Contents at a Glance

CHAPTER 1 Introduction to Software Engineering 33

PART ONE **The Software Process** 51

CHAPTER 2 A Generic View of Process 52
CHAPTER 3 Process Models 77
CHAPTER 4 An Agile View of Process 103

PART TWO **Software Engineering Practice** 127

CHAPTER 5 Software Engineering Practice 128
CHAPTER 6 System Engineering 154
CHAPTER 7 Requirements Engineering 174
CHAPTER 8 Building the Analysis Model 207
CHAPTER 9 Design Engineering 258
CHAPTER 10 Creating an Architectural Design 286
CHAPTER 11 Modeling Component-Level Design 324
CHAPTER 12 Performing User Interface Design 356
CHAPTER 13 Testing Strategies 386
CHAPTER 14 Testing Tactics 420
CHAPTER 15 Product Metrics 461

PART THREE **Applying Web Engineering** 499

CHAPTER 16 Web Engineering 500
CHAPTER 17 Initiating a WebApp Project 514
CHAPTER 18 Analysis for WebApps 539
CHAPTER 19 Design for WebApps 559
CHAPTER 20 Testing for WebApps 594

PART FOUR **Managing Software Projects** 627

CHAPTER 21 Project Management 628
CHAPTER 22 Metrics for Process and Project 649
CHAPTER 23 Estimation 674
CHAPTER 24 Project Scheduling 705
CHAPTER 25 Risk Management 726

CHAPTER 26 Quality Management 744
CHAPTER 27 Change Management 771

PART FIVE **Advanced Topics in Software Engineering** 801

CHAPTER 28 Formal Methods 802
CHAPTER 29 Cleanroom Software Engineering 828
CHAPTER 30 Component-Based Development 847
CHAPTER 31 Reengineering 869
CHAPTER 32 The Road Ahead 892

TABLE OF CONTENTS

Preface 25

Walkthrough 29

CHAPTER 1 INTRODUCTION TO SOFTWARE ENGINEERING 33

1.1 The Evolving Role of Software 34
1.2 Software 36
1.3 The Changing Nature of Software 40
1.4 Legacy Software 42
  1.4.1 The Quality of Legacy Software 43
  1.4.2 Software Evolution 43
1.5 Software Myths 45
1.6 How It All Starts 47
1.7 Summary 48
REFERENCES 49
PROBLEMS AND POINTS TO PONDER 49
FURTHER READINGS AND INFORMATION SOURCES 50

PART ONE—THE SOFTWARE PROCESS 51

CHAPTER 2 A GENERIC VIEW OF PROCESS 52

2.1 Software Engineering—A Layered Technology 53
2.2 A Process Framework 54
2.3 The Capability Maturity Model Integration (CMMI) 59
2.4 Process Patterns 63
2.5 Process Assessment 66
2.6 Personal and Team Process Models 68
  2.6.1 Personal Software Process (PSP) 68
  2.6.2 Team Software Process (TSP) 70
2.7 Process Technology 71
2.8 Product and Process 72
2.9 Summary 73
REFERENCES 74
PROBLEMS AND POINTS TO PONDER 75
FURTHER READINGS AND INFORMATION SOURCES 75

✓ **CHAPTER 3 PROCESS MODELS 77**

3.1 Prescriptive Models 78
3.2 The Waterfall Model 79
3.3 Incremental Process Models 80
  3.3.1 The Incremental Model 80
  3.3.2 The RAD Model 81
3.4 Evolutionary Process Models 83
  3.4.1 Prototyping 83
  3.4.2 The Spiral Model 86

3.4.3 The Concurrent Development Model 88
3.4.4 A Final Comment on Evolutionary Processes 89
3.5 Specialized Process Models 91
3.5.1 Component-Based Development 91
3.5.2 The Formal Methods Model 92
3.5.3 Aspect-Oriented Software Development 93
3.6 The Unified Process 94
3.6.1 A Brief History 95
3.6.2 Phases of the Unified Process 96
3.6.3 Unified Process Work Products 98
3.7 Summary 99
REFERENCES 100
PROBLEMS AND POINTS TO PONDER 101
FURTHER READINGS AND INFORMATION SOURCES 102

CHAPTER 4 AN AGILE VIEW OF PROCESS 103

4.1 What Is Agility? 105
4.2 What Is an Agile Process? 106
4.2.1 The Politics of Agile Development 107
4.2.2 Human Factors 108
4.3 Agile Process Models 109
4.3.1 Extreme Programming (XP) 110
4.3.2 Adaptive Software Development (ASD) 114
4.3.3 Dynamic Systems Development Method (DSDM) 116
4.3.4 Scrum 117
4.3.5 Crystal 119
4.3.6 Feature Driven Development (FDD) 120
4.3.7 Agile Modeling (AM) 121
4.4 Summary 123
REFERENCES 124
PROBLEMS AND POINTS TO PONDER 125
FURTHER READINGS AND INFORMATION SOURCES 126

