

Roger S. Pressman

sixth edition

# SOFTWARE ENGINEERING

A Practitioner's Approach



# Software Engineering

**A PRACTITIONER'S APPROACH**

**GIFT OF THE ASIA FOUNDATION  
NOT FOR RE-SALE**

**QUÀ TẶNG CỦA QUỸ CHÂU Á  
KHÔNG ĐƯỢC BÁN LẠI**

# Software Engineering

**A PRACTITIONER'S APPROACH**

SIXTH EDITION

**Roger S. Pressman, Ph.D.**



**Higher Education**

Boston Burr Ridge, IL Dubuque, IA Madison, WI New York San Francisco St. Louis  
Bangkok Bogotá Caracas Kuala Lumpur Lisbon London Madrid Mexico City  
Milan Montreal New Delhi Santiago Seoul Singapore Sydney Taipei Toronto

# CONTENTS AT A GLANCE

CHAPTER 1 Software and Software Engineering 1

## **PART ONE The Software Process 19**

CHAPTER 2 Process: A Generic View 20  
CHAPTER 3 Prescriptive Process Models 45  
CHAPTER 4 Agile Development 71

## **PART TWO Software Engineering Practice 95**

CHAPTER 5 Practice: A Generic View 96  
CHAPTER 6 System Engineering 122  
CHAPTER 7 Requirements Engineering 142  
CHAPTER 8 Analysis Modeling 175  
CHAPTER 9 Design Engineering 226  
CHAPTER 10 Architectural Design 254  
CHAPTER 11 Component-Level Design 292  
CHAPTER 12 User Interface Design 324  
CHAPTER 13 Software Testing Strategies 354  
CHAPTER 14 Software Testing Techniques 388  
CHAPTER 15 Product Metrics for Software 429

## **PART THREE Applying Web Engineering 467**

CHAPTER 16 Web Engineering 468  
CHAPTER 17 Formulation and Planning 482  
CHAPTER 18 Analysis Modeling for Web Applications 507  
CHAPTER 19 Design Modeling for Web Applications 527  
CHAPTER 20 Testing Web Applications 562

## **PART FOUR Managing Software Projects 595**

CHAPTER 21 Project Management Concepts 596  
CHAPTER 22 Process and Project Metrics 617  
CHAPTER 23 Estimation for Software Projects 642  
CHAPTER 24 Software Project Scheduling 673  
CHAPTER 25 Risk Management 694

CHAPTER 26	Quality Management	712
CHAPTER 27	Change Management	739

**PART FIVE****Advanced Topics in Software Engineering** 769

CHAPTER 28	Formal Methods	770
CHAPTER 29	Cleanroom Software Engineering	796
CHAPTER 30	Component-Based Software Engineering	815
CHAPTER 31	Reengineering	837
CHAPTER 32	The Road Ahead	860

# TABLE OF CONTENTS

*Preface* xxv

*Walkthrough* xxix

## **CHAPTER 1 SOFTWARE AND SOFTWARE ENGINEERING 1**

---

- 1.1 The Evolving Role of Software 2
- 1.2 Software 4
- 1.3 The Changing Nature of Software 8
- 1.4 Legacy Software 10
  - 1.4.1 The Quality of Legacy Software 11
  - 1.4.2 Software Evolution 11
- 1.5 Software Myths 13
- 1.6 How It All Starts 15
- 1.7 Summary 16
- REFERENCES 17
- PROBLEMS AND POINTS TO PONDER 17
- FURTHER READINGS AND INFORMATION SOURCES 18

## **PART ONE—THE SOFTWARE PROCESS 19**

---

### **CHAPTER 2 PROCESS: A GENERIC VIEW 20**

---

- 2.1 Software Engineering—A Layered Technology 21
- 2.2 A Process Framework 22
- 2.3 The Capability Maturity Model Integration (CMMI) 27
- 2.4 Process Patterns 31
- 2.5 Process Assessment 34
- 2.6 Personal and Team Process Models 36
  - 2.6.1 Personal Software Process (PSP) 36
  - 2.6.2 Team Software Process (TSP) 38
- 2.7 Process Technology 39
- 2.8 Product and Process 40
- 2.9 Summary 41
- REFERENCES 42
- PROBLEMS AND POINTS TO PONDER 43
- FURTHER READINGS AND INFORMATION SOURCES 43

### **CHAPTER 3 PRESCRIPTIVE PROCESS MODELS 45**

---

- 3.1 Prescriptive Models 46
- 3.2 The Waterfall Model 47
- 3.3 Incremental Process Models 48
  - 3.3.1 The Incremental Model 48
  - 3.3.2 The RAD Model 49
- 3.4 Evolutionary Process Models 51
  - 3.4.1 Prototyping 51
  - 3.4.2 The Spiral Model 54

