

Praise for *Agile Testing*

“As Agile methods have entered the mainstream, we’ve learned a lot about how the testing discipline fits into Agile projects. Lisa and Janet give us a solid look at what to do, and what to avoid, in Agile testing.”

—Ron Jeffries, www.XProgramming.com

“An excellent introduction to agile and how it affects the software test community!”

—Gerard Meszaros, Agile Practice Lead and Chief Test Strategist at Solution Frameworks, Inc., an agile coaching and lean software development consultancy

“In sports and music, people know the importance of practicing technique until it becomes a part of the way they do things. This book is about some of the most fundamental techniques in software development—how to build quality into code—techniques that should become second nature to every development team. The book provides both broad and in-depth coverage of how to move testing to the front of the development process, along with a liberal sprinkling of real-life examples that bring the book to life.”

—Mary Poppendieck, Author of *Lean Software Development* and *Implementing Lean Software Development*

“Refreshingly pragmatic. Chock-full of wisdom. Absent of dogma. This book is a game-changer. Every software professional should read it.”

—Uncle Bob Martin, Object Mentor, Inc.

“With *Agile Testing*, Lisa and Janet have used their holistic sensibility of testing to describe a culture shift for testers and teams willing to elevate their test effectiveness. The combination of real-life project experiences and specific techniques provide an excellent way to learn and adapt to continually changing project needs.”

—Adam Geras, M.Sc. Developer-Tester, Ideaca Knowledge Services

“On Agile projects, everyone seems to ask, ‘But, what about testing?’ Is it the development team’s responsibility entirely, the testing team, or a collaborative effort between developers and testers? Or, ‘How much testing should we automate?’ Lisa and Janet have written a book that finally answers these types of questions and more! Whether you’re a tester, developer, or manager, you’ll learn many great examples and stories from the real-world work experiences they’ve shared in this excellent book.”

—Paul Duvall, CTO of Stelligent and co-author of *Continuous Integration: Improving Software Quality and Reducing Risk*

“Finally a book for testers on Agile teams that acknowledges there is not just one right way! *Agile Testing* provides comprehensive coverage of the issues testers face when they move to Agile: from tools and metrics to roles and process. Illustrated with numerous stories and examples from many contributors, it gives a clear picture of what successful Agile testers are doing today.”

—Bret Pettichord, Chief Technical Officer of WatirCraft and Lead Developer of Watir

This page intentionally left blank

AGILE TESTING

This page intentionally left blank

AGILE TESTING

A PRACTICAL GUIDE FOR TESTERS AND AGILE TEAMS

Lisa Crispin
Janet Gregory

◆◆ Addison-Wesley

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

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States, please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data:

Crispin, Lisa.

Agile testing : a practical guide for testers and agile teams /
Lisa Crispin, Janet Gregory. — 1st ed.

p. cm.

Includes bibliographical references and index.

ISBN-13: 978-0-321-53446-0 (pbk. : alk. paper)

ISBN-10: 0-321-53446-8 (pbk. : alk. paper) 1. Computer software—
Testing. 2. Agile software development. I. Gregory, Janet. II. Title.

QA76.76.T48C75 2009
005.1—dc22

2008042444

Copyright © 2009 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.
Rights and Contracts Department
501 Boylston Street, Suite 900
Boston, MA 02116
Fax (617) 671-3447

ISBN-13: 978-0-321-53446-0

ISBN-10: 0-321-53446-8

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

First printing, December 2008

To my husband, Bob Downing—you're the bee's knees!

—Lisa

To Jack, Dana, and Susan, and to all the writers in my family.

—Janet

And to all our favorite donkeys and dragons.

—Lisa and Janet

This page intentionally left blank

CONTENTS

Foreword by Mike Cohn	xxiii
Foreword by Brian Marick	xxv
Preface	xxvii
Acknowledgments	xxxvii
About the Authors	xli

Part I	Introduction	1
---------------	---------------------	----------

Chapter 1	What Is Agile Testing, Anyway?	3
	Agile Values	3
	What Do We Mean by “Agile Testing”?	4
	A Little Context for Roles and Activities on an Agile Team	7
	Customer Team	7
	Developer Team	7
	Interaction between Customer and Developer Teams	8
	How Is Agile Testing Different?	9
	Working on Traditional Teams	9
	Working on Agile Teams	10
	Traditional vs. Agile Testing	12
	Whole-Team Approach	15
	Summary	17
Chapter 2	Ten Principles for Agile Testers	19
	What’s an Agile Tester?	19
	The Agile Testing Mind-Set	20

Applying Agile Principles and Values	21
Provide Continuous Feedback	22
Deliver Value to the Customer	22
Enable Face-to-Face Communication	23
Have Courage	25
Keep It Simple	26
Practice Continuous Improvement	27
Respond to Change	28
Self-Organize	29
Focus on People	30
Enjoy	31
Adding Value	31
Summary	33

Part II Organizational Challenges 35

Chapter 3 Cultural Challenges	37
Organizational Culture	37
Quality Philosophy	38
Sustainable Pace	40
Customer Relationships	41
Organization Size	42
Empower Your Team	44
Barriers to Successful Agile Adoption by Test/QA Teams	44
Loss of Identity	44
Additional Roles	45
Lack of Training	45
Not Understanding Agile Concepts	45
Past Experience/Attitude	48
Cultural Differences among Roles	48
Introducing Change	49
Talk about Fears	49
Give Team Ownership	50
Celebrate Success	50
Management Expectations	52
Cultural Changes for Managers	52
Speaking the Manager's Language	55
Change Doesn't Come Easy	56
Be Patient	56
Let Them Feel Pain	56