PART TWO—SOFTWARE ENGINEERING PRACTICE 127

CHAPTER 5 SOFTWARE ENGINEERING PRACTICE 128

5.1 Software Engineering Practice 129
5.1.1 The Essence of Practice 129
5.1.2 Core Principles 131
5.2 Communication Practices 133
5.3 Planning Practices 136
5.4 Modeling Practices 139
5.4.1 Analysis Modeling Principles 140
5.4.2 Design Modeling Principles 141
5.5 Construction Practice 144
5.5.1 Coding Principles and Concepts 145
5.5.2 Testing Principles 146
5.6 Deployment 148
5.7 Summary 150

REFERENCES 151
PROBLEMS AND POINTS TO PONDER 152
FURTHER READINGS AND INFORMATION SOURCES 152

CHAPTER 6 SYSTEM ENGINEERING 154

6.1 Computer-Based Systems 155
6.2 The System Engineering Hierarchy 157
6.2.1 System Modeling 158
6.2.2 System Simulation 160
6.3 Business Process Engineering: An Overview 161
6.4 Product Engineering: An Overview 162
6.5 System Modeling 164
6.5.1 Hatley-Pirbhai Modeling 165
6.5.2 System Modeling with UML 167
6.6 Summary 171
REFERENCES 172
PROBLEMS AND POINTS TO PONDER 172
FURTHER READINGS AND INFORMATION SOURCES 173

CHAPTER 7 REQUIREMENTS ENGINEERING 174

7.1 A Bridge to Design and Construction 175
7.2 Requirements Engineering Tasks 176
7.2.1 Inception 176
7.2.2 Elicitation 177
7.2.3 Elaboration 177
7.2.4 Negotiation 178
7.2.5 Specification 179
7.2.6 Validation 179
7.2.7 Requirements Management 180
7.3 Initiating the Requirements Engineering Process 181
7.3.1 Identifying the Stakeholders 182
7.3.2 Recognizing Multiple Viewpoints 182
7.3.3 Working toward Collaboration 183
7.3.4 Asking the First Questions 183
7.4 Eliciting Requirements 184
7.4.1 Collaborative Requirements Gathering 185
7.4.2 Quality Function Deployment 188
7.4.3 User Scenarios 189
7.4.4 Elicitation Work Products 190
7.5 Developing Use-Cases 191
7.6 Building the Analysis Model 196
7.6.1 Elements of the Analysis Model 196
7.6.2 Analysis Patterns 200
7.7 Negotiating Requirements 201
7.8 Validating Requirements 203
7.9 Summary 204
REFERENCES 204
PROBLEMS AND POINTS TO PONDER 205
FURTHER READINGS AND INFORMATION SOURCES 206

CHAPTER 8 BUILDING THE ANALYSIS MODEL 207

8.1 Requirements Analysis 208
- 8.1.1 Overall Objective and Philosophy 209
- 8.1.2 Analysis Rules of Thumb 210
- 8.1.3 Domain Analysis 210

8.2 Analysis Modeling Approaches 211
8.3 Data Modeling Concepts 213
- 8.3.1 Data Objects 213
- 8.3.2 Data Attributes 214
- 8.3.3 Relationships 214
- 8.3.4 Cardinality and Modality 215

8.4 Object-Oriented Analysis 217
8.5 Scenario-Based Modeling 218
- 8.5.1 Writing Use-Cases 218
- 8.5.2 Developing an Activity Diagram 223
- 8.5.3 Swimlane Diagrams 224

8.6 Flow-Oriented Modeling 226
- 8.6.1 Creating a Data Flow Model 226
- 8.6.2 Creating a Control Flow Model 229
- 8.6.3 The Control Specification 230
- 8.6.4 The Process Specification 232

8.7 Class-Based Modeling 233
- 8.7.1 Identifying Analysis Classes 233
- 8.7.2 Specifying Attributes 236
- 8.7.3 Defining Operations 237
- 8.7.4 Class-Responsibility-Collaborator (CRC) Modeling 240
- 8.7.5 Associations and Dependencies 246
- 8.7.6 Analysis Packages 247

