Contents

Preface to the Third Edition **ix**

1 Introduction **1**
1.1 The Problem Domain 2
1.1.1 Industrial Strength Software 2
1.1.2 Software is Expensive 4
1.1.3 Late and Unreliable 5
1.1.4 Maintenance and Rework 6
1.2 The Software Engineering Challenges 8
1.2.1 Scale 9
1.2.2 Quality and Productivity 11
1.2.3 Consistency and Repeatability 14
1.2.4 Change 15
1.3 The Software Engineering Approach 15
1.3.1 Phased Development Process 16
1.3.2 Managing the Process 20
1.4 Summary 21
Exercises 22

2 Software Processes **25**
2.1 Software Process 25
2.1.1 Processes and Process Models 26
2.1.2 Component Software Processes 27
2.1.3 ETVX Approach for Process Specification 29
2.2 Desired Characteristics of Software Process 31
2.2.1 Predictability 31
2.2.2 Support Testability and Maintainability 33
2.2.3 Support Change 34
2.2.4 Early Defect Removal 35

2.2.5 Process Improvement and Feedback 36
2.3 Software Development Process Models 37
2.3.1 Waterfall Model 37
2.3.2 Prototyping 41
2.3.3 Iterative Development 43
2.3.4 Timeboxing Model 46
2.3.5 Comparision of Models 50
2.4 Other Software Processes 50
2.4.1 Project Management Process 52
2.4.2 The Inspection Process 54
2.4.3 Software Configuration Management Process 61
2.4.4 Requirements Change Management Process 67
2.4.5 Process Management Process 69
2.5 Summary 73
Exercises 75

3 Software Requirements Analysis and Specification 79
3.1 Software Requirements 80
3.1.1 Need for SRS 81
3.1.2 Requirement Process 85
3.2 Problem Analysis 89
3.2.1 Informal Approach 90
3.2.2 Data Flow Modeling 91
3.2.3 Object-Oriented Modeling 103
3.2.4 Prototyping 113
3.3 Requirements Specification 117
3.3.1 Characteristics of an SRS 118
3.3.2 Components of an SRS 120
3.3.3 Specification Language 124
3.3.4 Structure of a Requirements Document 125
3.4 Functional Specification with Use Cases 128
3.4.1 Basics 129
3.4.2 Examples 132
3.4.3 Extensions 135
3.4.4 Developing Use Cases 136
3.5 Validation 138
3.6 Metrics 142
3.6.1 Size—Function Points 142
3.6.2 Quality Metrics 147
3.7 Summary 148
Exercises 150

Case Studies 152

4 Software Architecture 159
4.1 Role of Software Architecture 160
4.2 Architecture Views 163
4.3 Component and Connector View 167
4.3.1 Components 167
4.3.2 Connectors 169
4.3.3 An Example 172
4.4 Architecture Styles for C&C View 176
4.4.1 Pipe and Filter 176
4.4.2 Shared-Data Style 178
4.4.3 Client-Server Style 181
4.4.4 Some Other Styles 182
4.5 Discussion 183
4.5.1 Architecture and Design 183
4.5.2 Preserving the Integrity of an Architecture 184
4.5.3 Deployment View and Performance Analysis 188
4.5.4 Documenting Architecture Design 190
4.6 Evaluating Architectures 194
4.6.1 The ATAM Analysis Method 195
4.6.2 An Example 196
4.7 Summary 199
Exercises 201
Case Studies 203

5 Planning a Software Project 207
5.1 Process Planning 208
5.2 Effort Estimation 208
5.2.1 Uncertainties in Effort Estimation 209
5.2.2 Building Effort Estimation Models 211
5.2.3 A Bottom-Up Estimation Approach 213
5.2.4 COCOMO Model 215
5.3 Project Scheduling and Staffing 219
5.3.1 Overall Scheduling 219
5.3.2 Detailed Scheduling 221
5.3.3 An Example 223
5.3.4 Team Structure 224
5.4 Software Configuration Management Plan 225
5.5 Quality Plan 226
5.5.1 Defect Injection and Removal Cycle 227

5.5.2 Approaches to Quality Management 228
5.5.3 Quality Plan 229
5.6 Risk Management 230
5.6.1 Risk Management Concepts 230
5.6.2 Risk Assessment 232
5.6.3 Risk Control 236
5.6.4 A Practical Risk Management Approach 237
5.7 Project Monitoring Plan 237
5.7.1 Measurements 239
5.7.2 Project Monitoring and Tracking 239
5.8 Summary 241
Exercises 243
Case Studies 245