Build Your Credibility	57
Work On Your Own Professional Development	57
Beware the Quality Police Mentality	57
Vote with Your Feet	57
Summary	58
Chapter 4 Team Logistics	59
Team Structure	59
Independent QA Teams	60
Integration of Testers into an Agile Project	61
Agile Project Teams	64
Physical Logistics	65
Resources	66
Tester-Developer Ratio	66
Hiring an Agile Tester	67
Building a Team	69
Self-Organizing Team	69
Involving Other Teams	69
Every Team Member Has Equal Value	70
Performance and Rewards	70
What Can You Do?	71
Summary	71
Chapter 5 Transitioning Typical Processes	73
Seeking Lightweight Processes	73
Metrics	74
Lean Measurements	74
Why We Need Metrics	75
What Not to Do with Metrics	77
Communicating Metrics	77
Metrics ROI	78
Defect Tracking	79
Why Should We Use a Defect Tracking System (DTS)?	80
Why Shouldn't We Use a DTS?	82
Defect Tracking Tools	83
Keep Your Focus	85
Test Planning	86
Test Strategy vs. Test Planning	86
Traceability	88

Existing Processes and Models	88
Audits	89
Frameworks, Models, and Standards	90
Summary	93
Part III The Agile Testing Quadrants	95
Chapter 6 The Purpose of Testing	97
The Agile Testing Quadrants	97
Tests that Support the Team	98
Tests that Critique the Product	101
Knowing When a Story Is Done	104
Shared Responsibility	105
Managing Technical Debt	106
Testing in Context	106
Summary	108
Chapter 7 Technology-Facing Tests that Support the Team	109
An Agile Testing Foundation	109
The Purpose of Quadrant 1 Tests	110
Supporting Infrastructure	111
Why Write and Execute These Tests?	112
Lets Us Go Faster and Do More	112
Making Testers' Jobs Easier	114
Designing with Testing in Mind	115
Timely Feedback	118
Where Do Technology-Facing Tests Stop?	119
What If the Team Doesn't Do These Tests?	121
What Can Testers Do?	121
What Can Managers Do?	122
It's a Team Problem	123
Toolkit	123
Source Code Control	123
IDEs	124
Build Tools	126
Build Automation Tools	126
Unit Test Tools	126
Summary	127

Chapter 8	Business-Facing Tests that Support the Team	129
	Driving Development with Business-Facing Tests	129
	The Requirements Quandary	132
	Common Language	134
	Eliciting Requirements	135
	Advance Clarity	140
	Conditions of Satisfaction	142
	Ripple Effects	143
	Thin Slices, Small Chunks	144
	How Do We Know We're Done?	146
	Tests Mitigate Risk	147
	Testability and Automation	149
	Summary	150
Chapter 9	Toolkit for Business-Facing Tests that Support the Team	153
	Business-Facing Test Tool Strategy	153
	Tools to Elicit Examples and Requirements	155
	Checklists	156
	Mind Maps	156
	Spreadsheets	159
	Mock-Ups	160
	Flow Diagrams	160
	Software-Based Tools	163
	Tools for Automating Tests Based on Examples	164
	Tools to Test below the GUI and API Level	165
	Tools for Testing through the GUI	170
	Strategies for Writing Tests	177
	Build Tests Incrementally	178
	Keep the Tests Passing	179
	Use Appropriate Test Design Patterns	179
	Keyword and Data-Driven Tests	182
	Testability	183
	Code Design and Test Design	184
	Automated vs. Manual Quadrant 2 Tests	185
	Test Management	186
	Summary	186
Chapter 10	Business-Facing Tests that Critique the Product	189
	Introduction to Quadrant 3	190
	Demonstrations	191

Scenario Testing	192
Exploratory Testing	195
Session-Based Testing	200
Automation and Exploratory Testing	201
An Exploratory Tester	201
Usability Testing	202
User Needs and Persona Testing	202
Navigation	204
Check Out the Competition	204
Behind the GUI	204
API Testing	205
Web Services	207
Testing Documents and Documentation	207
User Documentation	207
Reports	208
Tools to Assist with Exploratory Testing	210
Test Setup	211
Test Data Generation	212
Monitoring Tools	212
Simulators	213
Emulators	213
Summary	214
Chapter 11 Critiquing the Product Using Technology-Facing Tests	217
Introduction to Quadrant 4	217
Who Does It?	220
When Do You Do It?	222
“ility” Testing	223
Security	223
Maintainability	227
Interoperability	228
Compatibility	229
Reliability	230
Installability	231
“ility” Summary	232
Performance, Load, Stress, and Scalability Testing	233
Scalability	233
Performance and Load Testing	234
Performance and Load-Testing Tools	234
Baseline	235

Test Environments	237
Memory Management	237
Summary	238
Chapter 12 Summary of Testing Quadrants	241
Review of the Testing Quadrants	241
A System Test Example	242
The Application	242
The Team and the Process	243
Tests Driving Development	244
Unit Tests	244
Acceptance Tests	245
Automation	245
The Automated Functional Test Structure	245
Web Services	247
Embedded Testing	248
Critiquing the Product with Business-Facing Tests	248
Exploratory Testing	248
Testing Data Feeds	249
The End-to-End Tests	249
User Acceptance Testing	250
Reliability	250
Documentation	251
Documenting the Test Code	251
Reporting the Test Results	251
Using the Agile Testing Quadrants	252
Summary	253

Part IV Automation 255

Chapter 13 Why We Want to Automate Tests and What Holds Us Back	257
Why Automate?	258
Manual Testing Takes Too Long	258
Manual Processes Are Error Prone	259
Automation Frees People to Do Their Best Work	259
Automated Regression Tests Provide a Safety Net	261
Automated Tests Give Feedback, Early and Often	262
Tests and Examples that Drive Coding Can Do More	262

