

A grid of small, light blue dots arranged in a regular pattern across the top of the cover.

**ALAN S. KOCH**

Several large, stylized concentric circles in shades of blue and yellow, some partially cut off by the edges of the cover, creating a dynamic, geometric background.

# **AGILE SOFTWARE DEVELOPMENT**

**Evaluating the Methods for Your Organization**

A grid of small, light blue dots arranged in a regular pattern across the bottom of the cover.

# Contents

Foreword by Kent Beck . . . . .	<i>xxi</i>
Foreword by Mark Paulk . . . . .	<i>xxiii</i>
Preface . . . . .	<i>xxvi</i>
Part I	
Adoption Considerations . . . . .	1
<b>1</b> Introducing the Agile Methods . . . . .	3
Historical and background information	3
The Agile methods, generally	4
Agility	5
Change	5
Planning	5
Communication	6
Learning	6
The Agile methods, specifically	7
Reference	8
<b>2</b> Considering Your Organizational Culture . . . . .	9
Hierarchical versus cooperative organizations	9
Hierarchical organizations	10
Cooperative organizations	10
Considerations: Hierarchical versus cooperative	10
Controlling change versus reacting to it	11
Controlling change	11
Reacting to change	12
Considerations: Controlling versus reacting	12
The role of organizational culture	13

<b>3</b>	<b>Considering Your Customers. . . . .</b>	<b>15</b>
	Contracts and statements of work	15
	Establishing and changing requirements	16
	Expectations about collaboration	18
	Your customers	19
<b>4</b>	<b>Considering Your Projects . . . . .</b>	<b>21</b>
	Size of project teams	21
	Colocation of team members	22
	Criticality of projects	23
	Safety and security requirements	24
	Multiple teams	25
	Subcontractors	26
	Integration with hardware and other software components	26
<b>5</b>	<b>Considering Your Tools and Processes . . . . .</b>	<b>29</b>
	Requirements Management	29
	Project Management	31
	Configuration Management	32
	Code control	33
	Document control	33
	Baseline maintenance	33
	Configuration Item identification	34
	Change control	34
	Build and release management	34
	Your tools and processes	35
<b>6</b>	<b>Considering Your Staff. . . . .</b>	<b>37</b>
	Superstars	37
	Changing work patterns	38
	Making changes stick	39
	Making the right change	39
	Building buy-in	40
	Changing the reward system	41
	Your staff	41
<b>7</b>	<b>Using This Book to Make Your Adoption Decisions .</b>	<b>43</b>
	Structure of this book	43
	The “Evaluating Agile Methods” Workbook	45

Evaluating the practices	46
Compiling the results	48
Final steps	49

<b>Part II</b>	
<b>Value: “Individuals and Interactions over Processes and Tools” . . . . .</b>	<b>51</b>

<b>8</b>	<b>About People, Processes, and Tools . . . . .</b>	<b>53</b>
	People versus processes versus tools	53
	The role of people	54
	The role of processes	55
	The role of tools	57
	Balancing people, process, and tools	58

<b>9</b>	<b>Motivated Individuals and Self-Organizing Teams . . . . .</b>	<b>61</b>
	Agile Principles	61
	Motivated individuals	61
	Self-organizing teams	62
	Agile practices	62
	Adaptive Software Development	63
	The Adaptive Conceptual Model: Project stakeholders as independent agents	63
	The Adaptive Development Model: Speculate: Project initiation and adaptive planning	63
	The Adaptive (Leadership-Collaboration) Management Model	64
	Dynamic System Development Method (DSDM)	64
	Principle 2: DSDM teams must be empowered to make decisions	64
	Extreme Programming (XP)	64
	The Planning Game	64
	Collective ownership	65
	Feature-Driven Development	65
	Class (code) ownership	65
	Feature teams	65
	Lean Software Development (LD)	66
	Empower the Team: Tool 13, Self-determination	66
	Empower the Team: Tool 14, Motivation	66
	Scrum	66
	Scrum teams	66
	Adoption implications	67
	Trusting the technical team	67
	Staffing with “motivated individuals”	68

Team structure and roles	68
Pair Programming	69
Chief Programmer	69
Method Coach	70
Project Manager	71
Motivated individuals and self-organizing teams	71