8.8 Creating a Behavioral Model 248
- 8.8.1 Identifying Events with the Use-Case 249
- 8.8.2 State Representations 250

8.9 Summary 253

REFERENCES 254

PROBLEMS AND POINTS TO PONDER 255

FURTHER READINGS AND INFORMATION SOURCES 256

CHAPTER 9 DESIGN ENGINEERING 258

9.1 Design within the Context of Software Engineering 259
9.2 Design Process and Design Quality 261
9.3 Design Concepts 265
- 9.3.1 Abstraction 265
- 9.3.2 Architecture 265
- 9.3.3 Patterns 266
- 9.3.4 Modularity 267
- 9.3.5 Information Hiding 268
- 9.3.6 Functional Independence 268
- 9.3.7 Refinement 269
- 9.3.8 Refactoring 270
- 9.3.9 Design Classes 271

9.4 The Design Model 274
- 9.4.1 Data Design Elements 275

9.4.2 Architectural Design Elements 275
9.4.3 Interface Design Elements 276
9.4.4 Component-Level Design Elements 278
9.4.5 Deployment-Level Design Elements 279
9.5 Pattern-Based Software Design 280
9.5.1 Describing a Design Pattern 280
9.5.2 Using Patterns in Design 281
9.5.3 Frameworks 281
9.6 Summary 282
REFERENCES 283
PROBLEMS AND POINTS TO PONDER 283
FURTHER READINGS AND INFORMATION SOURCES 284

CHAPTER 10 CREATING AN ARCHITECTURAL DESIGN 286

10.1 Software Architecture 287
10.1.1 What Is Architecture? 287
10.1.2 Why Is Architecture Important? 288
10.2 Data Design 289
10.2.1 Data Design at the Architectural Level 289
10.2.2 Data Design at the Component Level 290
10.3 Architectural Styles and Patterns 291
10.3.1 A Brief Taxonomy of Architectural Styles 292
10.3.2 Architectural Patterns 296
10.3.3 Organization and Refinement 297
10.4 Architectural Design 298
10.4.1 Representing the System in Context 298
10.4.2 Defining Archetypes 300
10.4.3 Refining the Architecture into Components 301
10.4.4 Describing Instantiations of the System 303
10.5 Assessing Alternative Architectural Designs 304
10.5.1 An Architecture Trade-Off Analysis Method 304
10.5.2 Architectural Complexity 306
10.5.3 Architectural Description Languages 306
10.6 Mapping Data Flow into a Software Architecture 307
10.6.1 Transform Flow 308
10.6.2 Transaction Flow 308
10.6.3 Transform Mapping 309
10.6.4 Transaction Mapping 316
10.6.5 Refining the Architectural Design 320
10.7 Summary 320
REFERENCES 321
PROBLEMS AND POINTS TO PONDER 322
FURTHER READINGS AND INFORMATION SOURCES 323

CHAPTER 11 MODELING COMPONENT-LEVEL DESIGN 324

11.1 What Is a Component? 325
11.1.1 An Object-Oriented View 326
11.1.2 The Conventional View 327
11.1.3 A Process-Related View 330
11.2 Designing Class-Based Components 330
11.2.1 Basic Design Principles 331

11.2.2 Component-Level Design Guidelines 334
11.2.3 Cohesion 335
11.2.4 Coupling 337
11.3 Conducting Component-Level Design 339
11.4 Object Constraint Language 345
11.5 Designing Conventional Components 347
11.5.1 Graphical Design Notation 348
11.5.2 Tabular Design Notation 349
11.5.3 Program Design Language 350
11.5.4 Comparison of Design Notation 352
11.6 Summary 353
REFERENCES 354
PROBLEMS AND POINTS TO PONDER 354
FURTHER READINGS AND INFORMATION SOURCES 355

CHAPTER 12 PERFORMING USER INTERFACE DESIGN 356

12.1 The Golden Rules 357
12.1.1 Place the User in Control 357
12.1.2 Reduce the User's Memory Load 359
12.1.3 Make the Interface Consistent 360
12.2 User Interface Analysis and Design 361
12.2.1 Interface Analysis and Design Models 362
12.2.2 The Process 363
12.3 Interface Analysis 365
12.3.1 User Analysis 365
12.3.2 Task Analysis and Modeling 367
12.3.3 Analysis of Display Content 372
12.3.4 Analysis of the Work Environment 373
12.4 Interface Design Steps 373
12.4.1 Applying Interface Design Steps 374
12.4.2 User Interface Design Patterns 375
12.4.3 Design Issues 377
12.5 Design Evaluation 381
12.6 Summary 383
REFERENCES 383
PROBLEMS AND POINTS TO PONDER 384
FURTHER READINGS AND INFORMATION SOURCES 385