Tests Are Great Documentation	263
ROI and Payback	264
Barriers to Automation—Things that Get in the Way	264
Bret’s List	264
Our List	265
Programmers’ Attitude—“Why Automate?”	265
The “Hump of Pain” (The Learning Curve)	266
Initial Investment	267
Code that’s Always in Flux	269
Legacy Code	269
Fear	269
Old Habits	270
Can We Overcome These Barriers?	270
Summary	271
Chapter 14 An Agile Test Automation Strategy	273
An Agile Approach to Test Automation	274
Automation Test Categories	274
Test Automation Pyramid	276
What Can We Automate?	279
Continuous Integration, Builds, and Deploys	280
Unit and Component Tests	282
API or Web Services Testing	282
Testing behind the GUI	282
Testing the GUI	282
Load Tests	283
Comparisons	283
Repetitive Tasks	284
Data Creation or Setup	284
What Shouldn’t We Automate?	285
Usability Testing	285
Exploratory Testing	286
Tests that Will Never Fail	286
One-Off Tests	286
What Might Be Hard to Automate?	287
Developing an Automation Strategy—Where Do We Start?	288
Where Does It Hurt the Most?	289
Multi-Layered Approach	290
Think about Test Design and Maintenance	292
Choosing the Right Tools	294

Applying Agile Principles to Test Automation	298
Keep It Simple	298
Iterative Feedback	299
Whole-Team Approach	300
Taking the Time to Do It Right	301
Learn by Doing	303
Apply Agile Coding Practices to Tests	303
Supplying Data for Tests	304
Data Generation Tools	304
Avoid Database Access	306
When Database Access Is Unavoidable or Even Desirable	307
Understand Your Needs	310
Evaluating Automation Tools	311
Identifying Requirements for Your Automation Tool	311
One Tool at a Time	312
Choosing Tools	313
Agile-Friendly Tools	316
Implementing Automation	316
Managing Automated Tests	319
Organizing Tests	319
Organizing Test Results	322
Go Get Started	324
Summary	324

Part V An Iteration in the Life of a Tester 327

Chapter 15 Tester Activities in Release or Theme Planning	329
The Purpose of Release Planning	330
Sizing	332
How to Size Stories	332
The Tester's Role in Sizing Stories	333
An Example of Sizing Stories	334
Prioritizing	338
Why We Prioritize Stories	338
Testing Considerations While Prioritizing	339
What's in Scope?	340
Deadlines and Timelines	340
Focus on Value	341

System-Wide Impact	342
Third-Party Involvement	342
Test Planning	345
Where to Start	345
Why Write a Test Plan?	345
Types of Testing	346
Infrastructure	346
Test Environments	347
Test Data	348
Test Results	349
Test Plan Alternatives	350
Lightweight Test Plans	350
Using a Test Matrix	350
Test Spreadsheet	353
A Whiteboard	353
Automated Test List	354
Preparing for Visibility	354
Tracking Test Tasks and Status	354
Communicating Test Results	357
Release Metrics	358
Summary	366
Chapter 16 Hit the Ground Running	369
Be Proactive	369
Benefits	370
Do You Really Need This?	372
Potential Downsides to Advance Preparation	373
Advance Clarity	373
Customers Speak with One Voice	373
Story Size	375
Geographically Dispersed Teams	376
Examples	378
Test Strategies	380
Prioritize Defects	381
Resources	381
Summary	382
Chapter 17 Iteration Kickoff	383
Iteration Planning	383
Learning the Details	384
Considering All Viewpoints	385

Writing Task Cards	389
Deciding on Workload	393
Testable Stories	393
Collaborate with Customers	396
High-Level Tests and Examples	397
Reviewing with Customers	400
Reviewing with Programmers	400
Test Cases as Documentation	402
Summary	403
Chapter 18 Coding and Testing	405
Driving Development	406
Start Simple	406
Add Complexity	407
Assess Risk	407
Coding and Testing Progress Together	409
Identify Variations	410
Power of Three	411
Focus on One Story	411
Tests that Critique the Product	412
Collaborate with Programmers	413
Pair Testing	413
“Show Me”	413
Talk to Customers	414
Show Customers	414
Understand the Business	415
Completing Testing Tasks	415
Dealing with Bugs	416
Is It a Defect or Is It a Feature?	417
Technical Debt	418
Zero Bug Tolerance	418
It’s All about Choices	419
Decide Which Bugs to Log	420
Choose When to Fix Your Bugs	421
Choose the Media You Should Use to Log a Bug	423
Alternatives and Suggestions for Dealing with Bugs	424
Start Simple	428
Facilitate Communication	429
Testers Facilitate Communication	429
Distributed Teams	431

Regression Tests	432
Keep the Build “Green”	433
Keep the Build Quick	433
Building a Regression Suite	434
Checking the “Big Picture”	434
Resources	434
Iteration Metrics	435
Measuring Progress	435
Defect Metrics	437
Summary	440
Chapter 19 Wrap Up the Iteration	443
Iteration Demo	443
Retrospectives	444
Start, Stop, and Continue	445
Ideas for Improvements	447
Celebrate Successes	449
Summary	451
Chapter 20 Successful Delivery	453
What Makes a Product?	453
Planning Enough Time for Testing	455
The End Game	456
Testing the Release Candidate	458
Test on a Staging Environment	458
Final Nonfunctional Testing	458
Integration with External Applications	459
Data Conversion and Database Updates	459
Installation Testing	461
Communication	462
What If It’s Not Ready?	463
Customer Testing	464
UAT	464
Alpha/Beta Testing	466
Post-Development Testing Cycles	467
Deliverables	468
Releasing the Product	470
Release Acceptance Criteria	470
Release Management	474
Packaging	474

