

Extreme Programming Installed

Ron Jeffries
Ann Anderson
Chet Hendrickson



ADDISON-WESLEY

Boston • San Francisco • New York • Toronto • Montreal
London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Contents

<i>Foreword</i>	xiii
-----------------------	------

<i>Preface</i>	xv
----------------------	----

Chapter 1 <i>Extreme Programming</i>	1
---	---

Extreme Programming is a discipline of software development with values of simplicity, communication, feedback, and courage. We focus on the roles of customer, manager, and programmer and accord key rights and responsibilities to the people in those roles.

Chapter 2 <i>The Circle of Life</i>	13
--	----

An XP project succeeds when the customers select business value to be implemented, based on the team's measured ability to deliver functionality over time.

Chapter 3 <i>On-Site Customer</i>	17
--	----

An XP project needs a full-time customer to provide guidance. Here's a summary of why.

Chapter 4 <i>User Stories</i>	23
--	----

Define requirements with stories, written on cards.

Chapter 5 Acceptance Tests 31

Surely you aren't going to assume you're getting what you need. Prove that it works! Acceptance tests allow the customer to know when the system works and tell the programmers what needs to be done.

Sidebar Acceptance Test Samples 35

At first it can be difficult figuring out how to do acceptance tests. With a little practice, it becomes easy.

Chapter 6 Story Estimation 37

Customers need to know how much stories will cost in order to choose which ones to do and which to defer. Programmers evaluate stories to provide that information. Here's how.

Interlude Sense of Completion 45

XP's nested planning and programming cycles keep the project on track and provide a healthy sense of accomplishment at frequent intervals.

Chapter 7 Small Releases 49

The outermost XP cycle is the release. Small and frequent releases provide early benefit to the customer while providing early feedback to the programmers. Here are some thoughts on how to make it happen.

Chapter 8 Customer Defines Release 55

In each release cycle, the customer controls scope, deciding what to do and what to defer, to provide the best possible release by the due date. Work fits into the calendar based on business value, difficulty, and the team's implementation velocity.

Chapter 9 Iteration Planning 61

Inside each release, an Extreme team plans just a few weeks at a time with clear objectives and solid estimates.

Chapter 10 Quick Design Session 69

Within each iteration, programmers don't stand alone. Here's a technique to help programmers move forward with courage. Make it part of your team's ritual.

Chapter 11 Programming 71

It's called Extreme Programming, after all. Here's how we do the programming part of things.

Sidebar Code Quality 83

A little more detail on something close to our hearts: simplicity.

Chapter 12 Pair Programming 87

On an Extreme Programming team, two programmers sitting together at the same machine write all production code.

Chapter 13 Unit Tests 93

Extreme Programmers test everything that could possibly break, using automated tests that must run perfectly all the time.

Sidebar xUnit 105

Use the world's lightest testing tool.

Chapter 14 Test First, by Intention 107

Code what you want, not how to do it. Chet and Ron do a small task test first trying always to express intention in the code rather than algorithm.

Chapter 15 Releasing Changes 121

Using collective code ownership and comprehensive unit tests, an XP team releases changes rapidly and reliably.

Chapter 16 *Do or Do Not* 127

We've now covered most of the programming aspects of XP. Here's a summary of things we do—and things we don't do.

Chapter 17 *Experience Improves Estimates* 131

With each iteration, we gain experience. Experience with stories helps us estimate future stories more easily and more accurately.

Chapter 18 *Resources, Scope, Quality, Time* 135

Who's doing what? How much is finished? How good is it? When will we be done? What metrics should we keep?

Chapter 19 *Steering* 147

The estimates are wrong. Your priorities will change. You must steer.

Chapter 20 *Steering the Iteration* 151

To steer each iteration, you need to track how many stories are getting done and how well the task estimates are holding up.

Chapter 21 *Steering the Release* 157

To steer the release, you need to track what's done, how fast you are going, and how well the system works.

Chapter 22 *Handling Defects* 161

Report 'em, schedule 'em, test and fix 'em, avoid 'em. Just don't call 'em bugs.

Sidebar *Advanced Issue: Defect Databases* 165

Sidebar *Advanced Practice: Tests as Database* 169

Chapter 23	<i>Conclusion</i>	171
-------------------	-------------------	-----

BONUS TRACKS		175
---------------------	--	-----

Here are some things we've paid a lot to learn. Since you bought the album, we wanted to give you a little something extra. Thank you, and we hope we passed the audition.

Chapter 24	<i>We'll Try</i>	177
-------------------	------------------	-----

"We'll try" can be the saddest words a programmer has ever spoken, and most of us have spoken them more than once. We've covered this material in other forms already, but it bears repeating here.

Chapter 25	<i>How to Estimate Anything</i>	185
-------------------	---------------------------------	-----

Sometimes estimating stories seems scary. Keep your heads, stick together, and break the story down into small parts. You'll be surprised what you can do.

Chapter 26	<i>Infrastructure</i>	189
-------------------	-----------------------	-----

What about that database you need to build first? What about that framework? What about that syntax-directed command compiler? Get over it!

Chapter 27	<i>It's Chet's Fault</i>	193
-------------------	--------------------------	-----

Are you looking for someone to blame? This chapter explains how to know whose fault it is. Now move on and solve your problems.

Chapter 28	<i>Balancing Hopes and Fears</i>	195
-------------------	----------------------------------	-----

Those of you who have heard Ron, Ann, or me speak about XP are probably wondering where are all the war stories. Well, here's one.

Chapter 29	<i>Testing Improves Code</i>	199
-------------------	------------------------------	-----

An example showing how writing some tests can help you to improve the code.

Chapter 30 *XPer Tries Java* 203

After the C3 project ended, most of the team was transferred to work on the human resources intranet. I found how they were using the principles of XP to improve their lives on a new project heartening. What follows is a description of how Rich Garzaniti, exC3er and devoted XPer, is introducing testing and modern development tools into an environment where none existed.

Chapter 31 *A Java Perspective* 211

We would like to thank Bill Wake for allowing us to use this article. It is the second in a series entitled “The Test/Code Cycle in XP.” His website, <http://users.vnet.net/wwake>, contains the entire series plus a whole lot more.

Chapter 32 *A True Story* 225

Ron Jeffries [re]learns something about simplicity.

Chapter 33 *Estimates and Promises* 229

*We **estimate** how long the project will take. We **promise** to tell the truth about how we’re doing.*

Chapter 34 *Everything That Could Possibly Break* 233

Test everything that could possibly break. What does this mean? How is it possible?

Afterword 243

Annotated Bibliography 245

Index 261