
Continuous Delivery

Jez Humble and David Farley

◆◆Addison-Wesley

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

Contents

Foreword	xxi
Preface	xxiii
Acknowledgments	xxxix
About the Authors	xxxix
Part I: Foundations	1
Chapter 1: The Problem of Delivering Software	3
Introduction	3
Some Common Release Antipatterns	4
<i>Antipattern: Deploying Software Manually</i>	5
<i>Antipattern: Deploying to a Production-like Environment Only after Development Is Complete</i>	7
<i>Antipattern: Manual Configuration Management of Production Environments</i>	9
<i>Can We Do Better?</i>	10
How Do We Achieve Our Goal?	11
<i>Every Change Should Trigger the Feedback Process</i>	13
<i>The Feedback Must Be Received as Soon as Possible</i>	14
<i>The Delivery Team Must Receive Feedback and Then Act on It</i>	15
<i>Does This Process Scale?</i>	16
What Are the Benefits?	17
<i>Empowering Teams</i>	17
<i>Reducing Errors</i>	18
<i>Lowering Stress</i>	20
<i>Deployment Flexibility</i>	21
<i>Practice Makes Perfect</i>	22



The Release Candidate	22
<i>Every Check-in Leads to a Potential Release</i>	23
Principles of Software Delivery	24
<i>Create a Repeatable, Reliable Process for Releasing Software</i>	24
<i>Automate Almost Everything</i>	25
<i>Keep Everything in Version Control</i>	26
<i>If It Hurts, Do It More Frequently, and Bring the Pain Forward</i>	26
<i>Build Quality In</i>	27
<i>Done Means Released</i>	27
<i>Everybody Is Responsible for the Delivery Process</i>	28
<i>Continuous Improvement</i>	28
Summary	29
Chapter 2: Configuration Management	31
Introduction	31
Using Version Control	32
<i>Keep Absolutely Everything in Version Control</i>	33
<i>Check In Regularly to Trunk</i>	35
<i>Use Meaningful Commit Messages</i>	37
Managing Dependencies	38
<i>Managing External Libraries</i>	38
<i>Managing Components</i>	39
Managing Software Configuration	39
<i>Configuration and Flexibility</i>	40
<i>Types of Configuration</i>	41
<i>Managing Application Configuration</i>	43
<i>Managing Configuration across Applications</i>	47
<i>Principles of Managing Application Configuration</i>	47
Managing Your Environments	49
<i>Tools to Manage Environments</i>	53
<i>Managing the Change Process</i>	53
Summary	54
Chapter 3: Continuous Integration	55
Introduction	55
Implementing Continuous Integration	56
<i>What You Need Before You Start</i>	56
<i>A Basic Continuous Integration System</i>	57

Prerequisites for Continuous Integration	59
<i>Check In Regularly</i>	59
<i>Create a Comprehensive Automated Test Suite</i>	60
<i>Keep the Build and Test Process Short</i>	60
<i>Managing Your Development Workspace</i>	62
Using Continuous Integration Software	63
<i>Basic Operation</i>	63
<i>Bells and Whistles</i>	63
Essential Practices	66
<i>Don't Check In on a Broken Build</i>	66
<i>Always Run All Commit Tests Locally before Committing, or Get Your</i> <i>CI Server to Do It for You</i>	66
<i>Wait for Commit Tests to Pass before Moving On</i>	67
<i>Never Go Home on a Broken Build</i>	68
<i>Always Be Prepared to Revert to the Previous Revision</i>	69
<i>Time-Box Fixing before Reverting</i>	70
<i>Don't Comment Out Failing Tests</i>	70
<i>Take Responsibility for All Breakages That Result from Your Changes</i> .	70
<i>Test-Driven Development</i>	71
Suggested Practices	71
<i>Extreme Programming (XP) Development Practices</i>	71
<i>Failing a Build for Architectural Breaches</i>	72
<i>Failing the Build for Slow Tests</i>	73
<i>Failing the Build for Warnings and Code Style Breaches</i>	73
Distributed Teams	75
<i>The Impact on Process</i>	75
<i>Centralized Continuous Integration</i>	76
<i>Technical Issues</i>	76
<i>Alternative Approaches</i>	77
Distributed Version Control Systems	79
Summary	82
Chapter 4: Implementing a Testing Strategy	83
Introduction	83
Types of Tests	84
<i>Business-Facing Tests That Support the Development Process</i>	85
<i>Technology-Facing Tests That Support the Development Process</i>	89
<i>Business-Facing Tests That Critique the Project</i>	89