Customer Expectations	475
Production Support	475
Understand Impact to Business	475
Summary	476

Part VI Summary 479

Chapter 21 Key Success Factors	481
Success Factor 1: Use the Whole-Team Approach	482
Success Factor 2: Adopt an Agile Testing Mind-Set	482
Success Factor 3: Automate Regression Testing	484
Success Factor 4: Provide and Obtain Feedback	484
Success Factor 5: Build a Foundation of Core Practices	486
Continuous Integration	486
Test Environments	487
Manage Technical Debt	487
Working Incrementally	488
Coding and Testing Are Part of One Process	488
Synergy between Practices	489
Success Factor 6: Collaborate with Customers	489
Success Factor 7: Look at the Big Picture	490
Summary	491
Glossary	493
Bibliography	501
Index	509

This page intentionally left blank

FOREWORD

By **Mike Cohn**

“Quality is baked in,” the programmers kept telling me. As part of a proposed acquisition, my boss had asked me to perform some final due diligence on the development team and its product. We’d already established that the company’s recently launched product was doing well in the market, but I was to make sure we were not about to buy more trouble than benefit. So I spent my time with the development team. I was looking for problems that might arise from having rushed the product into release. I wondered, “Was the code clean? Were there modules that could only be worked on by one developer? Were there hundreds or thousands of defects waiting to be discovered?” And when I asked about the team’s approach to testing, “Quality is baked in” was the answer I got.

Because this rather unusual colloquialism could have meant just about anything, I pressed further. What I found was that this was the company founder’s shorthand for expressing one of quality pioneer W. Edwards Deming’s famous fourteen points: Build quality into the product rather than trying to test it in later.

The idea of building quality into their products is at the heart of how agile teams work. Agile teams work in short iterations in part to ensure that the application remains at a known state of quality. Agile teams are highly cross-functional, with programmers, testers, and others working side by side throughout each iteration so that quality can be baked into products through techniques such as acceptance-test driven development, a heavy emphasis on automated testing, and whole-team thinking. Good agile teams bake quality in by building their products continuously, integrating new work within minutes of its being completed. Agile teams utilize techniques such as refactoring and a preference for simplicity in order to prevent technical debt from accumulating.

Learning how to do these things is difficult, and especially so for testers, whose role has been given scant attention in previous books. Fortunately, the book you now hold in your hands answers questions on the mind of every tester who's beginning to work on an agile project, such as:

- What are my roles and responsibilities?
- How do I work more closely with programmers?
- How much do we automate, and how do we start automating?

The experience of Lisa and Janet shines through on every page of the book. However, this book is not just their story. Within this book, they incorporate dozens of stories from real-world agile testers. These stories form the heart of the book and are what makes it so unique. It's one thing to shout from the ivory tower, "Here's how to do agile testing." It's another to tell the stories of the teams that have struggled and then emerged agile and victorious over challenges such as usability testing, legacy code that resists automation, transitioning testers used to traditional phase-gate development, testing that "keeps up" with short iterations, and knowing when a feature is "done."

Lisa and Janet were there at the beginning, learning how to do agile testing back when the prevailing wisdom was that agile teams didn't need testers and that programmers could bake quality in by themselves. Over the years and through articles, conference presentations, and working with their clients and teams, Lisa and Janet have helped us see the rich role to be filled by testers on agile projects. In this book, Lisa and Janet use a test automation pyramid, the agile testing quadrants of Brian Marick (himself another world-class agile tester), and other techniques to show how much was missing from a mind-set that said testing is necessary but testers aren't.

If you want to learn how to bake quality into your products or are an aspiring agile tester seeking to understand your role, I can think of no better guides than Lisa and Janet.

FOREWORD

By Brian Marick

Imagine yourself skimming over a landscape thousands of years ago, looking at the people below. They're barely scraping out a living in a hostile territory, doing some hunting, some fishing, and a little planting. Off in the distance, you see the glitter of a glacier. Moving closer, you see that it's melting fast and that it's barely damming a huge lake. As you watch, the lake breaks through, sweeping down a riverbed, carving it deeper, splashing up against cliffs on the far side of the landscape—some of which collapse.

As you watch, the dazed inhabitants begin to explore the opening. On the other side, there's a lush landscape, teeming with bigger animals than they've ever seen before, some grazing on grass with huge seed heads, some squabbling over mounds of fallen fruit.

People move in. Almost immediately, they begin to live better. But as the years fly past, you see them adapt. They begin to use nets to fish in the fast-running streams. They learn the teamwork needed to bring down the larger animals, though not without a few deaths along the way. They find ever-better ways to cultivate this new grass they've come to call "wheat."

As you watch, the mad burst of innovation gives way to a stable solution, a good way to live in this new land, a way that's taught to each new generation. Although just over there, you spy someone inventing the wheel . . .

■ ■ ■

In the early years of this century, the adoption of Agile methods sometimes seemed like a vast dam breaking, opening up a way to a better—more productive, more joyful—way of developing software. Many early adopters saw benefits right away, even though they barely knew what they were doing.

Some had an easier time of it than others. Programmers were like the hunters in the fable above. Yes, they had to learn new skills in order to hunt bison, but they knew how to hunt rabbits, more or less, and there were plenty of rabbits around. Testers were more like spear-fishers in a land where spear-fishing wouldn't work. Going from spear-fishing to net-fishing is a much bigger conceptual jump than going from rabbit to bison. And, while some of the skills—cleaning fish, for example—were the same in the new land, the testers had to invent new skills of net-weaving before they could truly pull their weight.

So testing lagged behind. Fortunately, we had early adopters like Lisa and Janet, people who dove right in alongside the programmers, testers who were not jealous of their role or their independence, downright *pleasant* people who could figure out the biggest change of all in Agile testing: the tester's new social role.

