SOFTWARE PROJECT MANAGEMENT

A Unified Framework

WALKER ROYCE

RATIONAL SOFTWARE CORPORATION

ADDISON–WESLEY

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

Contents

List of Figures xiii
List of Tables xvii
Foreword xxi
Preface xxiii

PART I SOFTWARE MANAGEMENT RENAISSANCE 1

CHAPTER 1 **Conventional Software Management** 5
1.1 The Waterfall Model 6
1.1.1 In Theory 6
1.1.2 In Practice 11
1.2 Conventional Software Management Performance 17

CHAPTER 2 **Evolution of Software Economics** 21
2.1 Software Economics 21
2.2 Pragmatic Software Cost Estimation 26

CHAPTER 3 **Improving Software Economics** 31
3.1 Reducing Software Product Size 33
3.1.1 Languages 34
3.1.2 Object-Oriented Methods and Visual Modeling 36
3.1.3 Reuse 38
3.1.4 Commercial Components 39
3.2 Improving Software Processes 40
3.3 Improving Team Effectiveness 43

3.4 Improving Automation through Software Environments 46
3.5 Achieving Required Quality 48
3.6 Peer Inspections: A Pragmatic View 51

CHAPTER 4 **The Old Way and the New** 55
4.1 The Principles of Conventional Software Engineering 55
4.2 The Principles of Modern Software Management 63
4.3 Transitioning to an Iterative Process 66

PART II **A SOFTWARE MANAGEMENT PROCESS FRAMEWORK** 69

CHAPTER 5 **Life-Cycle Phases** 73
5.1 Engineering and Production Stages 74
5.2 Inception Phase 76
5.3 Elaboration Phase 77
5.4 Construction Phase 79
5.5 Transition Phase 80

CHAPTER 6 **Artifacts of the Process** 83
6.1 The Artifact Sets 84
6.1.1 The Management Set 85
6.1.2 The Engineering Sets 86
6.1.3 Artifact Evolution over the Life Cycle 92
6.1.4 Test Artifacts 93
6.2 Management Artifacts 96
6.3 Engineering Artifacts 103
6.4 Pragmatic Artifacts 105

CHAPTER 7 **Model-Based Software Architectures** 109
7.1 Architecture: A Management Perspective 110
7.2 Architecture: A Technical Perspective 111

CHAPTER 8 **Workflows of the Process** 117
8.1 Software Process Workflows 118
8.2 Iteration Workflows 121

CHAPTER 9 **Checkpoints of the Process** 125
9.1 Major Milestones 126
9.2 Minor Milestones 132
9.3 Periodic Status Assessments 133

PART III SOFTWARE MANAGEMENT DISCIPLINES 135

CHAPTER 10 Iterative Process Planning 139
10.1 Work Breakdown Structures 139
10.1.1 Conventional WBS Issues 140
10.1.2 Evolutionary Work Breakdown Structures 142
10.2 Planning Guidelines 146
10.3 The Cost and Schedule Estimating Process 149
10.4 The Iteration Planning Process 150
10.5 Pragmatic Planning 153

CHAPTER 11 Project Organizations and Responsibilities 155
11.1 Line-of-Business Organizations 156
11.2 Project Organizations 158
11.3 Evolution of Organizations 165

CHAPTER 12 Process Automation 167
12.1 Tools: Automation Building Blocks 168
12.2 The Project Environment 172
12.2.1 Round-Trip Engineering 173
12.2.2 Change Management 174
12.2.3 Infrastructures 181
12.2.4 Stakeholder Environments 184

CHAPTER 13 Project Control and Process Instrumentation 187
13.1 The Seven Core Metrics 188
13.2 Management Indicators 190
13.2.1 Work and Progress 190
13.2.2 Budgeted Cost and Expenditures 191
13.2.3 Staffing and Team Dynamics 195
13.3 Quality Indicators 196
13.3.1 Change Traffic and Stability 196
13.3.2 Breakage and Modularity 197
13.3.3 Rework and Adaptability 197
13.3.4 MTBF and Maturity 198
13.4 Life-Cycle Expectations 199
13.5 Pragmatic Software Metrics 201
13.6 Metrics Automation 202