3.4.3	The Concurrent Development Model	56
3.4.4	A Final Comment on Evolutionary Processes	57
3.5	Specialized Process Models	59
3.5.1	Component-Based Development	59
3.5.2	The Formal Methods Model	60
3.5.3	Aspect-Oriented Software Development	61
3.6	The Unified Process	62
3.6.1	A Brief History	63
3.6.2	Phases of the Unified Process	64
3.6.3	Unified Process Work Products	66
3.7	Summary	67
	REFERENCES	68
	PROBLEMS AND POINTS TO PONDER	69
	FURTHER READINGS AND INFORMATION SOURCES	70

## **CHAPTER 4 AGILE DEVELOPMENT 71**

---

4.1	What Is Agility?	73
4.2	What Is an Agile Process?	74
4.2.1	The Politics of Agile Development	75
4.2.2	Human Factors	76
4.3	Agile Process Models	77
4.3.1	Extreme Programming (XP)	78
4.3.2	Adaptive Software Development (ASD)	82
4.3.3	Dynamic Systems Development Method (DSDM)	84
4.3.4	Scrum	85
4.3.5	Crystal	87
4.3.6	Feature Driven Development (FDD)	88
4.3.7	Agile Modeling (AM)	89
4.4	Summary	91
	REFERENCES	92
	PROBLEMS AND POINTS TO PONDER	93
	FURTHER READINGS AND INFORMATION SOURCES	94

## **PART TWO—SOFTWARE ENGINEERING PRACTICE 95**

---

### **CHAPTER 5 PRACTICE: A GENERIC VIEW 96**

---

5.1	Software Engineering Practice	97
5.1.1	The Essence of Practice	97
5.1.2	Core Principles	99
5.2	Communication Practices	101
5.3	Planning Practices	104
5.4	Modeling Practices	107
5.4.1	Analysis Modeling Principles	108
5.4.2	Design Modeling Principles	109
5.5	Construction Practice	112
5.5.1	Coding Principles and Concepts	113
5.5.2	Testing Principles	114
5.6	Deployment	116
5.7	Summary	118

REFERENCES	119
PROBLEMS AND POINTS TO PONDER	120
FURTHER READINGS AND INFORMATION SOURCES	120

## **CHAPTER 6    SYSTEM ENGINEERING    122**

---

6.1	Computer-Based Systems	123
6.2	The System Engineering Hierarchy	125
6.2.1	System Modeling	126
6.2.2	System Simulation	128
6.3	Business Process Engineering: An Overview	129
6.4	Product Engineering: An Overview	130
6.5	System Modeling	132
6.5.1	Hatley-Pirbhai Modeling	133
6.5.2	System Modeling with UML	135
6.6	Summary	139
REFERENCES	140	
PROBLEMS AND POINTS TO PONDER	140	
FURTHER READINGS AND INFORMATION SOURCES	141	

## **CHAPTER 7    REQUIREMENTS ENGINEERING    142**

---

7.1	A Bridge to Design and Construction	143
7.2	Requirements Engineering Tasks	144
7.2.1	Inception	144
7.2.2	Elicitation	145
7.2.3	Elaboration	145
7.2.4	Negotiation	146
7.2.5	Specification	147
7.2.6	Validation	147
7.2.7	Requirements Management	148
7.3	Initiating the Requirements Engineering Process	149
7.3.1	Identifying the Stakeholders	150
7.3.2	Recognizing Multiple Viewpoints	150
7.3.3	Working toward Collaboration	151
7.3.4	Asking the First Questions	151
7.4	Eliciting Requirements	152
7.4.1	Collaborative Requirements Gathering	153
7.4.2	Quality Function Deployment	156
7.4.3	User Scenarios	157
7.4.4	Elicitation Work Products	158
7.5	Developing Use-Cases	159
7.6	Building the Analysis Model	164
7.6.1	Elements of the Analysis Model	164
7.6.2	Analysis Patterns	168
7.7	Negotiating Requirements	169
7.8	Validating Requirements	171
7.9	Summary	172
REFERENCES	172	
PROBLEMS AND POINTS TO PONDER	173	
FURTHER READINGS AND INFORMATION SOURCES	174	