CHAPTER 13 TESTING STRATEGIES 386

13.1 A Strategic Approach to Software Testing 387
13.1.1 Verification and Validation 388
13.1.2 Organizing for Software Testing 388
13.1.3 A Software Testing Strategy for Conventional Architectures 390
13.1.4 A Software Testing Strategy for Object-Oriented Architectures 391
13.1.5 Criteria for Completion of Testing 392
13.2 Strategic Issues 393
13.3 Test Strategies for Conventional Software 394
13.3.1 Unit Testing 394
13.3.2 Integration Testing 397
13.4 Test Strategies for Object-Oriented Software 404
13.4.1 Unit Testing in the OO Context 404
13.4.2 Integration Testing in the OO Context 405

13.5 Validation Testing 406
- 13.5.1 Validation Test Criteria 406
- 13.5.2 Configuration Review 406
- 13.5.3 Alpha and Beta Testing 406

13.6 System Testing 408
- 13.6.1 Recovery Testing 409
- 13.6.2 Security Testing 409
- 13.6.3 Stress Testing 409
- 13.6.4 Performance Testing 410

13.7 The Art of Debugging 411
- 13.7.1 The Debugging Process 411
- 13.7.2 Psychological Considerations 413
- 13.7.3 Debugging Strategies 414
- 13.7.4 Correcting the Error 416

13.8 Summary 416

REFERENCES 417

PROBLEMS AND POINTS TO PONDER 417

FURTHER READINGS AND INFORMATION SOURCES 418

CHAPTER 14 TESTING TACTICS 420

14.1 Software Testing Fundamentals 421

14.2 Black-Box and White-Box Testing 423

14.3 White-Box Testing 424

14.4 Basis Path Testing 425
- 14.4.1 Flow Graph Notation 425
- 14.4.2 Independent Program Paths 426
- 14.4.3 Deriving Test Cases 428
- 14.4.4 Graph Matrices 431

14.5 Control Structure Testing 432
- 14.5.1 Condition Testing 432
- 14.5.2 Data Flow Testing 432
- 14.5.3 Loop Testing 433

14.6 Black-Box Testing 434
- 14.6.1 Graph-Based Testing Methods 435
- 14.6.2 Equivalence Partitioning 437
- 14.6.3 Boundary Value Analysis 438
- 14.6.4 Orthogonal Array Testing 439

14.7 Object-Oriented Testing Methods 442
- 14.7.1 The Test Case Design Implications of OO Concepts 442
- 14.7.2 Applicability of Conventional Test Case Design Methods 443
- 14.7.3 Fault-Based Testing 443
- 14.7.4 Test Cases and Class Hierarchy 444
- 14.7.5 Scenario-Based Testing 444
- 14.7.6 Testing Surface Structure and Deep Structure 446

14.8 Testing Methods Applicable at the Class Level 447
- 14.8.1 Random Testing for OO Classes 447
- 14.8.2 Partition Testing at the Class Level 448

14.9 InterClass Test Case Design 449
- 14.9.1 Multiple Class Testing 449
- 14.9.2 Tests Derived from Behavior Models 450

14.10 Testing for Specialized Environments, Architectures, and Applications 452
- 14.10.1 Testing GUIs 452

14.10.2 Testing of Client/Server Architectures 452
14.10.3 Testing Documentation and Help Facilities 453
14.10.4 Testing for Real-Time Systems 454
14.11 Testing Patterns 456
14.12 Summary 457
REFERENCES 458
PROBLEMS AND POINTS TO PONDER 459
FURTHER READINGS AND INFORMATION SOURCES 460

CHAPTER 15 PRODUCT METRICS 461

15.1 Software Quality 462
15.1.1 McCall's Quality Factors 463
15.1.2 ISO 9126 Quality Factors 464
15.1.3 The Transition to a Quantitative View 465
15.2 A Framework for Product Metrics 466
15.2.1 Measures, Metrics, and Indicators 466
15.2.2 The Challenge of Product Metrics 466
15.2.3 Measurement Principles 467
15.2.4 Goal-Oriented Software Measurement 468
15.2.5 The Attributes of Effective Software Metrics 469
15.2.6 The Product Metrics Landscape 470
15.3 Metrics for the Analysis Model 472
15.3.1 Function-Based Metrics 472
15.3.2 Metrics for Specification Quality 476
15.4 Metrics for the Design Model 477
15.4.1 Architectural Design Metrics 477
15.4.2 Metrics for Object-Oriented Design 480
15.4.3 Class-Oriented Metrics—The CK Metrics Suite 481
15.4.4 Class-Oriented Metrics—The MOOD Metrics Suite 484
15.4.5 OO Metrics Proposed by Lorenz and Kidd 485
15.4.6 Component-Level Design Metrics 486
15.4.7 Operation-Oriented Metrics 488
15.4.8 User Interface Design Metrics 489
15.5 Metrics for Source Code 490
15.6 Metrics for Testing 491
15.6.1 Halstead Metrics Applied to Testing 491
15.6.2 Metrics for Object-Oriented Testing 491
15.7 Metrics for Maintenance 492
15.8 Summary 493
REFERENCES 494
PROBLEMS AND POINTS TO PONDER 496
FURTHER READINGS AND INFORMATION SOURCES 497

