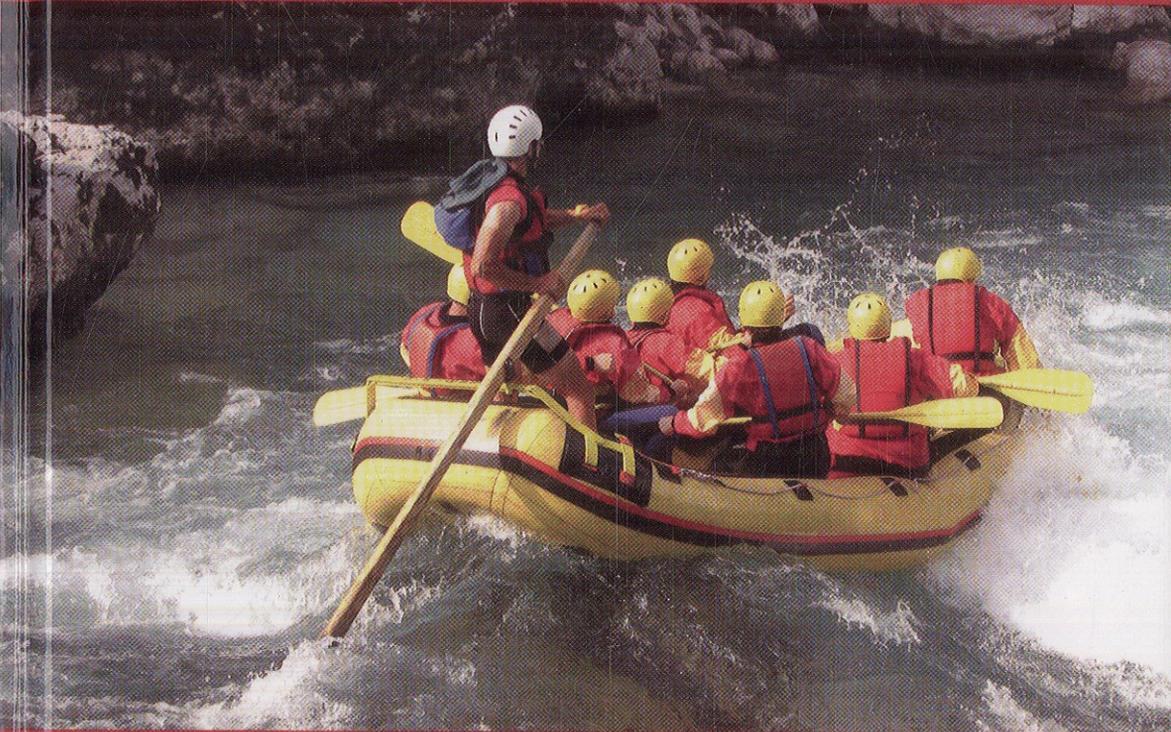


CHAPMAN & HALL/CRC INNOVATIONS IN  
SOFTWARE ENGINEERING AND SOFTWARE DEVELOPMENT

# SOFTWARE DEVELOPMENT

AN OPEN SOURCE APPROACH



ALLEN TUCKER  
RALPH MORELLI  
CHAMINDRA DE SILVA



CRC Press

Taylor & Francis Group

A CHAPMAN & HALL BOOK

# Contents

List of Figures . . . . .	xi
List of Tables . . . . .	xv
Preface . . . . .	xvii
Acknowledgments . . . . .	xxv
Authors . . . . .	xxvii
<b>1 Overview and Motivation . . . . .</b>	<b>1</b>
1.1 Software . . . . .	1
1.1.1 Types of Software . . . . .	2
1.1.2 The Changing Landscape . . . . .	5
1.1.3 Who Are the Developers? . . . . .	7
1.1.4 Strategic Choices . . . . .	8
1.2 Free and Open Source Software (FOSS) . . . . .	11
1.2.1 Origins and Growth . . . . .	12
1.2.2 Licensing . . . . .	16
1.2.3 Worldwide Impact . . . . .	18
1.2.4 Humanitarian FOSS . . . . .	19
1.3 Two Case Studies . . . . .	20
1.3.1 RMH Homebase . . . . .	20
1.3.2 Sahana . . . . .	20
1.4 Summary . . . . .	22
Exercises . . . . .	23
<b>2 Working with a Project Team . . . . .</b>	<b>27</b>
2.1 Key FOSS Activities . . . . .	27
2.1.1 Agile Development . . . . .	27
2.1.2 Using Patterns . . . . .	29
2.1.3 Reading and Writing Code . . . . .	31
2.1.4 Documentation . . . . .	34
2.1.5 On-Line Help . . . . .	37
2.2 Client-Oriented vs Community-Oriented Projects . . . . .	37
2.2.1 Project Evolution . . . . .	40
2.2.2 Similarities and Differences . . . . .	42
2.3 Working on a Client-Oriented Project . . . . .	44
2.3.1 Members, Roles, and Tasks . . . . .	44
2.3.2 Team Dynamics . . . . .	47
2.3.3 Scheduling, Milestones, and To-Do Lists . . . . .	48
2.4 Joining a Community-Oriented Project . . . . .	50