**CHAPTER 8 ANALYSIS MODELING 175**

8.1	Requirements Analysis	176
8.1.1	Overall Objective and Philosophy	177
8.1.2	Analysis Rules of Thumb	178
8.1.3	Domain Analysis	178
8.2	Analysis Modeling Approaches	179
8.3	Data Modeling Concepts	181
8.3.1	Data Objects	181
8.3.2	Data Attributes	182
8.3.3	Relationships	182
8.3.4	Cardinality and Modality	183
8.4	Object-Oriented Analysis	185
8.5	Scenario-Based Modeling	186
8.5.1	Writing Use-Cases	186
8.5.2	Developing an Activity Diagram	191
8.5.3	Swimlane Diagrams	192
8.6	Flow-Oriented Modeling	194
8.6.1	Creating a Data Flow Model	194
8.6.2	Creating a Control Flow Model	197
8.6.3	The Control Specification	198
8.6.4	The Process Specification	200
8.7	Class-Based Modeling	201
8.7.1	Identifying Analysis Classes	201
8.7.2	Specifying Attributes	204
8.7.3	Defining Operations	205
8.7.4	Class-Responsibility-Collaborator (CRC) Modeling	208
8.7.5	Associations and Dependencies	214
8.7.6	Analysis Packages	215
8.8	Creating a Behavioral Model	216
8.8.1	Identifying Events with the Use-Case	217
8.8.2	State Representations	218
8.9	Summary	221
	REFERENCES	222
	PROBLEMS AND POINTS TO PONDER	223
	FURTHER READINGS AND INFORMATION SOURCES	224

**CHAPTER 9 DESIGN ENGINEERING 226**

9.1	Design within the Context of Software Engineering	227
9.2	Design Process and Design Quality	229
9.3	Design Concepts	233
9.3.1	Abstraction	233
9.3.2	Architecture	233
9.3.3	Patterns	234
9.3.4	Modularity	235
9.3.5	Information Hiding	236
9.3.6	Functional Independence	236
9.3.7	Refinement	237
9.3.8	Refactoring	238
9.3.9	Design Classes	239
9.4	The Design Model	242
9.4.1	Data Design Elements	243

9.4.2	Architectural Design Elements	243
9.4.3	Interface Design Elements	244
9.4.4	Component-Level Design Elements	246
9.4.5	Deployment-Level Design Elements	247
9.5	Pattern-Based Software Design	248
9.5.1	Describing a Design Pattern	248
9.5.2	Using Patterns in Design	249
9.5.3	Frameworks	249
9.6	Summary	250
	REFERENCES	251
	PROBLEMS AND POINTS TO PONDER	251
	FURTHER READINGS AND INFORMATION SOURCES	252

## **CHAPTER 10 ARCHITECTURAL DESIGN 254**

---

10.1	Software Architecture	255
10.1.1	What Is Architecture?	255
10.1.2	Why Is Architecture Important?	256
10.2	Data Design	257
10.2.1	Data Design at the Architectural Level	257
10.2.2	Data Design at the Component Level	258
10.3	Architectural Styles and Patterns	259
10.3.1	A Brief Taxonomy of Architectural Styles	260
10.3.2	Architectural Patterns	264
10.3.3	Organization and Refinement	265
10.4	Architectural Design	266
10.4.1	Representing the System in Context	266
10.4.2	Defining Archetypes	268
10.4.3	Refining the Architecture into Components	269
10.4.4	Describing Instantiations of the System	271
10.5	Assessing Alternative Architectural Designs	272
10.5.1	An Architecture Trade-Off Analysis Method	272
10.5.2	Architectural Complexity	274
10.5.3	Architectural Description Languages	274
10.6	Mapping Data Flow into a Software Architecture	275
10.6.1	Transform Flow	276
10.6.2	Transaction Flow	276
10.6.3	Transform Mapping	277
10.6.4	Transaction Mapping	284
10.6.5	Refining the Architectural Design	288
10.7	Summary	288
	REFERENCES	289
	PROBLEMS AND POINTS TO PONDER	290
	FURTHER READINGS AND INFORMATION SOURCES	291

## **CHAPTER 11 COMPONENT-LEVEL DESIGN 292**

---

11.1	What Is a Component?	293
11.1.1	An Object-Oriented View	294
11.1.2	The Conventional View	295
11.1.3	A Process-Related View	298
11.2	Designing Class-Based Components	298
11.2.1	Basic Design Principles	299

11.2.2	Component-Level Design Guidelines	302
11.2.3	Cohesion	303
11.2.4	Coupling	305
11.3	Conducting Component-Level Design	307
11.4	Object Constraint Language	313
11.5	Designing Conventional Components	315
11.5.1	Graphical Design Notation	316
11.5.2	Tabular Design Notation	317
11.5.3	Program Design Language	318
11.5.4	Comparison of Design Notation	320
11.6	Summary	321
	REFERENCES	322
	PROBLEMS AND POINTS TO PONDER	322
	FURTHER READINGS AND INFORMATION SOURCES	323

## **CHAPTER 12 USER INTERFACE DESIGN 324**

---

12.1	The Golden Rules	325
12.1.1	Place the User in Control	325
12.1.2	Reduce the User's Memory Load	327
12.1.3	Make the Interface Consistent	328
12.2	User Interface Analysis and Design	329
12.2.1	Interface Analysis and Design Models	330
12.2.2	The Process	331
12.3	Interface Analysis	333
12.3.1	User Analysis	333
12.3.2	Task Analysis and Modeling	335
12.3.3	Analysis of Display Content	340
12.3.4	Analysis of the Work Environment	341
12.4	Interface Design Steps	341
12.4.1	Applying Interface Design Steps	342
12.4.2	User Interface Design Patterns	343
12.4.3	Design Issues	345
12.5	Design Evaluation	349
12.6	Summary	351
	REFERENCES	351
	PROBLEMS AND POINTS TO PONDER	352
	FURTHER READINGS AND INFORMATION SOURCES	353