As a result, we have this book. It's the stable solution, the good way for testers to live in this new Agile land of ours. It's not the final word—we *could* use the wheel, and I myself am eager for someone to invent antibiotics—but what's taught here will serve you well until someone, perhaps Lisa and Janet, brings the next big change.

PREFACE

We were early adopters of Extreme Programming (XP), testing on XP teams that weren't at all sure where testers or their brand of testing fit in. At the time, there wasn't much in the agile (which wasn't called agile yet) literature about acceptance testing, or how professional testers might contribute. We learned not only from our own experiences but from others in the small agile community. In 2002, Lisa co-wrote *Testing Extreme Programming* with Tip House, with lots of help from Janet. Since then, agile development has evolved, and the agile testing community has flourished. With so many people contributing ideas, we've learned a whole lot more about agile testing.

Individually and together, we've helped teams transition to agile, helped testers learn how to contribute on agile teams, and worked with others in the agile community to explore ways that agile teams can be more successful at testing. Our experiences differ. Lisa has spent most of her time as an agile tester on stable teams working for years at a time on web applications in the retail, telephony, and financial industries. Janet has worked with software organizations developing enterprise systems in a variety of industries. These agile projects have included developing a message-handling system, an environmental-tracking system, a remote data management system (including an embedded application, with a communication network as well as the application), an oil and gas production accounting application, and applications in the airline transportation industry. She has played different roles—sometimes tester, sometimes coach—but has always worked to better integrate the testers with the rest of the team. She has been with teams from as little as six months to as long as one-and-a-half years.

With these different points of view, we have learned to work together and complement each other's skill sets, and we have given many presentations and tutorials together.

WHY WE WROTE THIS BOOK

Several excellent books oriented toward agile development on testing and test patterns have been published (see our bibliography). These books are generally focused on helping the developer. We decided to write a book aimed at helping agile teams be more successful at delivering business value using tests that the business can understand. We want to help testers and quality assurance (QA) professionals who have worked in more traditional development methodologies make the transition to agile development.

We've figured out how to apply—on a practical, day-to-day level—the fruits of our own experience working with teams of all sizes and a variety of ideas from other agile practitioners. We've put all this together in this book to help testers, quality assurance managers, developers, development managers, product owners, and anyone else with a stake in effective testing on agile projects to deliver the software their customers need. However, we've focused on the role of the tester, a role that may be adopted by a variety of professionals.

Agile testing practices aren't limited to members of agile teams. They can be used to improve testing on projects using traditional development methodologies as well. This book is also intended to help testers working on projects using any type of development methodology.

Agile development isn't the only way to successfully deliver software. However, all of the successful teams we've been on, agile or waterfall, have had several critical commonalities. The programmers write and automate unit and integration tests that provide good code coverage. They are disciplined in the use of source code control and code integration. Skilled testers are involved from the start of the development cycle and are given time and resources to do an adequate job of all necessary forms of testing. An automated regression suite that covers the system functionality at a higher level is run and checked regularly. The development team understands the customers' jobs and their needs, and works closely together with the business experts.

People, not methodologies or tools, make projects successful. We enjoy agile development because its values, principles, and core practices enable people to do their best work, and testing and quality are central to agile development. In this book, we explain how to apply agile values and principles to your unique testing situation and enable your teams to succeed. We have more about that in Chapter 1, "What Is Agile Testing, Anyway?" and in Chapter 2, "Ten Principles for Agile Testers."

HOW WE WROTE THIS BOOK

Having experienced the benefits of agile development, we used agile practices to produce this book. As we began work on the book, we talked to agile testers and teams from around the globe to find out what problems they encountered and how they addressed them. We planned how we would cover these areas in the book.

We made a release plan based on two-week iterations. Every two weeks, we delivered two rough-draft chapters to our book website. Because we aren't co-located, we found tools to use to communicate, provide "source code control" for our chapters, deliver the product to our customers, and get their feedback. We couldn't "pair" much real-time, but we traded chapters back and forth for review and revision, and had informal "stand-ups" daily via instant message.

Our "customers" were the generous people in the agile community who volunteered to review draft chapters. They provided feedback by email or (if we were lucky) in person. We used the feedback to guide us as we continued writing and revising. After all the rough drafts were done, we made a new plan to complete the revisions, incorporating all the helpful ideas from our "customers."

Our most important tool was mind maps. We started out by creating a mind map of how we envisioned the whole book. We then created mind maps for each section of the book. Before writing each chapter, we brainstormed with a mind map. As we revised, we revisited the mind maps, which helped us think of ideas we may have missed.

Because we think the mind maps added so much value, we've included the mind map as part of the opening of each chapter. We hope they'll help you get an overview of all the information included in the chapter, and inspire you to try using mind maps yourself.

OUR AUDIENCE

This book will help you if you've ever asked any of the following excellent questions, which we've heard many times:

- If developers are writing tests, what do the testers do?
- I'm a QA manager, and our company is implementing agile development (Scrum, XP, DSDM, name your flavor). What's my role now?

- I've worked as a tester on a traditional waterfall team, and I'm really excited by what I've read about agile. What do I need to know to work on an agile team?
- What's an "agile tester"?
- I'm a developer on an agile team. We're writing code test-first, but our customers still aren't happy with what we deliver. What are we missing?
- I'm a developer on an agile team. We're writing our code test-first. We make sure we have tests for all our code. Why do we need testers?
- I coach an agile development team. Our QA team can't keep up with us, and testing always lags behind. Should we just plan to test an iteration behind development?
- I'm a software development manager. We recently transitioned to agile, but all our testers quit. Why?
- I'm a tester on a team that's going agile. I don't have any programming or automation skills. Is there any place for me on an agile team?
- How can testing possibly keep up with two-week iterations?
- What about load testing, performance testing, usability testing, all the other "ilities"? Where do these fit in?
- We have audit requirements. How does agile development and testing address these?