2.4.1	Project Selection . . . . .	52
2.4.2	First Contact with the Project . . . . .	53
2.4.3	Norms for Good Citizenship . . . . .	56
2.4.4	Becoming a User First . . . . .	58
2.5	Summary . . . . .	60
	Exercises . . . . .	60
<b>3</b>	<b>Using Project Tools</b>	<b>63</b>
3.1	Collaboration Tools . . . . .	63
3.1.1	Asynchronous Communication . . . . .	64
3.1.2	Synchronous Communication . . . . .	65
3.1.3	Shared Documents . . . . .	66
3.2	Code Management Tools . . . . .	67
3.2.1	The IDE . . . . .	68
3.2.2	The Software Stack . . . . .	70
3.2.3	The Version Control System . . . . .	72
3.2.4	The Bug Tracker . . . . .	77
3.3	Run-Time System Constraints . . . . .	82
3.3.1	Performance . . . . .	82
3.3.2	Web Hosting . . . . .	83
3.3.3	Licensing . . . . .	83
3.3.4	Platform . . . . .	84
3.4	Summary . . . . .	84
	Exercises . . . . .	85
<b>4</b>	<b>Software Architecture</b>	<b>87</b>
4.1	Architectural Patterns . . . . .	87
4.2	Layers, Cohesion, and Coupling . . . . .	89
4.2.1	Using Metrics to Evaluate Cohesion and Coupling . . . . .	93
4.3	Security . . . . .	95
4.3.1	Architectural Vulnerabilities . . . . .	96
4.3.2	User-Level Security . . . . .	97
4.4	Concurrency, Race Conditions, and Deadlocks . . . . .	100
4.5	Summary . . . . .	105
	Exercises . . . . .	105
<b>5</b>	<b>Working with Code</b>	<b>107</b>
5.1	Bad Smells and Metrics . . . . .	108
5.1.1	Identifying Bad Smells . . . . .	108
5.1.2	Software Metrics . . . . .	110
5.2	Refactoring . . . . .	111
5.2.1	Example 1: Removing Useless Functions . . . . .	114
5.2.2	Example 2: Removing a Layering Violation . . . . .	114
5.3	Testing . . . . .	117
5.3.1	Unit Testing Tools . . . . .	119

5.3.2	Test Case Design . . . . .	120
5.3.3	A Strategy for Sequencing Unit Tests . . . . .	128
5.4	Debugging . . . . .	129
5.4.1	Tool Use vs Developer Skill . . . . .	130
5.4.2	Example 1: A User Interface Bug . . . . .	131
5.4.3	Example 2: A Multi-Level Bug . . . . .	133
5.5	Extending the Software for a New Project . . . . .	134
5.5.1	A New Use Case . . . . .	135
5.5.2	Impact on the Code Base . . . . .	136
5.5.3	Team Discussions . . . . .	139
5.6	Summary . . . . .	140
	Exercises . . . . .	140
<b>6</b>	<b>Developing the Domain Classes</b>	<b>143</b>
6.1	Understanding the Current System . . . . .	143
6.1.1	Reading a Design Document . . . . .	144
6.1.2	Reading Code . . . . .	147
6.1.3	Examining the Domain Classes . . . . .	150
6.2	Adding New Features . . . . .	151
6.2.1	Top-Down Analysis/Bottom-Up Development . . . . .	154
6.2.2	Modifying the Domain Classes . . . . .	155
6.2.3	Documentation and Bulletproofing . . . . .	158
6.3	Class Design Principles and Practice . . . . .	162
6.3.1	Using What's Already There . . . . .	163
6.3.2	Adding a New Domain Class . . . . .	164
6.4	Managing the Ripple Effect . . . . .	166
6.4.1	Unit Testing the New Code . . . . .	166
6.4.2	Refactoring the New Code Base . . . . .	169
6.5	Summary . . . . .	171
	Exercises . . . . .	171
<b>7</b>	<b>Developing the Database Modules</b>	<b>173</b>
7.1	Design Principles and Practice . . . . .	174
7.1.1	Database Creation . . . . .	175
7.1.2	Connecting the Program to the Database . . . . .	176
7.1.3	Tables . . . . .	178
7.1.4	Normalization and Keys . . . . .	180
7.1.5	Backup and Recovery . . . . .	181
7.2	Working with a Database . . . . .	182
7.2.1	Table Creation . . . . .	184
7.2.2	Table Searching . . . . .	185
7.2.3	Table Insertion, Deletion, and Updating . . . . .	187
7.3	Database Security and Integrity . . . . .	188
7.3.1	Database-Level Permissions . . . . .	189
7.3.2	User-Level Permissions . . . . .	189

