

CONTENTS

Preface xv

PART I

Introduction to Software Engineering 1

Chapter 1

The Scope of Software Engineering 3

- 1.1 Historical Aspects 4
- 1.2 Economic Aspects 7
- 1.3 Maintenance Aspects 8
- 1.4 Specification and Design Aspects 13
- 1.5 Team Programming Aspects 15
- 1.6 The Object-Oriented Paradigm 17
- 1.7 Terminology 21
- Chapter Review 23
- For Further Reading 24
- Problems 25
- References 26

Chapter 2

The Software Process 30

- 2.1 Client, Developer, and User 32
- 2.2 Requirements Phase 33
 - 2.2.1 Requirements Phase Testing 34
 - 2.2.2 Requirements Phase Documentation 35
- 2.3 Specification Phase 35
 - 2.3.1 Specification Phase Testing 37
 - 2.3.2 Specification Phase Documentation 38
- 2.4 Design Phase 38
 - 2.4.1 Design Phase Testing 39
 - 2.4.2 Design Phase Documentation 40

- 2.5 Implementation Phase 40
 - 2.5.1 Implementation Phase Testing 40
 - 2.5.2 Implementation Phase Documentation 40
- 2.6 Integration Phase 41
 - 2.6.1 Integration Phase Testing 41
 - 2.6.2 Integration Phase Documentation 42
- 2.7 Maintenance Phase 42
 - 2.7.1 Maintenance Phase Testing 43
 - 2.7.2 Maintenance Phase Documentation 43
- 2.8 Retirement 43
- 2.9 Problems with Software Production: Essence and Accidents 44
 - 2.9.1 Complexity 45
 - 2.9.2 Conformity 47
 - 2.9.3 Changeability 48
 - 2.9.4 Invisibility 49
 - 2.9.5 No Silver Bullet? 50
- 2.10 Improving the Software Process 51
- 2.11 Capability Maturity Models 51
- 2.12 Other Software Process Improvement Initiatives 54
- 2.13 Costs and Benefits of Software Process Improvement 55
- Chapter Review 57
- For Further Reading 58
- Problems 59
- References 60

Chapter 3

Software Life-Cycle Models 64

- 3.1 Build-and-Fix Model 64
- 3.2 Waterfall Model 65
 - 3.2.1 Analysis of the Waterfall Model 68

3.3	Rapid Prototyping Model	70
3.3.1	Integrating the Waterfall and Rapid Prototyping Models	71
3.4	Incremental Model	72
3.4.1	Analysis of the Incremental Model	73
3.5	Extreme Programming	75
3.6	Synchronize-and-Stabilize Model	77
3.7	Spiral Model	78
3.7.1	Analysis of the Spiral Model	82
3.8	Object-Oriented Life-Cycle Models	82
3.9	Comparison of Life-Cycle Models	84
	Chapter Review	86
	For Further Reading	86
	Problems	87
	References	87

Chapter 4 **Teams 90**

4.1	Team Organization	90
4.2	Democratic Team Approach	92
4.2.1	Analysis of the Democratic Team Approach	93
4.3	Classical Chief Programmer Team Approach	93
4.3.1	The <i>New York Times</i> Project	95
4.3.2	Impracticality of the Classical Chief Programmer Team Approach	96
4.4	Beyond Chief Programmer and Democratic Teams	97
4.5	Synchronize-and-Stabilize Teams	101
4.6	Extreme Programming Teams	102
	Chapter Review	103
	For Further Reading	104
	Problems	104
	References	105

Chapter 5 **The Tools of the Trade 106**

5.1	Stepwise Refinement	106
5.1.1	Stepwise Refinement Example	107
5.2	Cost–Benefit Analysis	113
5.3	Software Metrics	114

5.4	CASE	115
5.5	Taxonomy of CASE	116
5.6	Scope of CASE	118
5.7	Software Versions	122
5.7.1	Revisions	122
5.7.2	Variations	123
5.8	Configuration Control	124
5.8.1	Configuration Control during Product Maintenance	126
5.8.2	Baselines	127
5.8.3	Configuration Control during Product Development	127
5.9	Build Tools	128
5.10	Productivity Gains with CASE Technology	129
	Chapter Review	131
	For Further Reading	131
	Problems	132
	References	133