PART THREE—APPLYING WEB ENGINEERING 499

CHAPTER 16 WEB ENGINEERING 500

16.1 Attributes of Web-Based Systems and Applications 501
16.2 WebApp Engineering Layers 504
16.2.1 Process 504
16.2.2 Methods 505
16.2.3 Tools and Technology 506

16.3 The Web Engineering Process 506
16.3.1 Defining the Framework 507
16.3.2 Refining the Framework 509
16.4 Web Engineering Best Practices 510
16.5 Summary 511
REFERENCES 512
PROBLEMS AND POINTS TO PONDER 512
FURTHER READINGS AND INFORMATION SOURCES 513

CHAPTER 17 INITIATING A WEBAPP PROJECT 514

17.1 Formulating Web-Based Systems 515
17.1.1 Formulation Questions 515
17.1.2 Requirements Gathering for WebApps 517
17.1.3 The Bridge to Analysis Modeling 521
17.2 Planning for Web Engineering Projects 522
17.3 The Web Engineering Team 523
17.3.1 The Players 523
17.3.2 Building the Team 524
17.4 Project Management Issues for Web Engineering 525
17.4.1 WebApp Planning—Outsourcing 526
17.4.2 WebApp Planning—In-House Web Engineering 530
17.5 Metrics for Web Engineering and WebApps 532
17.5.1 Metrics for Web Engineering Effort 533
17.5.2 Metrics for Assessing Business Value 534
17.6 "Worst Practices" for WebApp Projects 534
17.7 Summary 536
REFERENCES 536
PROBLEMS AND POINTS TO PONDER 537
FURTHER READINGS AND INFORMATION SOURCES 538

CHAPTER 18 ANALYSIS FOR WEBAPPS 539

18.1 Requirements Analysis for WebApps 540
18.1.1 The User Hierarchy 541
18.1.2 Developing Use-Cases 542
18.1.3 Refining the Use-Case Model 544
18.2 The Analysis Model for WebApps 545
18.3 The Content Model 545
18.3.1 Defining Content Objects 546
18.3.2 Content Relationships and Hierarchy 546
18.3.3 Analysis Classes for WebApps 547
18.4 The Interaction Model 548
18.5 The Functional Model 551
18.6 The Configuration Model 553
18.7 Relationship-Navigation Analysis 553
18.7.1 Relationship Analysis—Key Questions 554
18.7.2 Navigation Analysis 555
18.8 Summary 556
REFERENCES 557
PROBLEMS AND POINTS TO PONDER 557
FURTHER READINGS AND INFORMATION SOURCES 558

CHAPTER 19 DESIGN FOR WEBAPPS 559

19.1 Design Issues for Web Engineering 560
- 19.1.1 Design and WebApp Quality 560
- 19.1.2 Design Goals 563

19.2 The WebE Design Pyramid 564
19.3 WebApp Interface Design 565
- 19.3.1 Interface Design Principles and Guidelines 566
- 19.3.2 Interface Control Mechanisms 571
- 19.3.3 Interface Design Workflow 571

19.4 Aesthetic Design 573
- 19.4.1 Layout Issues 574
- 19.4.2 Graphic Design Issues 574

19.5 Content Design 575
- 19.5.1 Content Objects 575
- 19.5.2 Content Design Issues 576

19.6 Architecture Design 577
- 19.6.1 Content Architecture 577
- 19.6.2 WebApp Architecture 579

19.7 Navigation Design 581
- 19.7.1 Navigation Semantics 581
- 19.7.2 Navigation Syntax 583

19.8 Component Level Design 584
19.9 Hypermedia Design Patterns 584
19.10 Object-Oriented Hypermedia Design Method (OOHDM) 586
- 19.10.1 Conceptual Design for OOHDM 586
- 19.10.2 Navigational Design for OOHDM 587
- 19.10.3 Abstract Interface Design and Implementation 588

19.11 Design Metrics for WebApps 588
19.12 Summary 589
REFERENCES 590
PROBLEMS AND POINTS TO PONDER 592
FURTHER READINGS AND INFORMATION SOURCES 593