## **CHAPTER 13 SOFTWARE TESTING STRATEGIES 354**

---

13.1	A Strategic Approach to Software Testing	355
13.1.1	Verification and Validation	356
13.1.2	Organizing for Software Testing	356
13.1.3	A Software Testing Strategy for Conventional Architectures	358
13.1.4	A Software Testing Strategy for Object-Oriented Architectures	359
13.1.5	Criteria for Completion of Testing	360
13.2	Strategic Issues	361
13.3	Test Strategies for Conventional Software	362
13.3.1	Unit Testing	362
13.3.2	Integration Testing	365
13.4	Test Strategies for Object-Oriented Software	372
13.4.1	Unit Testing in the OO Context	372
13.4.2	Integration Testing in the OO Context	373

13.5	Validation Testing	374
13.5.1	Validation Test Criteria	374
13.5.2	Configuration Review	374
13.5.3	Alpha and Beta Testing	374
13.6	System Testing	376
13.6.1	Recovery Testing	377
13.6.2	Security Testing	377
13.6.3	Stress Testing	377
13.6.4	Performance Testing	378
13.7	The Art of Debugging	379
13.7.1	The Debugging Process	379
13.7.2	Psychological Considerations	381
13.7.3	Debugging Strategies	382
13.7.4	Correcting the Error	384
13.8	Summary	384
	REFERENCES	385
	PROBLEMS AND POINTS TO PONDER	385
	FURTHER READINGS AND INFORMATION SOURCES	386

## **CHAPTER 14 SOFTWARE TESTING TECHNIQUES 388**

---

14.1	Software Testing Fundamentals	389
14.2	Black-Box and White-Box Testing	391
14.3	White-Box Testing	392
14.4	Basis Path Testing	393
14.4.1	Flow Graph Notation	393
14.4.2	Independent Program Paths	394
14.4.3	Deriving Test Cases	396
14.4.4	Graph Matrices	399
14.5	Control Structure Testing	400
14.5.1	Condition Testing	400
14.5.2	Data Flow Testing	400
14.5.3	Loop Testing	401
14.6	Black-Box Testing	402
14.6.1	Graph-Based Testing Methods	403
14.6.2	Equivalence Partitioning	405
14.6.3	Boundary Value Analysis	406
14.6.4	Orthogonal Array Testing	407
14.7	Object-Oriented Testing Methods	410
14.7.1	The Test Case Design Implications of OO Concepts	410
14.7.2	Applicability of Conventional Test Case Design Methods	411
14.7.3	Fault-Based Testing	411
14.7.4	Test Cases and Class Hierarchy	412
14.7.5	Scenario-Based Testing	412
14.7.6	Testing Surface Structure and Deep Structure	414
14.8	Testing Methods Applicable at the Class Level	415
14.8.1	Random Testing for OO Classes	415
14.8.2	Partition Testing at the Class Level	416
14.9	InterClass Test Case Design	417
14.9.1	Multiple Class Testing	417
14.9.2	Tests Derived from Behavior Models	418
14.10	Testing for Specialized Environments, Architectures, and Applications	420
14.10.1	Testing GUIs	420

14.10.2	Testing of Client/Server Architectures	420
14.10.3	Testing Documentation and Help Facilities	421
14.10.4	Testing for Real-Time Systems	422
14.11	Testing Patterns	424
14.12	Summary	425
	REFERENCES	426
	PROBLEMS AND POINTS TO PONDER	427
	FURTHER READINGS AND INFORMATION SOURCES	428

## **CHAPTER 15 PRODUCT METRICS FOR SOFTWARE 429**

---

15.1	Software Quality	430
15.1.1	McCall's Quality Factors	431
15.1.2	ISO 9126 Quality Factors	432
15.1.3	The Transition to a Quantitative View	433
15.2	A Framework for Product Metrics	434
15.2.1	Measures, Metrics, and Indicators	434
15.2.2	The Challenge of Product Metrics	434
15.2.3	Measurement Principles	435
15.2.4	Goal-Oriented Software Measurement	436
15.2.5	The Attributes of Effective Software Metrics	437
15.2.6	The Product Metrics Landscape	438
15.3	Metrics for the Analysis Model	440
15.3.1	Function-Based Metrics	440
15.3.2	Metrics for Specification Quality	444
15.4	Metrics for the Design Model	445
15.4.1	Architectural Design Metrics	445
15.4.2	Metrics for Object-Oriented Design	448
15.4.3	Class-Oriented Metrics—The CK Metrics Suite	449
15.4.4	Class-Oriented Metrics—The MOOD Metrics Suite	452
15.4.5	OO Metrics Proposed by Lorenz and Kidd	453
15.4.6	Component-Level Design Metrics	454
15.4.7	Operation-Oriented Metrics	456
15.4.8	User Interface Design Metrics	457
15.5	Metrics for Source Code	458
15.6	Metrics for Testing	459
15.6.1	Halstead Metrics Applied to Testing	459
15.6.2	Metrics for Object-Oriented Testing	459
15.7	Metrics for Maintenance	460
15.8	Summary	461
	REFERENCES	462
	PROBLEMS AND POINTS TO PONDER	464
	FURTHER READINGS AND INFORMATION SOURCES	465