Chapter 6 **Testing 136**

6.1	Quality Issues	137
6.1.1	Software Quality Assurance	137
6.1.2	Managerial Independence	138
6.2	Nonexecution-Based Testing	139
6.2.1	Walkthroughs	139
6.2.2	Managing Walkthroughs	140
6.2.3	Inspections	141
6.2.4	Comparison of Inspections and Walkthroughs	143
6.2.5	Strengths and Weaknesses of Reviews	144
6.2.6	Metrics for Inspections	144
6.3	Execution-Based Testing	145
6.4	What Should Be Tested?	145
6.4.1	Utility	146
6.4.2	Reliability	147
6.4.3	Robustness	147
6.4.4	Performance	148
6.4.5	Correctness	149
6.5	Testing versus Correctness Proofs	151
6.5.1	Example of a Correctness Proof	151
6.5.2	Correctness Proof Case Study	154

6.5.3	Correctness Proof and Software Engineering	155
6.6	Who Should Perform Execution-Based Testing?	158
6.7	When Testing Stops	160
	Chapter Review	160
	For Further Reading	161
	Problems	162
	References	164
Chapter 7		
From Modules to Objects 167		
7.1	What Is a Module?	167
7.2	Cohesion	171
7.2.1	Coincidental Cohesion	171
7.2.2	Logical Cohesion	172
7.2.3	Temporal Cohesion	173
7.2.4	Procedural Cohesion	174
7.2.5	Communicational Cohesion	174
7.2.6	Functional Cohesion	175
7.2.7	Informational Cohesion	175
7.2.8	Cohesion Example	176
7.3	Coupling	177
7.3.1	Content Coupling	178
7.3.2	Common Coupling	178
7.3.3	Control Coupling	180
7.3.4	Stamp Coupling	180
7.3.5	Data Coupling	182
7.3.6	Coupling Example	182
7.3.7	The Importance of Coupling	182
7.4	Data Encapsulation	184
7.4.1	Data Encapsulation and Product Development	186
7.4.2	Data Encapsulation and Product Maintenance	188
7.5	Abstract Data Types	194
7.6	Information Hiding	195
7.7	Objects	198
7.8	Inheritance, Polymorphism, and Dynamic Binding	201
7.9	Cohesion and Coupling of Objects	203
7.10	The Object-Oriented Paradigm	204
	Chapter Review	207
	For Further Reading	207
	Problems	208
	References	209

Chapter 8	
Reusability, Portability, and Interoperability 212	
8.1	Reuse Concepts 212
8.2	Impediments to Reuse 214
8.3	Reuse Case Studies 216
8.3.1	Raytheon Missile Systems Division 216
8.3.2	Toshiba Software Factory 217
8.3.3	NASA Software 218
8.3.4	GTE Data Services 219
8.3.5	Hewlett-Packard 220
8.3.6	European Space Agency 221
8.4	Objects and Reuse 222
8.5	Reuse during the Design and Implementation Phases 222
8.5.1	Design Reuse 222
8.5.2	Application Frameworks 224
8.5.3	Design Patterns 225
8.5.4	Software Architecture 229
8.6	Reuse and Maintenance 230
8.7	Portability 231
8.7.1	Hardware Incompatibilities 232
8.7.2	Operating Systems Incompatibilities 233
8.7.3	Numerical Software Incompatibilities 233
8.7.4	Compiler Incompatibilities 235
8.8	Why Portability? 239
8.9	Techniques for Achieving Portability 240
8.9.1	Portable System Software 240
8.9.2	Portable Application Software 241
8.9.3	Portable Data 242
8.10	Interoperability 243
8.10.1	COM 243
8.10.2	CORBA 244
8.10.3	Comparing COM and CORBA 245
8.11	Future Trends in Interoperability 245
	Chapter Review 246
	For Further Reading 247
	Problems 248
	References 250