CHAPTER 20 TESTING FOR WEBAPPS 594

20.1 Testing Concepts for WebApps 595
- 20.1.1 Dimensions of Quality 595
- 20.1.2 Errors within a WebApp Environment 596
- 20.1.3 Testing Strategy 597
- 20.1.4 Test Planning 598

20.2 The Testing Process—An Overview 598
20.3 Content Testing 601
- 20.3.1 Content Testing Objectives 601
- 20.3.2 Database Testing 603

20.4 User Interface Testing 605
- 20.4.1 Interface Testing Strategy 605
- 20.4.2 Testing Interface Mechanisms 606
- 20.4.3 Testing Interface Semantics 608
- 20.4.4 Usability Tests 608
- 20.4.5 Compatibility Tests 610

20.5 Component-Level Testing 611

20.6 Navigation Testing 613
20.6.1 Testing Navigation Syntax 613
20.6.2 Testing Navigation Semantics 614
20.7 Configuration Testing 615
20.7.1 Server-Side Issues 616
20.7.2 Client-Side Issues 616
20.8 Security Testing 617
20.9 Performance Testing 619
20.9.1 Performance Testing Objectives 619
20.9.2 Load Testing 620
20.9.3 Stress Testing 620
20.10 Summary 622
REFERENCES 623
PROBLEMS AND POINTS TO PONDER 624
FURTHER READINGS AND INFORMATION SOURCES 625

PART FOUR—MANAGING SOFTWARE PROJECTS 627

CHAPTER 21 PROJECT MANAGEMENT 628

21.1 The Management Spectrum 629
21.1.1 The People 629
21.1.2 The Product 630
21.1.3 The Process 630
21.1.4 The Project 630
21.2 The People 631
21.2.1 The Stakeholders 631
21.2.2 Team Leaders 632
21.2.3 The Software Team 633
21.2.4 Agile Teams 636
21.2.5 Coordination and Communication Issues 637
21.3 The Product 638
21.3.1 Software Scope 638
21.3.2 Problem Decomposition 639
21.4 The Process 640
21.4.1 Melding the Product and the Process 640
21.4.2 Process Decomposition 641
21.5 The Project 642
21.6 The W^5HH Principle 644
21.7 Critical Practices 644
21.8 Summary 645
REFERENCES 646
PROBLEMS AND POINTS TO PONDER 646
FURTHER READINGS AND INFORMATION SOURCES 647

CHAPTER 22 METRICS FOR PROCESS AND PROJECTS 649

22.1 Metrics in the Process and Project Domains 650
22.1.1 Process Metrics and Software Process Improvement 650
22.1.2 Project Metrics 653
22.2 Software Measurement 654
22.2.1 Size-Oriented Metrics 655
22.2.2 Function-Oriented Metrics 656

SRINIVAS COLLEGE OF P.G.

22.2.3 Reconciling LOC and FP Metrics 656
22.2.4 Object-Oriented Metrics 658
22.2.5 Use-Case Oriented Metrics 659
22.2.6 Web Engineering Project Metrics 659
22.3 Metrics for Software Quality 661
22.3.1 Measuring Quality 662
22.3.2 Defect Removal Efficiency 663
22.4 Integrating Metrics within the Software Process 664
22.4.1 Arguments for Software Metrics 665
22.4.2 Establishing a Baseline 665
22.4.3 Metrics Collection, Computation, and Evaluation 666
22.5 Metrics for Small Organizations 666
22.6 Establishing a Software Metrics Program 668
22.7 Summary 670
REFERENCES 670
PROBLEMS AND POINTS TO PONDER 671
FURTHER READINGS AND INFORMATION SOURCES 672

CHAPTER 23 ESTIMATION 674

23.1 Observations on Estimation 675
23.2 The Project Planning Process 676
23.3 Software Scope and Feasibility 677
23.4 Resources 677
23.4.1 Human Resources 678
23.4.2 Reusable Software Resources 678
23.4.3 Environmental Resources 679
23.5 Software Project Estimation 680
23.6 Decomposition Techniques 681
23.6.1 Software Sizing 681
23.6.2 Problem-Based Estimation 682
23.6.3 An Example of LOC-Based Estimation 683
23.6.4 An Example of FP-Based Estimation 685
23.6.5 Process-Based Estimation 686
23.6.6 An Example of Process-Based Estimation 687
23.6.7 Estimation with Use-Cases 688
23.6.8 An Example of Use-Case Based Estimation 689
23.6.9 Reconciling Estimates 690
23.7 Empirical Estimation Models 691
23.7.1 The Structure of Estimation Models 692
23.7.2 The COCOMO II Model 692
23.7.3 The Software Equation 694
23.8 Estimation for Object-Oriented Projects 695
23.9 Specialized Estimation Techniques 696
23.9.1 Estimation for Agile Development 696
23.9.2 Estimation for Web Engineering Projects 697
23.10 The Make/Buy Decision 698
23.10.1 Creating a Decision Tree 699
23.10.2 Outsourcing 700
23.11 Summary 701
REFERENCES 702
PROBLEMS AND POINTS TO PONDER 703
FURTHER READINGS AND INFORMATION SOURCES 703