<i>Technology-Facing Tests That Critique the Project</i>	91
<i>Test Doubles</i>	91
Real-Life Situations and Strategies	92
<i>New Projects</i>	92
<i>Midproject</i>	94
<i>Legacy Systems</i>	95
<i>Integration Testing</i>	96
Process	99
<i>Managing Defect Backlogs</i>	100
Summary	101
Part II: The Deployment Pipeline	103
Chapter 5: Anatomy of the Deployment Pipeline	105
Introduction	105
What Is a Deployment Pipeline?	106
<i>A Basic Deployment Pipeline</i>	111
Deployment Pipeline Practices	113
<i>Only Build Your Binaries Once</i>	113
<i>Deploy the Same Way to Every Environment</i>	115
<i>Smoke-Test Your Deployments</i>	117
<i>Deploy into a Copy of Production</i>	117
<i>Each Change Should Propagate through the Pipeline Instantly</i>	118
<i>If Any Part of the Pipeline Fails, Stop the Line</i>	119
The Commit Stage	120
<i>Commit Stage Best Practices</i>	121
The Automated Acceptance Test Gate	122
<i>Automated Acceptance Test Best Practices</i>	124
Subsequent Test Stages	126
<i>Manual Testing</i>	128
<i>Nonfunctional Testing</i>	128
Preparing to Release	128
<i>Automating Deployment and Release</i>	129
<i>Backing Out Changes</i>	131
<i>Building on Success</i>	132
Implementing a Deployment Pipeline	133
<i>Modeling Your Value Stream and Creating a Walking Skeleton</i>	133
<i>Automating the Build and Deployment Process</i>	134

<i>Automating the Unit Tests and Code Analysis</i>	135
<i>Automating Acceptance Tests</i>	136
<i>Evolving Your Pipeline</i>	136
Metrics	137
Summary	140
Chapter 6: Build and Deployment Scripting	143
Introduction	143
An Overview of Build Tools	144
<i>Make</i>	146
<i>Ant</i>	147
<i>NAnt and MSBuild</i>	148
<i>Maven</i>	149
<i>Rake</i>	150
<i>Buildr</i>	151
<i>Psake</i>	151
Principles and Practices of Build and Deployment Scripting	152
<i>Create a Script for Each Stage in Your Deployment Pipeline</i>	152
<i>Use an Appropriate Technology to Deploy Your Application</i>	152
<i>Use the Same Scripts to Deploy to Every Environment</i>	153
<i>Use Your Operating System's Packaging Tools</i>	154
<i>Ensure the Deployment Process Is Idempotent</i>	155
<i>Evolve Your Deployment System Incrementally</i>	157
Project Structure for Applications That Target the JVM	157
<i>Project Layout</i>	157
Deployment Scripting	160
<i>Deploying and Testing Layers</i>	162
<i>Testing Your Environment's Configuration</i>	163
Tips and Tricks	164
<i>Always Use Relative Paths</i>	164
<i>Eliminate Manual Steps</i>	165
<i>Build In Traceability from Binaries to Version Control</i>	165
<i>Don't Check Binaries into Version Control as Part of Your Build</i>	166
<i>Test Targets Should Not Fail the Build</i>	166
<i>Constrain Your Application with Integrated Smoke Tests</i>	167
<i>.NET Tips and Tricks</i>	167
Summary	168

Chapter 7: The Commit Stage	169
Introduction	169
Commit Stage Principles and Practices	170
<i>Provide Fast, Useful Feedback</i>	171
<i>What Should Break the Commit Stage?</i>	172
<i>Tend the Commit Stage Carefully</i>	172
<i>Give Developers Ownership</i>	173
<i>Use a Build Master for Very Large Teams</i>	174
The Results of the Commit Stage	174
<i>The Artifact Repository</i>	175
Commit Test Suite Principles and Practices	177
<i>Avoid the User Interface</i>	178
<i>Use Dependency Injection</i>	179
<i>Avoid the Database</i>	179
<i>Avoid Asynchrony in Unit Tests</i>	180
<i>Using Test Doubles</i>	180
<i>Minimizing State in Tests</i>	183
<i>Faking Time</i>	184
<i>Brute Force</i>	185
Summary	185
Chapter 8: Automated Acceptance Testing	187
Introduction	187
Why Is Automated Acceptance Testing Essential?	188
<i>How to Create Maintainable Acceptance Test Suites</i>	190
<i>Testing against the GUI</i>	192
Creating Acceptance Tests	193
<i>The Role of Analysts and Testers</i>	193
<i>Analysis on Iterative Projects</i>	193
<i>Acceptance Criteria as Executable Specifications</i>	195
The Application Driver Layer	198
<i>How to Express Your Acceptance Criteria</i>	200
<i>The Window Driver Pattern: Decoupling the Tests from the GUI</i>	201
Implementing Acceptance Tests	204
<i>State in Acceptance Tests</i>	204
<i>Process Boundaries, Encapsulation, and Testing</i>	206
<i>Managing Asynchrony and Timeouts</i>	207
<i>Using Test Doubles</i>	210