## **PART THREE—APPLYING WEB ENGINEERING 467**

---

### **CHAPTER 16 WEB ENGINEERING 468**

---

16.1	Attributes of Web-Based Systems and Applications	469
16.2	WebApp Engineering Layers	472
16.2.1	Process	472
16.2.2	Methods	473
16.2.3	Tools and Technology	474

- 16.3 The Web Engineering Process 474
  - 16.3.1 Defining the Framework 475
  - 16.3.2 Refining the Framework 477
- 16.4 Web Engineering Best Practices 478
- 16.5 Summary 479

REFERENCES 480

PROBLEMS AND POINTS TO PONDER 480

FURTHER READINGS AND INFORMATION SOURCES 481

## **CHAPTER 17 FORMULATION AND PLANNING FOR WEB ENGINEERING 482**

---

- 17.1 Formulating Web-Based Systems 483
  - 17.1.1 Formulation Questions 483
  - 17.1.2 Requirements Gathering for WebApps 485
  - 17.1.3 The Bridge to Analysis Modeling 489
- 17.2 Planning for Web Engineering Projects 490
- 17.3 The Web Engineering Team 491
  - 17.3.1 The Players 491
  - 17.3.2 Building the Team 492
- 17.4 Project Management Issues for Web Engineering 493
  - 17.4.1 WebApp Planning—Outsourcing 494
  - 17.4.2 WebApp Planning—In-House Web Engineering 498
- 17.5 Metrics for Web Engineering and WebApps 500
  - 17.5.1 Metrics for Web Engineering Effort 501
  - 17.5.2 Metrics for Assessing Business Value 502
- 17.6 “Worst Practices” for WebApp Projects 502
- 17.7 Summary 504

REFERENCES 504

PROBLEMS AND POINTS TO PONDER 505

FURTHER READINGS AND INFORMATION SOURCES 506

## **CHAPTER 18 ANALYSIS MODELING FOR WEB APPLICATIONS 507**

---

- 18.1 Requirements Analysis for WebApps 508
  - 18.1.1 The User Hierarchy 509
  - 18.1.2 Developing Use-Cases 510
  - 18.1.3 Refining the Use-Case Model 512
- 18.2 The Analysis Model for WebApps 513
- 18.3 The Content Model 513
  - 18.3.1 Defining Content Objects 514
  - 18.3.2 Content Relationships and Hierarchy 514
  - 18.3.3 Analysis Classes for WebApps 515
- 18.4 The Interaction Model 516
- 18.5 The Functional Model 519
- 18.6 The Configuration Model 521
- 18.7 Relationship-Navigation Analysis 521
  - 18.7.1 Relationship Analysis—Key Questions 522
  - 18.7.2 Navigation Analysis 523
- 18.8 Summary 524

REFERENCES 525

PROBLEMS AND POINTS TO PONDER 525

FURTHER READINGS AND INFORMATION SOURCES 526

**CHAPTER 19 DESIGN MODELING FOR WEB APPLICATIONS 527**

19.1	Design Issues for Web Engineering	528
19.1.1	Design and WebApp Quality	528
19.1.2	Design Goals	531
19.2	The WebE Design Pyramid	532
19.3	WebApp Interface Design	533
19.3.1	Interface Design Principles and Guidelines	534
19.3.2	Interface Control Mechanisms	539
19.3.3	Interface Design Workflow	539
19.4	Aesthetic Design	541
19.4.1	Layout Issues	542
19.4.2	Graphic Design Issues	542
19.5	Content Design	543
19.5.1	Content Objects	543
19.5.2	Content Design Issues	544
19.6	Architecture Design	545
19.6.1	Content Architecture	545
19.6.2	WebApp Architecture	547
19.7	Navigation Design	549
19.7.1	Navigation Semantics	549
19.7.2	Navigation Syntax	551
19.8	Component Level Design	552
19.9	Hypermedia Design Patterns	552
19.10	Object-Oriented Hypermedia Design Method (OOHDM)	554
19.10.1	Conceptual Design for OOHDM	554
19.10.2	Navigational Design for OOHDM	555
19.10.3	Abstract Interface Design and Implementation	556
19.11	Design Metrics for WebApps	556
19.12	Summary	557
	REFERENCES	558
	PROBLEMS AND POINTS TO PONDER	560
	FURTHER READINGS AND INFORMATION SOURCES	561