6 Function-Oriented Design 247
6.1 Design Principles 248
6.1.1 Problem Partitioning and Hierarchy 250
6.1.2 Abstraction 251
6.1.3 Modularity 253
6.1.4 Top-Down and Bottom-Up Strategies 254
6.2 Module-Level Concepts 255
6.2.1 Coupling 255
6.2.2 Cohesion 257
6.3 Design Notation and Specification 260
6.3.1 Structure Charts 261
6.3.2 Specification 265
6.4 Structured Design Methodology 266
6.4.1 Restate the Problem as a Data Flow Diagram 267
6.4.2 Identify the Most Abstract Input and Output Data Elements 269
6.4.3 First-Level Factoring 271
6.4.4 Factoring the Input, Output, and Transform Branches 273
6.4.5 Design Heuristics 276
6.4.6 Transaction Analysis 277
6.4.7 Discussion 279
6.5 Verification 281
6.6 Metrics 283
6.6.1 Network Metrics 284
6.6.2 Stability Metrics 285
6.6.3 Information Flow Metrics 288
6.7 Summary 290

Exercises . . . 292
Case Studies . . . 294

7 Object-Oriented Design 303
7.1 OO Analysis and OO Design . . . 304
7.2 OO Concepts . . . 306
7.2.1 Classes and Objects . . . 307
7.2.2 Relationships Among Objects . . . 312
7.2.3 Inheritance and Polymorphism . . . 315
7.3 Design Concepts . . . 323
7.3.1 Coupling . . . 323
7.3.2 Cohesion . . . 325
7.3.3 The Open-Closed Principle . . . 327
7.3.4 Some Design Guidelines . . . 329
7.4 Unified Modeling Language (UML) . . . 331
7.4.1 Class Diagram . . . 331
7.4.2 Sequence and Collaboration Diagrams . . . 335
7.4.3 Other Diagrams and Capabilities . . . 339
7.5 A Design Methodology . . . 341
7.5.1 Dynamic Modeling . . . 343
7.5.2 Functional Modeling . . . 345
7.5.3 Defining Internal Classes and Operations . . . 346
7.5.4 Optimize and Package . . . 347
7.5.5 Examples . . . 348
7.6 Metrics . . . 356
7.7 Summary . . . 360
Exercises . . . 362
Case Studies . . . 364

8 Detailed Design 371
8.1 Detailed Design and PDL . . . 371
8.1.1 PDL . . . 371
8.1.2 Logic/Algorithm Design . . . 374
8.1.3 State Modeling of Classes . . . 378
8.2 Verification . . . 380
8.2.1 Design Walkthroughs . . . 380
8.2.2 Critical Design Review . . . 381
8.2.3 Consistency Checkers . . . 382
8.3 Metrics . . . 383
8.3.1 Cyclomatic Complexity . . . 383
8.3.2 Data Bindings . . . 386

8.3.3 Cohesion Metric 387
8.4 Summary 388
Exercises 389

9 Coding **391**
9.1 Programming Principles and Guidelines 392
9.1.1 Common Coding Errors 393
9.1.2 Structured Programming 398
9.1.3 Information Hiding 401
9.1.4 Some Programming Practices 402
9.1.5 Coding Standards 406
9.2 Coding Process 409
9.2.1 An Incremental Coding Process 410
9.2.2 Test Driven Development 411
9.2.3 Pair Programming 413
9.2.4 Source Code Control and Build 414
9.3 Refactoring 416
9.3.1 Basic Concepts 417
9.3.2 An example 419
9.3.3 Bad Smells 422
9.3.4 Common Refactorings 424
9.4 Verification 429
9.4.1 Code Inspections 429
9.4.2 Static Analysis 431
9.4.3 Proving Correctness 437
9.4.4 Unit Testing 444
9.4.5 Combining Different Techniques 449
9.5 Metrics 451
9.5.1 Size Measures 452
9.5.2 Complexity Metrics 453
9.6 Summary 456
Exercises 458
Case Studies 462

10 Testing **465**
10.1 Testing Fundamentals 466
10.1.1 Error, Fault, and Failure 466
10.1.2 Test Oracles 468
10.1.3 Test Cases and Test Criteria 469
10.1.4 Psychology of Testing 471
10.2 Black-Box Testing 472

10.2.1 Equivalence Class Partitioning 473
10.2.2 Boundary Value Analysis 475
10.2.3 Cause-Effect Graphing 477
10.2.4 Pair-wise Testing 480
10.2.5 Special Cases 483
10.2.6 State-Based Testing 484
10.3 White-Box Testing 487
10.3.1 Control Flow-Based Criteria 488
10.3.2 Data Flow-Based Testing 491
10.3.3 An Example 495
10.3.4 Mutation Testing 498
10.3.5 Test Case Generation and Tool Support 502
10.4 Testing Process 504
10.4.1 Levels of Testing 505
10.4.2 Test Plan 507
10.4.3 Test Case Specifications 509
10.4.4 Test Case Execution and Analysis 511
10.4.5 Defect Logging and Tracking 513
10.5 Defect Analysis and Prevention 516
10.5.1 Pareto Analysis 517
10.5.2 Perform Causal Analysis 517
10.5.3 Develop and Implement Solutions 520
10.6 Metrics—Reliability Estimation 521
10.6.1 Basic Concepts and Definitions 522
10.6.2 A Reliability Model 524
10.6.3 Failure Data and Parameter Estimation 529
10.6.4 Translating to Calendar Time 532
10.6.5 An Example 532
10.7 Summary 534
Exercises 536
Case Studies 539

Bibliography **543**

Index **553**