## **10** Face-to-Face Communication . . . . . 73

Agile Principle	73
Face-to-face communication	73
Agile practices	74
Extreme Programming	74
Facilities Strategy	74
Pair Programming	75
On-Site Customer	76
The Planning Game	76
Scrum	76
Daily Scrum Meetings	76
Adoption implications	77
Richness	77
Memory	79
Persistence	79
Availability	80
Communication	<b>81</b>

## **11** Sustainable Pace . . . . . 83

Agile Principle	83
Sustainable pace	83
Agile practices	84
Extreme Programming (XP)	84
40-hour week	84
Adoption implications	84
Overtime versus the Agile methods	85
Initial analysis	85
Incremental development	86
Testing	87
Integration	87
A sustainable pace	88

## **12** The Unstated Principle: Appropriate Processes and Tools . . . . . 89

Agile practices	90
Feature-Driven Development (FDD)	90
Configuration Management	90
Lean Software Development (LD)	90
Amplify Learning: Tool 5, Synchronization	90
Deliver as Fast as Possible: Tool 10, Pull Systems	92
Deliver as Fast as Possible: Tool 11, Queuing Theory	92
Deliver as Fast as Possible: Tool 12, Cost of Delay	92
See the Whole: Tool 21, Measurements	93
Adoption implications	93
Processes	93
Configuration Management	94
Code control	94
Establishing baselines	95
Change requests	95
Configuration integrity	96
Build automation	96
Test automation	97
Processes and tools	97
Reference	97

Part III

Value: “Working Software over Comprehensive Documentation” . . . . . 99

<b>13</b>	<b>The Role of Documentation in a Software Project . . . . .</b>	<b>101</b>
	Purpose of a document	101
	Audience for a document	102
	Value of a document versus its cost	103
	Avoiding waste in documentation	104

<b>14</b>	<b>Incremental Delivery of Working Software . . . . .</b>	<b>105</b>
	Agile Principles	105
	Early and continuous delivery	105
	Deliver working software frequently	106
	Working software: Primary measure of progress	107
	Agile practices	107
	Adaptive Software Development	107
	The Adaptive Life Cycle	107
	Learn: Quality Review: Customer Focus Group Reviews	109
	Dynamic Systems Development Method	109

3) Frequent delivery	109
4) Fitness for business purpose	109
5) Iterative and incremental development	110
Extreme Programming	110
Small releases	110
Continuous integration	110
Feature-Driven Development	111
Developing by feature	111
Regular build schedule	111
Reporting/Visibility of results	111
Lean Software Development	112
Amplify Learning: Tool 3, Feedback	112
Amplify Learning: Tool 4, Iterations	112
Scrum	113
Sprint	113
Sprint Review	113
Adoption implications	114
Time-boxed development	114
Continuous integration	115
Incremental delivery	115
Incremental development versus hacking	116
Deliver working software to whom?	116
Minimizing documentation	117
Incremental development	118
Reference	118

**Part IV**  
**Value: “Customer Collaboration over Contract Negotiation” . . . . . 119**

<b>15</b>	<b>Defining the Customer Relationship . . . . . 121</b>
	Types of customers 121
	Role of contracts 122
	Role of ongoing collaboration 123
	Balancing contracts and collaboration 124

<b>16</b>	<b>Daily Collaboration of All Stakeholders . . . . . 127</b>
	Agile Principle 127
	All Stakeholders Must Work Together Daily 127
	Agile practices 128
	Adaptive Software Development (ASD) 128

Project stakeholders as independent agents	129
Adaptive (Leadership-Collaboration) Management Model	129
Dynamic Systems Development Method (DSDM)	129
Active user involvement	130
Collaborative and cooperative approach	130
Extreme Programming (XP)	130
On-site customer	130
Lean Software Development (LD)	131
Build Integrity In: Tools 17 and 18, Perceived and Conceptual Integrity	131
See The Whole: Tool 22, Contracts	131
Scrum	131
Product Backlog	131
Adoption implications	132
Establishing requirements	132
Managing requirements changes	133
Ensuring product quality	133
Acceptance	134
The reluctant customer	135
Project course corrections	135
Contract as a weapon	136
Customer collaboration	137
Reference	137