The Acceptance Test Stage	213
<i>Keeping Acceptance Tests Green</i>	214
<i>Deployment Tests</i>	217
Acceptance Test Performance	218
<i>Refactor Common Tasks</i>	219
<i>Share Expensive Resources</i>	219
<i>Parallel Testing</i>	220
<i>Using Compute Grids</i>	220
Summary	222
Chapter 9: Testing Nonfunctional Requirements	225
Introduction	225
Managing Nonfunctional Requirements	226
<i>Analyzing Nonfunctional Requirements</i>	227
Programming for Capacity	228
Measuring Capacity	231
<i>How Should Success and Failure Be Defined for Capacity Tests?</i>	232
The Capacity-Testing Environment	234
Automating Capacity Testing	238
<i>Capacity Testing via the User Interface</i>	240
<i>Recording Interactions against a Service or Public API</i>	241
<i>Using Recorded Interaction Templates</i>	241
<i>Using Capacity Test Stubs to Develop Tests</i>	244
Adding Capacity Tests to the Deployment Pipeline	244
Additional Benefits of a Capacity Test System	247
Summary	248
Chapter 10: Deploying and Releasing Applications	249
Introduction	249
Creating a Release Strategy	250
<i>The Release Plan</i>	251
<i>Releasing Products</i>	252
Deploying and Promoting Your Application	253
<i>The First Deployment</i>	253
<i>Modeling Your Release Process and Promoting Builds</i>	254
<i>Promoting Configuration</i>	257
<i>Orchestration</i>	258
<i>Deployments to Staging Environments</i>	258

Rolling Back Deployments and Zero-Downtime Releases	259
<i>Rolling Back by Redeploying the Previous Good Version</i>	260
Zero-Downtime Releases	260
Blue-Green Deployments	261
Canary Releasing	263
Emergency Fixes	265
Continuous Deployment	266
<i>Continuously Releasing User-Installed Software</i>	267
Tips and Tricks	270
<i>The People Who Do the Deployment Should Be Involved in Creating the Deployment Process</i>	270
Log Deployment Activities	271
Don't Delete the Old Files, Move Them	271
Deployment Is the Whole Team's Responsibility	271
Server Applications Should Not Have GUIs	271
Have a Warm-Up Period for a New Deployment	272
Fail Fast	273
Don't Make Changes Directly on the Production Environment	273
Summary	273
Part III: The Delivery Ecosystem	275
Chapter 11: Managing Infrastructure and Environments	277
Introduction	277
Understanding the Needs of the Operations Team	279
Documentation and Auditing	280
Alerts for Abnormal Events	281
IT Service Continuity Planning	282
Use the Technology the Operations Team Is Familiar With	282
Modeling and Managing Infrastructure	283
Controlling Access to Your Infrastructure	285
Making Changes to Infrastructure	287
Managing Server Provisioning and Configuration	288
Provisioning Servers	288
Ongoing Management of Servers	290
Managing the Configuration of Middleware	295
Managing Configuration	296
Research the Product	298
Examine How Your Middleware Handles State	298