If you have similar questions and you're looking for practical advice about how testers contribute to agile teams and how agile teams can do an effective job of testing, you've picked up the right book.

There are many "flavors" of agile development, but they all have much in common. We support the Agile Manifesto, which we explain in Chapter 1, "What Is Agile Testing, Anyway?" Whether you're practicing Scrum, Extreme Programming, Crystal, DSDM, or your own variation of agile development, you'll find information here to help with your testing efforts.

A User Story for an Agile Testing Book

When Robin Dymond, a managing consultant and trainer who has helped many teams adopt lean and agile, heard we were writing this book, he sent us the user story he'd like to have fulfilled. It encapsulates many of the requirements we planned to deliver.

**Book Story 1**

As a QA professional, I can understand the main difference between traditional QA professionals and agile team members with a QA background, so that I can begin internalizing my new responsibilities and deliver value to the customer sooner and with less difficulty.

Acceptance conditions:

- My concerns and fears about losing control of testing are addressed.
- My concerns and fears about having to write code (never done it) are addressed.
- As a tester I understand my new value to the team.
- As a tester new to Agile, I can easily read about things that are most important to my new role.
- As a tester new to Agile, I can easily ignore things that are less important to my new role.
- As a tester new to Agile, I can easily get further detail about agile testing that is important to MY context.

Were I to suggest a solution to this problem, I think of Scrum versus XP. With Scrum you get a simple view that enables people to quickly adopt Agile. However, Scrum is the tip of the iceberg for successful agile teams. For testers who are new, I would love to see agile testing ideas expressed in layers of detail. What do I need to know today, what should I know tomorrow, and what context-sensitive things should I consider for continuous improvement?

We've tried to provide these layers of detail in this book. We'll approach agile testing from a few different perspectives: transitioning into agile development, using an agile testing matrix to guide testing efforts, and explaining all the different testing activities that take place throughout the agile development cycle.

HOW TO USE THIS BOOK

If you aren't sure where to start in this book, or you just want a quick overview, we suggest you read the last chapter, Chapter 22, "Key Success Factors," and follow wherever it leads you.

Part I: Introduction

If you want quick answers to questions such as "Is agile testing different than testing on waterfall projects?" or "What's the difference between a tester on a traditional team and an agile tester?," start with Part I, which includes the following chapters:

- Chapter 1: What Is Agile Testing, Anyway?
- Chapter 2: Ten Principles for Agile Testers

These chapters are the "tip of the iceberg" that Robin requested in his user story. They include an overview of how agile differs from a traditional phased approach and explore the "whole team" approach to quality and testing.

In this part of the book we define the "agile testing mind-set" and what makes testers successful on agile teams. We explain how testers apply agile values and principles to contribute their particular expertise.

Part II: Organizational Challenges

If you're a tester or manager on a traditional QA team, or you're coaching a team that's moving to agile, Part II will help you with the organizational challenges faced by teams in transition. The "whole team" attitude represents a lot of cultural changes to team members, but it helps overcome the fear testers have when they wonder how much control they'll have or whether they'll be expected to write code.

Some of the questions answered in Part II are:

- How can we engage the QA team?
- What about management's expectations?
- How should we structure our agile team, and where do the testers fit?
- What do we look for when hiring an agile tester?
- How do we cope with a team distributed across the globe?

Part II also introduces some topics we don't always enjoy talking about. We explore ideas about how to transition processes and models, such as audits or SOX compliance, that are common in traditional environments.

Metrics and how they're applied can be a controversial issue, but there are positive ways to use them to benefit the team. Defect tracking easily becomes a point of contention for teams, with questions such as "Do we use a defect-tracking system?" or "When do we log bugs?"

Two common questions about agile testing from people with traditional test team experience are "What about test plans?" and "Is it true there's no documentation on agile projects?" Part II clears up these mysteries.

The chapters in Part II are as follows:

- Chapter 3: Cultural Challenges
- Chapter 4: Team Logistics
- Chapter 5: Transitioning Typical Processes

Part III: The Agile Testing Quadrants

Do you want more details on what types of testing are done on agile projects? Are you wondering who does what testing? How do you know whether you've done all the testing that's needed? How do you decide what practices, techniques, and tools fit your particular situation? If these are your concerns, check out Part III.

We use Brian Marick's Agile Testing Quadrants to explain the purpose of testing. The quadrants help you define all the different areas your testing should address, from unit level tests to reliability and other "ilities," and everything in between. This is where we get down into the nitty-gritty of how to deliver a high-quality product. We explain techniques that can help you to communicate well with your customers and better understand their requirements. This part of the book shows how tests drive development at multiple levels. It also provides tools for your toolkit that can help you to effectively define, design, and execute tests that support the team and critique the product. The chapters include the following:

- Chapter 6: The Purpose of Testing
- Chapter 7: Technology-Facing Tests that Support the Team

- Chapter 8: Business-Facing Tests that Support the Team
- Chapter 9: Toolkit for Business-Facing Tests that Support the Team
- Chapter 10: Business-Facing Tests that Critique the Product
- Chapter 11: Critiquing the Product Using Technology-Facing Tests
- Chapter 12: Summary of Testing Quadrants

Part IV: Automation

Test automation is a central focus of successful agile teams, and it's a scary topic for lots of people (we know, because it's had us running scared before!). How do you squeeze test automation into short iterations and still get all the stories completed?