**Part V**  
**Value: “Responding to Change over Following a Plan” . . . . . 139**

<b>17</b>	<b>Understanding Change in Software Projects . . . . . 141</b>
	The nature of change . . . . . 141
	External changes . . . . . 142
	Internal change: customers learn . . . . . 143
	Internal change: developers learn . . . . . 143
	Capitalizing on what we learn . . . . . 144
	Planning for change . . . . . 144
	Change happens . . . . . 145

<b>18</b>	<b>Welcome Changing Requirements . . . . . 147</b>
	Agile Principle . . . . . 147
	Welcome changing requirements . . . . . 147
	Agile practices . . . . . 148
	Adaptive Software Development (ASD) . . . . . 148



Adaptive Life Cycle	148
Dynamic Systems Development Method (DSDM)	148
All changes are reversible	148
Requirements are baselined at a high level	149
Extreme Programming (XP)	150
Metaphor	150
Refactoring	150
Feature-Driven Development (FDD)	151
Domain Object Modeling	151
Lean Software Development (LD)	151
Decide as Late as Possible: Tool 7, Options Thinking, Tool 8, The Last Responsible Moment, Tool 9, Making Decisions	152
Build Integrity In: Tool 19, Refactoring	152
Scrum	153
Sprint Planning Meeting	153
Adoption implications	153
Incremental planning	154
Tracking and reporting progress	154
When the project deviates from the plan	155
Handling customer change requests	155
Changes injected by the development team	156
Welcoming change	156
Reference	157

## Part VI

### The Unstated Value: Keeping the Process Agile . . . 159

#### **19** Maintaining the Process . . . . . 161

Agile is not antiprocess	161
You are using a process	162
Process efficiency and effectiveness	162
Mary, Mary, quite contrary, how does your [process] grow?	163
Continuous process improvement	164

#### **20** Technical Excellence . . . . . 165

Agile Principle	165
Continuous attention to technical excellence and good design	165
Agile practices	166
Adaptive Software Development (ASD)	166
Learn: Quality Review: Software inspections	166
Dynamic Systems Development Method (DSDM)	167

Testing throughout the life cycle	167
Extreme Programming (XP)	168
Test First	168
Coding Standards	169
Feature-Driven Development (FDD)	170
Inspections	170
Lean Software Development (LD)	170
Amplify Learning: Tool 6, Set-Based Development	170
Empower the Team: Tool 15, Leadership	170
Empower the Team: Tool 16, Expertise	171
Build Integrity In: Tool 20, Testing	171
Scrum	172
Scrum Master	172
Adoption implications	172
Project roles	173
Developers' attention to quality	174
Technical excellence	174
Reference	175

<b>21</b>	<b>Simplicity . . . . .</b>	<b>177</b>
	Agile Principle	177
	Simplicity: Maximizing work not done	177
	Agile practices	177
	Extreme Programming (XP)	178
	Simple Design	178
	Lean Software Development (LD)	178
	Eliminate Waste: Tool 1, Seeing Waste and Tool 2, Value Stream Mapping	179
	Adoption implications	180
	Object-Orientation	180
	Identifying the expendable	181
	Simplicity	182
	Reference	182

<b>22</b>	<b>Retrospectives . . . . .</b>	<b>183</b>
	Agile Principle	183
	Regular team retrospectives	183
	Agile practices	184
	Adaptive Software Development (ASD)	184
	Learn: Quality Review: Postmortems	184
	Adoption implications	185

When to hold a retrospective	185
How to capitalize on a retrospective	186
Process change in mid-project	186
Conclusion	187

## Part VII

### The Adoption Decision . . . . . 189

## 23

### Making the Adoption Decision . . . . . 191

Compiling your “Evaluating Agile Methods Workbook” data	191
Conclusions about Agile Values and Principles	192
We have come to value individuals and interactions over processes and tools	193
We have come to value working software over comprehensive documentation	195
We have come to value customer collaboration over contract negotiation	196
We have come to value responding to change over following a plan	197
The unstated value: Keeping the process agile	198
Agile Values in your organization	200
Conclusions about the Agile Methods and Practices	200
Adaptive Software Development	200
Dynamic Systems Development Method	201
Extreme Programming	202
Feature-Driven Development	203
Lean Software Development	203
Scrum	205
The Agile Methods in your organization	205
Marketing your conclusions in your organization	206
Agreeing together on an action plan	207