CHAPTER 14 **Tailoring the Process** 209
14.1 Process Discriminants 209
14.1.1 Scale 210
14.1.2 Stakeholder Cohesion or Contention 214
14.1.3 Process Flexibility or Rigor 215
14.1.4 Process Maturity 215
14.1.5 Architectural Risk 217
14.1.6 Domain Experience 217
14.2 Example: Small-Scale Project versus Large-Scale Project 218

PART IV **LOOKING FORWARD** 221

CHAPTER 15 **Modern Project Profiles** 225
15.1 Continuous Integration 226
15.2 Early Risk Resolution 227
15.3 Evolutionary Requirements 228
15.4 Teamwork among Stakeholders 229
15.5 Top 10 Software Management Principles 231
15.6 Software Management Best Practices 232

CHAPTER 16 **Next-Generation Software Economics** 237
16.1 Next-Generation Cost Models 237
16.2 Modern Software Economics 242

CHAPTER 17 **Modern Process Transitions** 247
17.1 Culture Shifts 248
17.2 Denouement 251

PART V **CASE STUDIES AND BACKUP MATERIAL** 255

APPENDIX A **The State of the Practice in Software Management** 259

APPENDIX B **The COCOMO Cost Estimation Model** 265
B.1 COCOMO 266
B.2 Ada COCOMO 269
B.3 COCOMO II 274

APPENDIX C **Change Metrics** 283

C.1 Overview 284

C.2 Metrics Derivation 286

C.2.1 Collected Statistics 288

C.2.2 End-Product Quality Metrics 291

C.2.3 In-Progress Indicators 293

C.3 Pragmatic Change Metrics 297

APPENDIX D **CCPDS-R Case Study** 299

D.1 Context for the Case Study 300

D.2 Common Subsystem Overview 301

D.3 Project Organization 304

D.4 Common Subsystem Product Overview 305

D.5 Process Overview 310

D.5.1 Risk Management: Build Content 312

D.5.2 The Incremental Design Process 315

D.5.3 Component Evolution 318

D.5.4 The Incremental Test Process 321

D.5.5 DOD-STD-2167A Artifacts 323

D.6 Demonstration-Based Assessment 326

D.7 Core Metrics 337

D.7.1 Development Progress 338

D.7.2 Test Progress 340

D.7.3 Stability 343

D.7.4 Modularity 343

D.7.5 Adaptability 344

D.7.6 Maturity 345

D.7.7 Cost/Effort Expenditures by Activity 345

D.8 Other Metrics 348

D.8.1 Software Size Evolution 348

D.8.2 Subsystem Process Improvements 352

D.8.3 SCO Resolution Profile 353

D.8.4 CSCI Productivities and Quality Factors 354

D.9 People Factors 356

D.9.1 Core Team 357

D.9.2 Award Fee Flowdown Plan 358

D.10 Conclusions 359

APPENDIX E **Process Improvement and Mapping to the CMM** 363
E.1 CMM Overview 363
E.2 Pragmatic Process Improvement 366
E.3 Maturity Questionnaire 367
E.4 Questions Not Asked by the Maturity Questionnaire 387
E.5 Overall Process Assessment 390

Glossary 391
References 397
Index 401

List of Figures

FIGURE 1-1 The waterfall model 7

FIGURE 1-2 Progress profile of a conventional software project 12

FIGURE 1-3 Risk profile of a conventional software project across its life cycle 14

FIGURE 1-4 Suboptimal software component organization resulting from a requirements-driven approach 16

FIGURE 2-1 Three generations of software economics leading to the target objective 23

FIGURE 2-2 Return on investment in different domains 25

FIGURE 2-3 The predominant cost estimation process 28

FIGURE 3-1 Cost and schedule investments necessary to achieve reusable components 39

