected time and budget.

Of the hundreds of billions of dollars spent on product development in the US alone, billions of dollars were wasted on projects that never deployed a single piece of functionality.

The problems associated with using outdated management and development approaches are not trivial. These problems waste billions of dollars a year. The billions of dollars lost in project failure in 2015 (see the sidebar, "[Software project success and failure](#)") could equate to millions of jobs around the world.