**CHAPTER 20 TESTING WEB APPLICATIONS 562**

20.1	Testing Concepts for WebApps	563
20.1.1	Dimensions of Quality	563
20.1.2	Errors within a WebApp Environment	564
20.1.3	Testing Strategy	565
20.1.4	Test Planning	566
20.2	The Testing Process—An Overview	566
20.3	Content Testing	569
20.3.1	Content Testing Objectives	569
20.3.2	Database Testing	571
20.4	User Interface Testing	573
20.4.1	Interface Testing Strategy	573
20.4.2	Testing Interface Mechanisms	574
20.4.3	Testing Interface Semantics	576
20.4.4	Usability Tests	576
20.4.5	Compatibility Tests	578
20.5	Component-Level Testing	579

20.6	Navigation Testing	581
20.6.1	Testing Navigation Syntax	581
20.6.2	Testing Navigation Semantics	582
20.7	Configuration Testing	583
20.7.1	Server-Side Issues	584
20.7.2	Client-Side Issues	584
20.8	Security Testing	585
20.9	Performance Testing	587
20.9.1	Performance Testing Objectives	587
20.9.2	Load Testing	588
20.9.3	Stress Testing	588
20.10	Summary	590
	REFERENCES	591
	PROBLEMS AND POINTS TO PONDER	592
	FURTHER READINGS AND INFORMATION SOURCES	593

## **PART FOUR—MANAGING SOFTWARE PROJECTS 595**

### **CHAPTER 21 PROJECT MANAGEMENT CONCEPTS 596**

21.1	The Management Spectrum	597
21.1.1	The People	597
21.1.2	The Product	598
21.1.3	The Process	598
21.1.4	The Project	598
21.2	The People	599
21.2.1	The Stakeholders	599
21.2.2	Team Leaders	600
21.2.3	The Software Team	601
21.2.4	Agile Teams	604
21.2.5	Coordination and Communication Issues	605
21.3	The Product	606
21.3.1	Software Scope	606
21.3.2	Problem Decomposition	607
21.4	The Process	608
21.4.1	Melding the Product and the Process	608
21.4.2	Process Decomposition	609
21.5	The Project	610
21.6	The W <sup>5</sup> HH Principle	612
21.7	Critical Practices	612
21.8	Summary	613
	REFERENCES	614
	PROBLEMS AND POINTS TO PONDER	614
	FURTHER READINGS AND INFORMATION SOURCES	615

### **CHAPTER 22 PROCESS AND PROJECT METRICS 617**

22.1	Metrics in the Process and Project Domains	618
22.1.1	Process Metrics and Software Process Improvement	618
22.1.2	Project Metrics	621
22.2	Software Measurement	622
22.2.1	Size-Oriented Metrics	623
22.2.2	Function-Oriented Metrics	624



22.2.3	Reconciling LOC and FP Metrics	624
22.2.4	Object-Oriented Metrics	626
22.2.5	Use-Case Oriented Metrics	627
22.2.6	Web Engineering Project Metrics	627
22.3	Metrics for Software Quality	629
22.3.1	Measuring Quality	630
22.3.2	Defect Removal Efficiency	631
22.4	Integrating Metrics within the Software Process	632
22.4.1	Arguments for Software Metrics	633
22.4.2	Establishing a Baseline	633
22.4.3	Metrics Collection, Computation, and Evaluation	634
22.5	Metrics for Small Organizations	634
22.6	Establishing a Software Metrics Program	636
22.7	Summary	638
	REFERENCES	638
	PROBLEMS AND POINTS TO PONDER	639
	FURTHER READINGS AND INFORMATION SOURCES	640

## **CHAPTER 23 ESTIMATION FOR SOFTWARE PROJECTS 642**

23.1	Observations on Estimation	643
23.2	The Project Planning Process	644
23.3	Software Scope and Feasibility	645
23.4	Resources	645
23.4.1	Human Resources	646
23.4.2	Reusable Software Resources	646
23.4.3	Environmental Resources	647
23.5	Software Project Estimation	648
23.6	Decomposition Techniques	649
23.6.1	Software Sizing	649
23.6.2	Problem-Based Estimation	650
23.6.3	An Example of LOC-Based Estimation	651
23.6.4	An Example of FP-Based Estimation	653
23.6.5	Process-Based Estimation	654
23.6.6	An Example of Process-Based Estimation	655
23.6.7	Estimation with Use-Cases	656
23.6.8	An Example of Use-Case Based Estimation	657
23.6.9	Reconciling Estimates	658
23.7	Empirical Estimation Models	659
23.7.1	The Structure of Estimation Models	660
23.7.2	The COCOMO II Model	660
23.7.3	The Software Equation	662
23.8	Estimation for Object-Oriented Projects	663
23.9	Specialized Estimation Techniques	664
23.9.1	Estimation for Agile Development	664
23.9.2	Estimation for Web Engineering Projects	665
23.10	The Make/Buy Decision	666
23.10.1	Creating a Decision Tree	667
23.10.2	Outsourcing	668
23.11	Summary	669
	REFERENCES	670
	PROBLEMS AND POINTS TO PONDER	671
	FURTHER READINGS AND INFORMATION SOURCES	671