FIGURE 4-1 The top five principles of a modern process 64

FIGURE 5-1 The phases of the life-cycle process 75

FIGURE 6-1 Overview of the artifact sets 85

FIGURE 6-2 Life-cycle focus on artifact sets 89

FIGURE 6-3 Life-cycle evolution of the artifact sets 92

FIGURE 6-4 Typical business case outline 97

FIGURE 6-5 Typical release specification outline 97

FIGURE 6-6 Typical software development plan outline 99

FIGURE 6-7 Typical release description outline 100

FIGURE 6-8 Artifact sequences across a typical life cycle 102

FIGURE 6-9 Typical vision document outline 103

FIGURE 6-10 Typical architecture description outline 105

FIGURE 7-1 Architecture, an organized and abstracted view into the design models 113
FIGURE 8-1 Activity levels across the life-cycle phases 119
FIGURE 8-2 The workflow of an iteration 121
FIGURE 8-3 Iteration emphasis across the life cycle 123
FIGURE 8-4 A typical build sequence associated with a layered architecture 124
FIGURE 9-1 A typical sequence of life-cycle checkpoints 127
FIGURE 9-2 Engineering artifacts available at the life-cycle architecture milestone .. 130
FIGURE 9-3 Default agendas for the life-cycle architecture milestone 131
FIGURE 9-4 Typical minor milestones in the life cycle of an iteration 133
FIGURE 10-1 Conventional work breakdown structure, following the product hierarchy 141
FIGURE 10-2 Default work breakdown structure 144
FIGURE 10-3 Evolution of planning fidelity in the WBS over the life cycle 147
FIGURE 10-4 Planning balance throughout the life cycle 151
FIGURE 11-1 Default roles in a software line-of-business organization 156
FIGURE 11-2 Default project organization and responsibilities 159
FIGURE 11-3 Software management team activities 160
FIGURE 11-4 Software architecture team activities 161
FIGURE 11-5 Software development team activities 162
FIGURE 11-6 Software assessment team activities 164
FIGURE 11-7 Software project team evolution over the life cycle 165
FIGURE 12-1 Typical automation and tool components that support the process workflows 169
FIGURE 12-2 Round-trip engineering 174
FIGURE 12-3 The primitive components of a software change order 176
FIGURE 12-4 Example release histories for a typical project and a typical product 179
FIGURE 12-5 Organization policy outline 183
FIGURE 12-6 Extending environments into stakeholder domains 185
FIGURE 13-1 Expected progress for a typical project with three major releases 191
FIGURE 13-2 The basic parameters of an earned value system 193
FIGURE 13-3 Assessment of book progress (example) 194
FIGURE 13-4 Typical staffing profile 196
FIGURE 13-5 Stability expectation over a healthy project's life cycle 197
FIGURE 13-6 Modularity expectation over a healthy project's life cycle 197
FIGURE 13-7 Adaptability expectation over a healthy project's life cycle 198

FIGURE 13-8 Maturity expectation over a healthy project's life cycle........................ 198
FIGURE 13-9 Examples of the fundamental metrics classes........................ 205
FIGURE 13-10 Example SPCP display for a top-level project situation........................ 206
FIGURE 14-1 The two primary dimensions of process variability........................ 210
FIGURE 14-2 Priorities for tailoring the process framework........................ 211
FIGURE 15-1 Progress profile of a modern project........................ 226
FIGURE 15-2 Risk profile of a typical modern project across its life cycle 229
FIGURE 15-3 Organization of software components resulting from a modern process........................ 230
FIGURE 15-4 Balanced application of modern principles to achieve economic results........................ 233
FIGURE 16-1 Next-generation cost models........................ 239
FIGURE 16-2 Differentiating potential solutions through cost estimation 240
FIGURE 16-3 Automation of the construction process in next-generation environments 242
FIGURE 17-1 Next-generation project performance........................ 252
FIGURE B-1 Profile of a conventional project........................ 270
FIGURE B-2 Software estimation over a project life cycle........................ 276
FIGURE B-3 COCOMO II estimation over a project life cycle........................ 276
FIGURE C-1 Expected trends for in-progress indicators........................ 294
FIGURE C-2 Expectations for quality trends........................ 295
FIGURE D-1 CCPDS-R life-cycle overview 302
FIGURE D-2 Full-scale development phase project organization........................ 306
FIGURE D-3 Common Subsystem SAS evolution........................ 309
FIGURE D-4 Overview of the CCPDS-R macroprocess, milestones, and schedule..... 311
FIGURE D-5 Common Subsystem builds 313
FIGURE D-6 Basic activities sequence for an individual build 315
FIGURE D-7 Incremental baseline evolution and test activity flow 322
FIGURE D-8 CCPDS-R first demonstration activities and schedule........................ 330
FIGURE D-9 Development progress summary 339
FIGURE D-10 Common Subsystem development progress........................ 340
FIGURE D-11 Common Subsystem test progress 342
FIGURE D-12 Common Subsystem stability 343
FIGURE D-13 Common Subsystem modularity 344
FIGURE D-14 Common Subsystem adaptability........................ 345
FIGURE D-15 Common Subsystem maturity 346
FIGURE D-16 Common Subsystem SCO change profile........................ 355
FIGURE E-1 Project performance expectations for CMM maturity levels........................ 365