7.3.3	Controlling Concurrency . . . . .	192
7.4	Adding New Software Features: Database Impact . . . . .	193
7.4.1	Items 1 and 9d: Volunteer Status and Application . . . . .	195
7.4.2	Item 3: Calendar View . . . . .	198
7.5	Summary . . . . .	201
	Exercises . . . . .	202
<b>8</b>	<b>Developing the User Interface</b>	<b>205</b>
8.1	Design Principles and Practice . . . . .	205
8.1.1	The Model-View-Controller Pattern . . . . .	207
8.1.2	Sessions, Query Strings, and Global Variables . . . . .	210
8.1.3	Ensuring Security at the User Interface . . . . .	213
8.2	Working with Code . . . . .	218
8.2.1	Reading Deeply . . . . .	219
8.2.2	Debugging as a Community Activity . . . . .	224
8.3	Adding New Features: User Interface Impact . . . . .	230
8.3.1	Item 1: Volunteer Status . . . . .	230
8.3.2	Item 2: Make Active/Inactive . . . . .	235
8.3.3	Item 3: Calendar View . . . . .	237
8.4	Summary . . . . .	239
	Exercises . . . . .	240
<b>9</b>	<b>User Support</b>	<b>243</b>
9.1	Technical Writing . . . . .	243
9.1.1	Knowing Your Audience . . . . .	244
9.1.2	Principles of Good Writing . . . . .	246
9.2	Types of User Support . . . . .	249
9.2.1	On-Line Help . . . . .	249
9.2.2	Reference Manuals . . . . .	251
9.2.3	Open Discussion Forums . . . . .	253
9.2.4	User Training and Feedback . . . . .	257
9.3	Example: RMH Homebase On-Line Help . . . . .	258
9.3.1	Help and the Code Base . . . . .	258
9.4	Summary . . . . .	263
	Exercises . . . . .	263
<b>10</b>	<b>Project Governance</b>	<b>265</b>
10.1	Origins and Evolution . . . . .	265
10.1.1	Starting a Client-Oriented Project . . . . .	267
10.1.2	Quality Assessment . . . . .	271
10.2	Evolving into a Democratic Meritocracy . . . . .	273
10.2.1	Incubation . . . . .	274
10.2.2	Organization . . . . .	277
10.2.3	Decision Making and Conflict Resolution . . . . .	281
10.2.4	Domain Constraints . . . . .	282

10.3	Releasing Code . . . . .	284
10.3.1	Licensing . . . . .	284
10.3.2	Finding a Project Host . . . . .	285
10.3.3	Release Strategies . . . . .	287
10.4	Summary . . . . .	289
	Exercises . . . . .	290
<b>11</b>	<b>New Project Conception</b>	<b>291</b>
11.1	Requirements Gathering . . . . .	292
11.1.1	Domain Analysis . . . . .	292
11.1.2	User Stories . . . . .	295
11.1.3	Use Cases . . . . .	297
11.2	Initial Design . . . . .	303
11.2.1	Domain Classes . . . . .	303
11.2.2	User Interface . . . . .	304
11.2.3	Performance and Platform . . . . .	305
11.2.4	System Architecture . . . . .	307
11.2.5	Design Alternatives . . . . .	308
11.2.6	Design Document . . . . .	308
11.3	Summary . . . . .	309
	Exercises . . . . .	309
	<b>Appendices</b>	<b>311</b>
<b>A</b>	<b>Details of the Case Study</b>	<b>311</b>
A.1	Requirements . . . . .	311
A.1.1	Domain Analysis . . . . .	312
A.1.2	Use Cases . . . . .	317
A.1.3	System Requirements . . . . .	327
A.2	Design . . . . .	328
A.2.1	Goals . . . . .	329
A.2.2	Software Architecture . . . . .	329
A.2.3	Domain Classes . . . . .	330
A.2.4	Database Design . . . . .	330
A.2.5	GUI Design . . . . .	333
A.2.6	Implementation Schedule . . . . .	336
A.2.7	User-System Interaction . . . . .	336
<b>B</b>	<b>New Features for an Existing Code Base</b>	<b>341</b>
B.1	Starting with a Request from the Client . . . . .	341
B.2	Impact on the Design and the Code Base . . . . .	343
B.3	Defining a Project that Implements these Features . . . . .	350
	<b>References</b>	<b>351</b>
	<b>Index</b>	<b>355</b>