**CHAPTER 24 SOFTWARE PROJECT SCHEDULING 673**

---

- 24.1 Basic Concepts 674
- 24.2 Project Scheduling 676
  - 24.2.1 Basic Principles 677
  - 24.2.2 The Relationship Between People and Effort 678
  - 24.2.3 Effort Distribution 680
- 24.3 Defining a Task Set for the Software Project 681
  - 24.3.1 A Task Set Example 682
  - 24.3.2 Refinement of Major Tasks 682
- 24.4 Defining a Task Network 683
- 24.5 Scheduling 684
  - 24.5.1 Timeline Charts 685
  - 24.5.2 Tracking the Schedule 686
  - 24.5.3 Tracking Progress for an OO Project 688
- 24.6 Earned Value Analysis 690
- 24.7 Summary 691
- REFERENCES 691
- PROBLEMS AND POINTS TO PONDER 692
- FURTHER READINGS AND INFORMATION SOURCES 693

**CHAPTER 25 RISK MANAGEMENT 694**

---

- 25.1 Reactive vs. Proactive Risk Strategies 695
- 25.2 Software Risks 696
- 25.3 Risk Identification 697
  - 25.3.1 Assessing Overall Project Risk 698
  - 25.3.2 Risk Components and Drivers 699
- 25.4 Risk Projection 700
  - 25.4.1 Developing a Risk Table 701
  - 25.4.2 Assessing Risk Impact 703
- 25.5 Risk Refinement 705
- 25.6 Risk Mitigation, Monitoring, and Management 705
- 25.7 The RMMM Plan 708
- 25.8 Summary 709
- REFERENCES 710
- PROBLEMS AND POINTS TO PONDER 710
- FURTHER READINGS AND INFORMATION SOURCES 711

**CHAPTER 26 QUALITY MANAGEMENT 712**

---

- 26.1 Quality Concepts 713
  - 26.1.1 Quality 714
  - 26.1.2 Quality Control 714
  - 26.1.3 Quality Assurance 715
  - 26.1.4 Cost of Quality 715
- 26.2 Software Quality Assurance 716
  - 26.2.1 Background Issues 717
  - 26.2.2 SQA Activities 717
- 26.3 Software Reviews 719
  - 26.3.1 Cost Impact of Software Defects 720
  - 26.3.2 Defect Amplification and Removal 720
- 26.4 Formal Technical Reviews 722
  - 26.4.1 The Review Meeting 722
  - 26.4.2 Review Reporting and Record Keeping 723

26.4.3	Review Guidelines	724
26.4.4	Sample-Driven Reviews	725
26.5	Formal Approaches to SQA	727
26.6	Statistical Software Quality Assurance	727
26.6.1	A Generic Example	728
26.6.2	Six Sigma for Software Engineering	729
26.7	Software Reliability	730
26.7.1	Measures of Reliability and Availability	731
26.7.2	Software Safety	731
26.8	The ISO 9000 Quality Standards	733
26.9	The SQA Plan	734
26.10	Summary	735
	REFERENCES	736
	PROBLEMS AND POINTS TO PONDER	737
	FURTHER READINGS AND INFORMATION SOURCES	737

## **CHAPTER 27 CHANGE MANAGEMENT 739**

---

27.1	Software Configuration Management	740
27.1.1	A SCM Scenario	741
27.1.2	Elements of a Configuration Management System	742
27.1.3	Baselines	743
27.1.4	Software Configuration Items	743
27.2	The SCM Repository	745
27.2.1	The Role of the Repository	745
27.2.2	General Features and Content	746
27.2.3	SCM Features	747
27.3	The SCM Process	748
27.3.1	Identification of Objects in the Software Configuration	749
27.3.2	Version Control	750
27.3.3	Change Control	752
27.3.4	Configuration Audit	755
27.3.5	Status Reporting	756
27.4	Configuration Management for Web Engineering	756
27.4.1	Configuration Management Issues for WebApps	757
27.4.2	WebApp Configuration Objects	758
27.4.3	Content Management	758
27.4.4	Change Management	761
27.4.5	Version Control	763
27.4.6	Auditing and Reporting	764
27.5	Summary	765
	REFERENCES	766
	PROBLEMS AND POINTS TO PONDER	767
	FURTHER READINGS AND INFORMATION SOURCES	768