CHAPTER 24 PROJECT SCHEDULING 705

24.1 Basic Concepts 706
24.2 Project Scheduling 708
  24.2.1 Basic Principles 709
  24.2.2 The Relationship Between People and Effort 710
  24.2.3 Effort Distribution 712
24.3 Defining a Task Set for the Software Project 713
  24.3.1 A Task Set Example 714
  24.3.2 Refinement of Major Tasks 714
24.4 Defining a Task Network 715
24.5 Scheduling 716
  24.5.1 Timeline Charts 717
  24.5.2 Tracking the Schedule 718
  24.5.3 Tracking Progress for an OO Project 720
24.6 Earned Value Analysis 722
24.7 Summary 723
REFERENCES 723
PROBLEMS AND POINTS TO PONDER 724
FURTHER READINGS AND INFORMATION SOURCES 725

CHAPTER 25 RISK MANAGEMENT 726

25.1 Reactive vs. Proactive Risk Strategies 727
25.2 Software Risks 728
25.3 Risk Identification 729
  25.3.1 Assessing Overall Project Risk 730
  25.3.2 Risk Components and Drivers 731
25.4 Risk Projection 732
  25.4.1 Developing a Risk Table 733
  25.4.2 Assessing Risk Impact 735
25.5 Risk Refinement 737
25.6 Risk Mitigation, Monitoring, and Management 737
25.7 The RMMM Plan 740
25.8 Summary 741
REFERENCES 742
PROBLEMS AND POINTS TO PONDER 742
FURTHER READIN20GS AND INFORMATION SOURCES 743

CHAPTER 26 QUALITY MANAGEMENT 744

26.1 Quality Concepts 745
  26.1.1 Quality 746
  26.1.2 Quality Control 746
  26.1.3 Quality Assurance 747
  26.1.4 Cost of Quality 747
26.2 Software Quality Assurance 748
  26.2.1 Background Issues 749
  26.2.2 SQA Activities 749
26.3 Software Reviews 751
  26.3.1 Cost Impact of Software Defects 752
  26.3.2 Defect Amplification and Removal 752
26.4 Formal Technical Reviews 754
  26.4.1 The Review Meeting 754
  26.4.2 Review Reporting and Record Keeping 755

26.4.3 Review Guidelines 756
26.4.4 Sample-Driven Reviews 757
26.5 Formal Approaches to SQA 759
26.6 Statistical Software Quality Assurance 759
26.6.1 A Generic Example 760
26.6.2 Six Sigma for Software Engineering 761
26.7 Software Reliability 762
26.7.1 Measures of Reliability and Availability 763
26.7.2 Software Safety 763
26.8 The ISO 9000 Quality Standards 765
26.9 The SQA Plan 766
26.10 Summary 767
REFERENCES 768
PROBLEMS AND POINTS TO PONDER 769
FURTHER READINGS AND INFORMATION SOURCES 769

CHAPTER 27 CHANGE MANAGEMENT 771

✓ 27.1 Software Configuration Management 772
27.1.1 A SCM Scenario 773
27.1.2 Elements of a Configuration Management System 774
27.1.3 Baselines 775
27.1.4 Software Configuration Items 775
27.2 The SCM Repository 777
27.2.1 The Role of the Repository 777
27.2.2 General Features and Content 778
27.2.3 SCM Features 779
27.3 The SCM Process 780
27.3.1 Identification of Objects in the Software Configuration 781
27.3.2 Version Control 782
27.3.3 Change Control 784
27.3.4 Configuration Audit 787
27.3.5 Status Reporting 788
27.4 Configuration Management for Web Engineering 788
27.4.1 Configuration Management Issues for WebApps 789
27.4.2 WebApp Configuration Objects 790
27.4.3 Content Management 790
27.4.4 Change Management 793
27.4.5 Version Control 795
27.4.6 Auditing and Reporting 796
27.5 Summary 797
REFERENCES 798
PROBLEMS AND POINTS TO PONDER 799
FURTHER READINGS AND INFORMATION SOURCES 800