<i>Look for a Configuration API</i>	299
<i>Use a Better Technology</i>	299
Managing Infrastructure Services	300
<i>Multihomed Systems</i>	301
Virtualization	303
<i>Managing Virtual Environments</i>	305
<i>Virtual Environments and the Deployment Pipeline</i>	308
<i>Highly Parallel Testing with Virtual Environments</i>	310
Cloud Computing	312
<i>Infrastructure in the Cloud</i>	313
<i>Platforms in the Cloud</i>	314
<i>One Size Doesn't Have to Fit All</i>	315
<i>Criticisms of Cloud Computing</i>	316
Monitoring Infrastructure and Applications	317
<i>Collecting Data</i>	318
<i>Logging</i>	320
<i>Creating Dashboards</i>	321
<i>Behavior-Driven Monitoring</i>	323
Summary	323
Chapter 12: Managing Data	325
Introduction	325
Database Scripting	326
<i>Initializing Databases</i>	327
Incremental Change	327
<i>Versioning Your Database</i>	328
<i>Managing Orchestrated Changes</i>	329
Rolling Back Databases and Zero-Downtime Releases	331
<i>Rolling Back without Losing Data</i>	331
<i>Decoupling Application Deployment from Database Migration</i>	333
Managing Test Data	334
<i>Faking the Database for Unit Tests</i>	335
<i>Managing the Coupling between Tests and Data</i>	336
<i>Test Isolation</i>	337
<i>Setup and Tear Down</i>	337
<i>Coherent Test Scenarios</i>	337
Data Management and the Deployment Pipeline	338
<i>Data in Commit Stage Tests</i>	338

<i>Data in Acceptance Tests</i>	339
<i>Data in Capacity Tests</i>	341
<i>Data in Other Test Stages</i>	342
Summary	343
Chapter 13: Managing Components and Dependencies	345
Introduction	345
Keeping Your Application Releasable	346
<i>Hide New Functionality Until It Is Finished</i>	347
<i>Make All Changes Incrementally</i>	349
<i>Branch by Abstraction</i>	349
Dependencies	351
<i>Dependency Hell</i>	352
<i>Managing Libraries</i>	354
Components	356
<i>How to Divide a Codebase into Components</i>	356
<i>Pipelining Components</i>	360
<i>The Integration Pipeline</i>	361
Managing Dependency Graphs	363
<i>Building Dependency Graphs</i>	363
<i>Pipelining Dependency Graphs</i>	365
<i>When Should We Trigger Builds?</i>	369
<i>Cautious Optimism</i>	370
<i>Circular Dependencies</i>	372
Managing Binaries	373
<i>How an Artifact Repository Should Work</i>	373
<i>How Your Deployment Pipeline Should Interact with the Artifact Repository</i>	374
Managing Dependencies with Maven	375
<i>Maven Dependency Refactorings</i>	377
Summary	379
Chapter 14: Advanced Version Control	381
Introduction	381
A Brief History of Revision Control	382
CVS	382
Subversion	383
Commercial Version Control Systems	385
Switch Off Pessimistic Locking	386

Branching and Merging	388
<i>Merging</i>	389
<i>Branches, Streams, and Continuous Integration</i>	390
Distributed Version Control Systems	393
<i>What Is a Distributed Version Control System?</i>	393
<i>A Brief History of Distributed Version Control Systems</i>	395
<i>Distributed Version Control Systems in Corporate Environments</i>	396
<i>Using Distributed Version Control Systems</i>	397
Stream-Based Version Control Systems	399
<i>What Is a Stream-Based Version Control System?</i>	399
<i>Development Models with Streams</i>	400
<i>Static and Dynamic Views</i>	403
<i>Continuous Integration with Stream-Based Version Control Systems</i> ...	403
Develop on Mainline	405
<i>Making Complex Changes without Branching</i>	406
Branch for Release	408
Branch by Feature	410
Branch by Team	412
Summary	415
Chapter 15: Managing Continuous Delivery	417
Introduction	417
A Maturity Model for Configuration and Release Management	419
<i>How to Use the Maturity Model</i>	419
Project Lifecycle	421
<i>Identification</i>	422
<i>Inception</i>	423
<i>Initiation</i>	424
<i>Develop and Release</i>	425
<i>Operation</i>	428
A Risk Management Process	429
<i>Risk Management 101</i>	429
<i>Risk Management Timeline</i>	430
<i>How to Do a Risk-Management Exercise</i>	431
Common Delivery Problems—Their Symptoms and Causes	432
<i>Infrequent or Buggy Deployments</i>	433
<i>Poor Application Quality</i>	434
<i>Poorly Managed Continuous Integration Process</i>	435

<i>Poor Configuration Management</i>	436
Compliance and Auditing	436
<i>Automation over Documentation</i>	437
<i>Enforcing Traceability</i>	438
<i>Working in Silos</i>	439
<i>Change Management</i>	440
Summary	442
Bibliography	443
Index	445