## **PART FIVE—ADVANCED TOPICS IN SOFTWARE ENGINEERING 769**

### **CHAPTER 28 FORMAL METHODS 770**

---

28.1	Basic Concepts	771
28.1.1	Deficiencies of Less Formal Approaches	772
28.1.2	Mathematics in Software Development	773
28.1.3	Formal Methods Concepts	773

28.2	Mathematical Preliminaries	776
28.2.1	Sets and Constructive Specification	776
28.2.2	Set Operators	778
28.2.3	Logic Operators	780
28.2.4	Sequences	780
28.3	Applying Mathematical Notation for Formal Specification	781
28.4	Formal Specification Languages	783
28.5	Object Constraint Language (OCL)	784
28.5.1	A Brief Overview of OCL Syntax and Semantics	784
28.5.2	An Example Using OCL	786
28.6	The Z Specification Language	788
28.6.1	A Brief Overview of Z Syntax and Semantics	788
28.6.2	An Example Using Z	788
28.7	The Ten Commandments of Formal Methods	791
28.8	Formal Methods—The Road Ahead	792
28.9	Summary	793
	REFERENCES	793
	PROBLEMS AND POINTS TO PONDER	794
	FURTHER READINGS AND INFORMATION SOURCES	795

---

## CHAPTER 29 CLEANROOM SOFTWARE ENGINEERING 796

---

29.1	The Cleanroom Approach	797
29.1.1	The Cleanroom Strategy	798
29.1.2	What Makes Cleanroom Different?	800
29.2	Functional Specification	801
29.2.1	Black-Box Specification	802
29.2.2	State-Box Specification	803
29.2.3	Clear-Box Specification	803
29.3	Cleanroom Design	804
29.3.1	Design Refinement and Verification	804
29.3.2	Advantages of Design Verification	808
29.4	Cleanroom Testing	809
29.4.1	Statistical Use Testing	810
29.4.2	Certification	811
29.5	Summary	812
	REFERENCES	812
	PROBLEMS AND POINTS TO PONDER	813
	FURTHER READINGS AND INFORMATION SOURCES	814

---

## CHAPTER 30 COMPONENT-BASED SOFTWARE ENGINEERING 815

---

30.1	Engineering of Component-Based Systems	816
30.2	The CBSE Process	818
30.3	Domain Engineering	819
30.3.1	The Domain Analysis Process	819
30.3.2	Characterization Functions	820
30.3.3	Structural Modeling and Structure Points	821
30.4	Component-Based Development	822
30.4.1	Component Qualification, Adaptation, and Composition	822
30.4.2	Component Engineering	825
30.4.3	Analysis and Design for Reuse	826

30.5	Classifying and Retrieving Components	827
30.5.1	Describing Reusable Components	827
30.5.2	The Reuse Environment	829
30.6	Economics of CBSE	830
30.6.1	Impact on Quality, Productivity, and Cost	830
30.6.2	Cost Analysis Using Structure Points	831
30.7	Summary	832
	REFERENCES	833
	PROBLEMS AND POINTS TO PONDER	834
	FURTHER READINGS AND INFORMATION SOURCES	835

### **CHAPTER 31 REENGINEERING 837**

---

31.1	Business Process Reengineering	838
31.1.1	Business Processes	839
31.1.2	A BPR Model	839
31.2	Software Reengineering	841
31.2.1	Software Maintenance	841
31.2.2	A Software Reengineering Process Model	842
31.3	Reverse Engineering	846
31.3.1	Reverse Engineering to Understand Data	848
31.3.2	Reverse Engineering to Understand Processing	848
31.3.3	Reverse Engineering User Interfaces	849
31.4	Restructuring	850
31.4.1	Code Restructuring	850
31.4.2	Data Restructuring	851
31.5	Forward Engineering	852
31.5.1	Forward Engineering for Client/Server Architectures	853
31.5.2	Forward Engineering for Object-Oriented Architectures	854
31.5.3	Forward Engineering User Interfaces	855
31.6	The Economics of Reengineering	855
31.7	Summary	856
	REFERENCES	857
	PROBLEMS AND POINTS TO PONDER	858
	FURTHER READINGS AND INFORMATION SOURCES	859

### **CHAPTER 32 THE ROAD AHEAD 860**

---

32.1	The Importance of Software—Revisited	861
32.2	The Scope of Change	861
32.3	People and the Way They Build Systems	863
32.4	The "New" Software Engineering Process	864
32.5	New Modes for Representing Information	865
32.6	Technology as a Driver	867
32.7	The Software Engineer's Responsibility	868
32.8	A Concluding Comment	870
	REFERENCES	871
	PROBLEMS AND POINTS TO PONDER	871
	FURTHER READINGS AND INFORMATION SOURCES	871