List of Tables

TABLE 1-1 Expenditures by activity for a conventional software project 13

TABLE 1-2 Results of conventional software project design reviews 17

TABLE 3-1 Important trends in improving software economics 32

TABLE 3-2 Language expressiveness of some of today's popular languages 34

TABLE 3-3 Advantages and disadvantages of commercial components versus custom software 40

TABLE 3-4 Three levels of process and their attributes 41

TABLE 3-5 General quality improvements with a modern process 49

TABLE 4-1 Modern process approaches for solving conventional problems 66

TABLE 5-1 The two stages of the life cycle: engineering and production 74

TABLE 8-1 The artifacts and life-cycle emphases associated with each workflow ... 120

TABLE 9-1 The general status of plans, requirements, and products across the major milestones 128

TABLE 9-2 Default content of status assessment reviews 134

TABLE 10-1 WBS budgeting defaults 148

TABLE 10-2 Default distributions of effort and schedule by phase 148

TABLE 12-1 Representative examples of changes at opposite ends of the project spectrum 180

TABLE 13-1 Overview of the seven core metrics 189

TABLE 13-2 Measurement of actual progress of book development (example) 194

TABLE 13-3 The default pattern of life-cycle metrics evolution 200

TABLE 14-1 Process discriminators that result from differences in project size 213

TABLE 14-2 Process discriminators that result from differences in stakeholder cohesion 214

TABLE 14-3 Process discriminators that result from differences in process flexibility 216

TABLE 14-4 Process discriminators that result from differences in process maturity 216

TABLE 14-5 Process discriminators that result from differences in architectural risk 217

TABLE 14-6 Process discriminators that result from differences in domain experience 218

TABLE 14-7 Schedule distribution across phases for small and large projects 218

TABLE 14-8 Differences in workflow priorities between small and large projects 219

TABLE 14-9 Differences in artifacts between small and large projects 220

TABLE 15-1 Differences in workflow cost allocations between a conventional process and a modern process 227

TABLE 15-2. Results of major milestones in a modern process 231

TABLE A-1 Technologies used on software projects 260

TABLE A-2 Social factors observed on software projects 261

TABLE A-3 Factors that affect the success of software projects 262

TABLE B-1 COCOMO project characterization parameters 267

TABLE B-2 Effort and schedule partition across conventional life-cycle phases 268