## 24

### Adopting New Practices . . . . . 209

Three critical things to do: communicate, communicate, communicate	209
1. Communicate while making the decision.	210
2. Communicate about the decision you made.	210
3. Communicate regularly about the status of the change effort.	211
Crafting your custom Agile Method	212
Training those who will be affected	213
Pilot testing the new method	214
Just-in-time training	214
Expert on call	214
Celebrate project milestones	214
Improving the Agile Method	215

Rolling it out to the whole organization	215
<b>25 Evaluating the Effects of Your Agile Method . . . . .</b>	<b>217</b>
Project performance	217
Management acceptance	219
Customer relationship	220
Team satisfaction	221
Continuously improving your Agile Method	222
<b>Appendix Introduction . . . . .</b>	<b>223</b>
<b>Appendix A</b>	
<b>The Agile Manifesto . . . . .</b>	<b>225</b>
Reference	226
<b>Appendix B</b>	
<b>The 12 Principles of Agile Methods . . . . .</b>	<b>227</b>
The 12 Principles of Agile Methods	227
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	228
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	228
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter time scale.	229
Business people and developers must work together daily throughout the project.	229
Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	229
The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	229
Working software is the primary measure of progress.	230
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	230
Continuous attention to technical excellence and good design enhances agility.	230
Simplicity—the art of maximizing the amount of work not done —is essential.	230
The best architectures, requirements, and designs emerge from self-organizing teams.	231
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	231
Agile Principles	231

## **Appendix C**

### **Adaptive Software Development . . . . . 233**

The Adaptive Life Cycle	233
Speculate: Project initiation	234
Learning Loop	234
Speculate: Adaptive Cycle Planning	235
Collaborate: Concurrent component engineering	235
Learn: Quality Review	235
Learn: Final Q/A and release	236
ASD's conceptual framework	236
Project stakeholders as independent agents	236
The Adaptive (Leadership-Collaboration) Management Model	236
ASD	238
References	238

## **Appendix D**

### **Dynamic Systems Development Method . . . . . 239**

The DSDM process	239
Nine principles of DSDM	241
Principle 1: Active user involvement is imperative.	241
Principle 2: DSDM teams must be empowered to make decisions.	241
Principle 3: The focus is on frequent delivery of products.	241
Principle 4: Fitness for business purpose is the essential criterion for acceptance of deliverables.	242
Principle 5: Iterative and incremental development is necessary to converge on an accurate business solution.	242
Principle 6: All changes during development are reversible.	242
Principle 7: Requirements are baselined at a high level.	242
Principle 8: Testing is integrated throughout the life cycle.	243
Principle 9: A collaborative and cooperative approach between all stakeholders is essential.	243
Reference	243

## **Appendix E**

### **Extreme Programming . . . . . 245**

XP's 12 practices	245
The Planning Game	245
Small releases	245
Metaphor	246
Simple design	246
Test First	246
Refactoring	247

Pair Programming	247
Collective ownership	247
Continuous integration	247
40-hour week	248
On-site customer	248
Coding standards	248
The XP Facilities Strategy	248
References	248

**Appendix F**  
**Feature-Driven Development . . . . . 249**

FDD practices	249
Domain Object Modeling	249
Developing by feature	249
Class (code) ownership	250
Feature teams	250
Inspections	251
Regular build schedule	251
Configuration Management	251
Reporting/Visibility of results	251
References	252

**Appendix G**  
**Lean Software Development. . . . . 253**

Lean Software Development principles and tools	253
Eliminate Waste	253
Amplify Learning	254
Decide as Late as Possible	254
Deliver as Fast as Possible	255
Empower the Team	255
Build Integrity In	256
See the Whole	256
References	256

**Appendix H**  
**Scrum . . . . . 257**

Scrum practices	257
The Scrum Master	257
Product Backlog	258
Scrum Teams	258
Daily Scrum Meetings	259
Sprint Planning Meeting	259

Sprint	259
Sprint Review	260
References	260
<b>Glossary . . . . .</b>	<b>261</b>
<b>About the Author . . . . .</b>	<b>267</b>
<b>Index . . . . .</b>	<b>269</b>