PART FIVE—ADVANCED TOPICS IN SOFTWARE ENGINEERING 801

CHAPTER 28 FORMAL METHODS 802

28.1 Basic Concepts 803
28.1.1 Deficiencies of Less Formal Approaches 804
28.1.2 Mathematics in Software Development 805
28.1.3 Formal Methods Concepts 805

28.2 Mathematical Preliminaries 808
28.2.1 Sets and Constructive Specification 808
28.2.2 Set Operators 810
28.2.3 Logic Operators 812
28.2.4 Sequences 812
28.3 Applying Mathematical Notation for Formal Specification 813
28.4 Formal Specification Languages 815
28.5 Object Constraint Language (OCL) 816
28.5.1 A Brief Overview of OCL Syntax and Semantics 816
28.5.2 An Example Using OCL 818
28.6 The Z Specification Language 820
28.6.1 A Brief Overview of Z Syntax and Semantics 820
28.6.2 An Example Using Z 820
28.7 The Ten Commandments of Formal Methods 823
28.8 Formal Methods—The Road Ahead 824
28.9 Summary 825
REFERENCES 825
PROBLEMS AND POINTS TO PONDER 826
FURTHER READINGS AND INFORMATION SOURCES 827

CHAPTER 29 CLEANROOM SOFTWARE ENGINEERING 828

29.1 The Cleanroom Approach 829
29.1.1 The Cleanroom Strategy 830
29.1.2 What Makes Cleanroom Different? 832
29.2 Functional Specification 833
29.2.1 Black-Box Specification 834
29.2.2 State-Box Specification 835
29.2.3 Clear-Box Specification 835
29.3 Cleanroom Design 836
29.3.1 Design Refinement and Verification 836
29.3.2 Advantages of Design Verification 840
29.4 Cleanroom Testing 841
29.4.1 Statistical Use Testing 842
29.4.2 Certification 843
29.5 Summary 844
REFERENCES 844
PROBLEMS AND POINTS TO PONDER 845
FURTHER READINGS AND INFORMATION SOURCES 846

CHAPTER 30 COMPONENT-BASED DEVELOPMENT 847

30.1 Engineering of Component-Based Systems 848
30.2 The CBSE Process 850
30.3 Domain Engineering 851
30.3.1 The Domain Analysis Process 851
30.3.2 Characterization Functions 852
30.3.3 Structural Modeling and Structure Points 853
30.4 Component-Based Development 854
30.4.1 Component Qualification, Adaptation, and Composition 854
30.4.2 Component Engineering 857
30.4.3 Analysis and Design for Reuse 858

30.5 Classifying and Retrieving Components 859
30.5.1 Describing Reusable Components 859
30.5.2 The Reuse Environment 861
30.6 Economics of CBSE 862
30.6.1 Impact on Quality, Productivity, and Cost 862
30.6.2 Cost Analysis Using Structure Points 863
30.7 Summary 864
REFERENCES 865
PROBLEMS AND POINTS TO PONDER 866
RURTHER READINGS AND INFORMATION SOURCES 867

CHAPTER 31 REENGINEERING 869

31.1 Business Process Reengineering 870
31.1.1 Business Processes 871
31.1.2 A BPR Model 871
31.2 Software Reengineering 873
31.2.1 Software Maintenance 873
31.2.2 A Software Reengineering Process Model 874
31.3 Reverse Engineering 878
31.3.1 Reverse Engineering to Understand Data 880
31.3.2 Reverse Engineering to Understand Processing 880
31.3.3 Reverse Engineering User Interfaces 881
31.4 Restructuring 882
31.4.1 Code Restructuring 882
31.4.2 Data Restructuring 883
31.5 Forward Engineering 884
31.5.1 Forward Engineering for Client/Server Architectures 885
31.5.2 Forward Engineering for Object-Oriented Architectures 886
31.5.3 Forward Engineering User Interfaces 887
31.6 The Economics of Reengineering 887
31.7 Summary 888
REFERENCES 889
PROBLEMS AND POINTS TO PONDER 890
FURTHER READINGS AND INFORMATION SOURCES 891

CHAPTER 32 THE ROAD AHEAD 892

32.1 The Importance of Software—Revisited 893
32.2 The Scope of Change 893
32.3 People and the Way They Build Systems 895
32.4 The "New" Software Engineering Process 896
32.5 New Modes for Representing Information 897
32.6 Technology as a Driver 899
32.7 The Software Engineer's Responsibility 900
32.8 A Concluding Comment 902
REFERENCES 903
PROBLEMS AND POINTS TO PONDER 903
FURTHER READINGS AND INFORMATION SOURCES 903

Index 905