Part IV gets into the details of when and why to automate, how to overcome barriers to test automation, and how to develop and implement a test automation strategy that works for your team. Because test automation tools change and evolve so rapidly, our aim is not to explain how to use specific tools, but to help you select and use the right tools for your situation. Our agile test automation tips will help you with difficult challenges such as testing legacy code.

The chapters are as follows:

- Chapter 13: Why We Want to Automate Tests and What Holds Us Back
- Chapter 14: An Agile Test Automation Strategy

Part V: An Iteration in the Life of a Tester

If you just want to get a feel for what testers do throughout the agile development cycle, or you need help putting together all the information in this book, go to Part V. Here we chronicle an iteration, and more, in the life of an agile tester. Testers contribute enormous value throughout the agile software development cycles. In Part V, we explain the activities that testers do on a daily basis. We start with planning releases and iterations to get each iteration off to a good start, and move through the iteration—collaborating with the customer and development teams, testing, and writing code. We end the iteration by delivering new features and finding ways for the team to improve the process.

The chapters break down this way:

- Chapter 15: Tester Activities in Release or Theme Planning
- Chapter 16: Hit the Ground Running

- Chapter 17: Iteration Kickoff
- Chapter 18: Coding and Testing
- Chapter 19: Wrap Up the Iteration
- Chapter 20: Successful Delivery

Part VI: Summary

In Chapter 21, “Key Success Factors,” we present seven key factors agile teams can use for successful testing. If you’re having trouble deciding where to start with agile testing, or how to work on improving what you’re doing now, these success factors will give you some direction.

Other Elements

We’ve also included a glossary we hope you will find useful, as well as references to books, articles, websites, and blogs in the bibliography.

JUST START DOING IT—TODAY!

Agile development is all about doing your best work. Every team has unique challenges. We’ve tried to present all the information that we think may help agile testers, their teams, managers, and customers. Apply the techniques that you think are appropriate for your situation. Experiment constantly, evaluate the results, and come back to this book to see what might help you improve. Our goal is to help testers and agile teams enjoy delivering the best and most valuable product they can.

When we asked Dierk König, founder and project manager of Canoo Web-Test, what he thought was the number one success factor for agile testing, he answered: “Start doing it—today!” You can take a baby step to improve your team’s testing right now. Go get started!

This page intentionally left blank

ACKNOWLEDGMENTS

So many people have helped us with this book that it's hard to know whom to thank first. Chris Guzikowski gave us the opportunity to write this book and kept encouraging us along the way. When we were deciding whether to take on such a mammoth task, Mike Cohn gave us the sage advice that the best reason to write a book is that you have something to say. We sure have lots to say about agile testing. Fortunately, so do lots of other people who were willing to lend us a hand.

Many thanks to Brian Marick and Mike Cohn for writing such kind forewords. We're honored that Mike selected our book for his signature series. We're grateful for the many ideas and observations of his that are included in this book.

Brian Marick's "Agile Testing Matrix" has guided both of us in our agile projects for several years, and it provides the core of Part III. Thank you, Brian, for thinking up the quadrants (and so many other contributions to agile testing) and letting us use them here.

We made constant use of the agile value of feedback. Many thanks to our official reviewers: Jennitta Andrea, Gerard Meszaros, Ron Jeffries, and Paul Duvall. Each one had unique and insightful comments that helped us greatly improve the book. Gerard also helped us be more consistent and correct in our testing terminology, and contributed some agile testing success stories.

Special thanks to two reviewers and top-notch agile testers who read every word we wrote and spent hours discussing the draft chapters with us in person: Pierre Veragen and Paul Rogers. Many of the good ideas in this book are theirs.

We interviewed several teams to learn what advice they would give new agile teams and testers, and solicited success stories and “lessons learned” from colleagues in the agile testing community. Heartfelt thanks to our many contributors of sidebars and quotes, as well as providers of helpful feedback, including (in no particular order) Robin Dymond, Bret Pettichord, Tae Chang, Bob Galen, Erika Boyer, Grig Gheorghiu, Erik Bos, Mark Benander, Jonathan Rasmusson, Andy Pols, Dierk König, Rafael Santos, Jason Holzer, Christophe Louvion, David Reed, John Voris, Chris McMahan, Declan Whelan, Michael Bolton, Elisabeth Hendrickson, Joe Yakich, Andrew Glover, Alessandro Collino, Coni Tartaglia, Markus Gärtner, Megan Sumrell, Nathan Silberman, Mike Thomas, Mike Busse, Steve Perkins, Joseph King, Jakub Oleszkiewicz, Pierre Veragen (again), Paul Rogers (again), Jon Hagar, Antony Marcano, Patrick Wilson-Welsh, Patrick Fleisch, Apurva Chandra, Ken De Souza, and Carol Vaage.

Many thanks also to the rest of our community of unofficial reviewers who read chapters, gave feedback and ideas, and let us bounce ideas off of them, including Tom Poppendieck, Jun Bueno, Kevin Lawrence, Hannu Kokko, Titus Brown, Wim van de Goor, Lucas Campos, Kay Johansen, Adrian Howard, Henrik Kniberg, Shelly Park, Robert Small, Senaka Suriyaachchi, and Erik Petersen. And if we’ve neglected to list you here, it’s not that we value your contribution any less, it’s just that we didn’t keep good enough notes! We hope you will see how your time and effort paid off in the finished book.

We appreciate the groundwork laid by the agile pioneers who have helped us and our teams succeed with agile. You’ll find some of their works in the bibliography. We are grateful for the agile teams that have given us so many open source test tools that help all of our teams deliver so much value. Some of those tools are also listed in the bibliography.

Thanks to Mike Thomas for taking many of the action photos of an agile team that appear in this book. We hope these photos show those of you new to agile testing and development that there’s no big mystery—it’s just good people getting together to discuss, demo, and draw pictures.