TABLE B-3 Default effort allocations across COCOMO WBS activities 269

TABLE B-4 Ada COCOMO improvements to the effort adjustment factors 272

TABLE B-5 Early design model effort adjustment factors 277

TABLE B-6 COCOMO II post-architecture model updates to Ada COCOMO and COCOMO 278

TABLE B-7 COCOMO II process exponent parameters 281

TABLE C-1 Definitions of collected statistics 288

TABLE C-2 End-product quality metrics 291

TABLE C-3 Definitions of in-progress indicators 293

TABLE D-1 CSCI summary 307

TABLE D-2 A typical component evolution from creation through turnover 319

TABLE D-3 NAS CSCI metrics summary at month 10 320

TABLE D-4 CCPDS-R software artifacts 325

TABLE D-5 Software development file evolution 326

TABLE D-6 SCO characteristics for build 2 BIT testing 341

TABLE D-7 Requirements verification work by test type and CSCI 342

TABLE D-8 Common Subsystem cost expenditures by top-level WBS element 346
TABLE D-9 Common Subsystem lower level WBS elements 347
TABLE D-10 Common Subsystem CSCI sizes.. 349
TABLE D-11 SLOC-to-ESLOC conversion factors 350
TABLE D-12 Common Subsystem CSCI sizes in ESLOC 352
TABLE D-13 CCPDS-R subsystem changes by CSCI 354
TABLE D-14 Common Subsystem CSCI summary 355
TABLE D-15 CCPDS-R technology improvements.. 360
TABLE E-1 Industry distribution across maturity levels 364

Foreword

This book blazes the way toward the next generation of software management practice. Many organizations still cling to the waterfall model because, even with its shortfalls, it provides the most fully elaborated management guidelines on how to proceed in a given software situation.

It has been difficult to find a fully articulated alternative management approach for dealing with such issues as commercial component integration, software reuse, risk management, and evolutionary/incremental/spiral software processes. This book provides a new experience-tested framework and set of guidelines on how to proceed.

Walker Royce developed and tested this software management approach during his inception-to-delivery participation in the large, successful CCPDS-R project performed by TRW for the U.S. Air Force. He then refined and generalized it across a wide spectrum of government, aerospace, and commercial software development experiences at Rational.

Chapters 1 through 4 of the book motivate the approach by showing how it gives you management control of the key software economics leverage points with respect to traditional software management. These are (1) reducing the amount of software you need to build, (2) reducing rework via improved processes and teamwork, and (3) reducing the labor-intensiveness of the remaining work via automation.

Chapters 5 through 10 present the specifics of a new organization of the software life cycle, which also forms the management basis for Rational's Unified process. It combines the flexibility of the spiral model with the discipline of risk management and a set of major life-cycle phases and milestones. These milestones are focused on major management commitments to life-cycle courses of action.

As with our Anchor Point approach at USC, the life-cycle objectives milestone involves a management commitment to engage in a software architecting effort based on a business case analysis (or not to engage, in which case the project is mercifully killed). The life-cycle architecture milestone involves a management commitment to proceed into full-scale development based on establishing and demonstrating a sound architecture and resolving all major risk items. The initial operational capability milestone involves a management commitment to proceed to beta testing the product with outside users, or its equivalent.

In these chapters, Royce provides a set of views showing how these milestones differ from conventional document-oriented or code-oriented milestones. Instead, the key product artifact sets (requirements, design, implementation, deployment) concurrently evolve and coalesce in a manner consistent with the project's objectives and its strategies for controlling risk.

In Chapters 10 through 14, Royce addresses how to ensure that the software project's management artifacts are also concurrently evolving and coalescing. These include the project's plans and associated cost and schedule estimates, the project's organization and team-building activities, and the project's metrics, instrumentation, and control processes. Chapter 14 is particularly noteworthy. It not only emphasizes that the management solutions are situation-dependent, it also provides guidelines for tailoring them to the project's scale, team culture, process maturity, architectural risk, and domain experience.

In Chapters 15 through 17, Royce looks forward to where the best software developers are going with their practices: toward product line management, round-trip engineering, and smaller teams with managers as performers and quality assurance as everyone's job. Appendixes relate his software management approach to the current state of the practice, to the COCOMO and COCOMO II family of cost models, and to the SEI Capability Maturity Model. Appendix D provides a convincing case study of how the approach was successfully used on the large, technically challenging CCPDS-R project.

Royce has a refreshing candor about some of the fads, follies, and excesses in the software field. This comes out particularly in several "pragmatic" sections that address such topics as software cost estimation, inspections, artifacts, planning, and metrics. Not everyone will agree with all of his assessments, particularly on inspections, but they are incisive and thought-provoking.

I feel extremely fortunate to have been able to work with both Walker Royce and his equally insightful father, Winston Royce; to have learned from their experiences; and to have interacted with them as they evolved their path-breaking ideas.

Barry Boehm
Director, USC Center for Software Engineering
April 1998