Thanks so much to our Addison-Wesley editorial and production team who patiently answered many questions and turned this into the professional-looking book you see here, including Raina Chrobak, Chris Zahn, John Fuller, Sally Gregg, Bonnie Granat, Diane Freed, Jack Lewis, and Kim Arney.

Lisa's Story

I'm eternally grateful to Janet for agreeing to write this book with me. She kept us organized and on track so we could juggle book writing with our full-time jobs and personal lives. I'm fortunate to have a writing partner whose experience is complementary to mine. Like any successful agile project, this is a true team effort. This has been hard work, but thanks to Janet it has been a lot of fun, too.

I'd like to thank the members of my current team at ePlan Services Inc. (formerly known as Fast401k), which I joined (thanks to Mike Cohn, our first leader) in 2003. All of us learned so much working for Mike that first year, and it's a testament to his leadership that we continue to improve and help the business grow. Thanks to my awesome teammates who have each helped me become a better tester and agile team member, and who were all good sports while Mike Thomas took action photos of us: Nanda Lankapalli, Tony Sweets, Jeff Thuss, Lisa Owens, Mike Thomas, Vince Palumbo, Mike Busse, Nehru Kaja, Trevor Sterritt, Steve Kives, and former but still beloved team members Joe Yakich, Jason Kay, Jennifer Riefenberg, Matt Tierney, and Charles LeRose. I also have been lucky enough to work with the best customer team anywhere. They are too numerous to mention here, but many thanks to them, and in particular to Steve Perkins, Anne Olguin, and Zachary Shannon, who help us focus on delivering value. Thanks also to Mark and Dan Gutrich, founders and leaders of ePlan Services, for giving us all the opportunity to succeed with agile development.

Thanks to Kay and Zhon Johansen for teaching me about mind maps at Agile 2006. I hope we have put this skill to good use in creating this book.

Much gratitude to all my friends and family, whom I neglected terribly during the many months spent writing this book, and who nevertheless supported me constantly. There are too many to mention, but I must specially thank Anna Blake for her constant understanding and provision of donkey therapy. Chester and Ernest, the donkeys of my heart, have kept pulling me along. Dodger didn't make the whole book-writing journey in this world, but his memory continues to lift me up. My little poodle and muse Tango was by my side every minute that I worked on this book at home, joined occasionally by Bruno, Bubba, Olive, Squiggy, Starsky, Bobcat, and Patty. Thanks to my parents for being proud of me and not complaining about my neglect of them during this book-writing time.

I know that my husband, Bob Downing, took a deep breath when I exclaimed, "I have the chance to write another book about agile testing," but he nevertheless encouraged me constantly and made it possible for me to find the time to write. He kept the "no-kill shelter" running, kept our lives rolling, kept my spirits up, and sustained me with many fabulous meals. He is the light of my life.

—Lisa

Janet's Story

Lisa and I made a great team; each of us had our own strengths. When one of us faltered and took some time to recoup, the other picked up the slack. I learned so much from Lisa (thanks go to her for giving me this opportunity), but I also found I learned a lot from myself. Just the process of articulating my thoughts helped to clarify things that had been rumbling around in my brain for a long time. The mindmaps helped immensely, so thanks to Kenji Hiranabe, who gave a workshop at Agile 2007 and made me realize what a powerful yet simple tool mind maps can be.

This book-writing journey was an amazing experience. Thanks to the people on all the teams I worked with who provided so many of the examples in this book.

It's been a pretty special year all the way around. During the year (or so) it took to write this book, my family increased in size. My two daughters, Dana and Susan, each gave me a grandson—those were some of the times Lisa picked up the slack. I would like to thank my granddaughter Lauren (currently three) for making me leave my computer and play. It kept me sane. Thanks to my sister Colleen who gave me long-distance encouragement many mornings using instant messenger when I was feeling overwhelmed with the sheer number of hours I was putting in.

And a very special thanks to Jack, my husband, who moved his office downstairs when I took over the existing one. There were times when I am sure he felt neglected and wondered if he even had a wife as he spend many long hours alone. However, he was there with me the whole way, encouraging and supporting me in this endeavor.

—Janet

ABOUT THE AUTHORS

Lisa Crispin is an agile testing practitioner and coach. She specializes in showing testers and agile teams how testers can add value and how to guide development with business-facing tests. Her mission is to bring agile joy to the software testing world and testing joy to the agile development world. Lisa joined her first agile team in 2000, having enjoyed many years working as a programmer, analyst, tester, and QA director. Since 2003, she's been a tester on a Scrum/XP team at ePlan Services, Inc. She frequently leads tutorials and workshops on agile testing at conferences in North America and Europe. Lisa regularly contributes articles about agile testing to publications such as *Better Software* magazine, *IEEE Software*, and *Methods and Tools*. Lisa also co-authored *Testing Extreme Programming* (Addison-Wesley, 2002) with Tip House.

For more about Lisa's work, visit her websites, www.lisa.crispin.com and www.agiletester.ca, or email her at lisa@agiletester.ca.

Janet Gregory is the founder of DragonFire Inc., an agile quality process consultancy and training firm. Her passion is helping teams build quality systems. For the past ten years, she has worked as a coach and tester, introducing agile practices into both large and small companies. Her focus is working with business users and testers to understand their role in agile projects. Janet's programming background is a definite plus when she partners with developers on her agile teams to implement innovative agile test automation solutions. Janet is a frequent speaker at agile and testing software conferences, and she is a major contributor to the North American agile testing community.

For more about Janet's work, visit her websites at www.janetgregory.ca, www.janetgregory.blogspot.com, and www.agiletester.ca, or you can email her at janet@agiletester.ca.