Chapter 9**Planning and Estimating 257**

- 9.1 Planning and the Software Process 257
- 9.2 Estimating Duration and Cost 259
 - 9.2.1 Metrics for the Size of a Product 260
 - 9.2.2 Techniques of Cost Estimation 264
 - 9.2.3 Intermediate COCOMO 267
 - 9.2.4 COCOMO II 270
 - 9.2.5 Tracking Duration and Cost Estimates 272
- 9.3 Components of a Software Project Management Plan 272
- 9.4 Software Project Management Plan Framework 274
- 9.5 IEEE Software Project Management Plan 274
- 9.6 Planning Testing 278
- 9.7 Planning Object-Oriented Projects 279
- 9.8 Training Requirements 280
- 9.9 Documentation Standards 281
- 9.10 CASE Tools for Planning and Estimating 282
- 9.11 Testing the Software Project Management Plan 282
- Chapter Review 283
- For Further Reading 283
- Problems 284
- References 285

PART 2**The Phases of the Software Life Cycle 289****Chapter 10****Requirements Phase 290**

- 10.1 Requirements Elicitation 291
 - 10.1.1 Interviews 291
 - 10.1.2 Scenarios 292

- 10.1.3 Other Requirements Elicitation Techniques 293

- 10.2 Requirements Analysis 294
- 10.3 Rapid Prototyping 294
- 10.4 Human Factors 296
- 10.5 Rapid Prototyping as a Specification Technique 298
- 10.6 Reusing the Rapid Prototype 300
- 10.7 Management Implications of the Rapid Prototyping Model 302
- 10.8 Experiences with Rapid Prototyping 304
- 10.9 Techniques for Requirements Elicitation and Analysis 305
- 10.10 Testing during the Requirements Phase 305
- 10.11 CASE Tools for the Requirements Phase 306
- 10.12 Metrics for the Requirements Phase 307
- 10.13 Object-Oriented Requirements? 308
- 10.14 Air Gourmet Case Study: Requirements Phase 308
- 10.15 Air Gourmet Case Study: Rapid Prototype 311
- 10.16 Challenges of the Requirements Phase 313
- Chapter Review 315
- For Further Reading 315
- Problems 316
- References 317

Chapter 11**Specification Phase 319**

- 11.1 The Specification Document 319
- 11.2 Informal Specifications 321
 - 11.2.1 Case Study: Text Processing 322
- 11.3 Structured Systems Analysis 323
 - 11.3.1 Sally's Software Shop 323
- 11.4 Other Semiformal Techniques 331
- 11.5 Entity-Relationship Modeling 332
- 11.6 Finite State Machines 335
 - 11.6.1 Elevator Problem: Finite State Machines 336

11.7	Petri Nets	341
11.7.1	Elevator Problem: Petri Nets	343
11.8	Z	346
11.8.1	Elevator Problem: Z	347
11.8.2	Analysis of Z	349
11.9	Other Formal Techniques	351
11.10	Comparison of Specification Techniques	352
11.11	Testing during the Specification Phase	353
11.12	CASE Tools for the Specification Phase	354
11.13	Metrics for the Specification Phase	355
11.14	Air Gourmet Case Study: Structured Systems Analysis	355
11.15	Air Gourmet Case Study: Software Project Management Plan	357
11.16	Challenges of the Specification Phase	358
	Chapter Review	358
	For Further Reading	359
	Problems	360
	References	362

Chapter 12 **Object-Oriented** **Analysis Phase 366**

12.1	Object-Oriented Analysis	366
12.2	Elevator Problem: Object-Oriented Analysis	369
12.3	Use-Case Modeling	369
12.4	Class Modeling	371
12.4.1	Noun Extraction	372
12.4.2	CRC Cards	374
12.5	Dynamic Modeling	375
12.6	Testing during the Object-Oriented Analysis Phase	378
12.7	CASE Tools for the Object-Oriented Analysis Phase	383
12.8	Air Gourmet Case Study: Object-Oriented Analysis	383

12.9	Challenges of the Object-Oriented Analysis Phase	390
	Chapter Review	391
	For Further Reading	391
	Problems	392
	References	393

Chapter 13 **Design Phase 395**

13.1	Design and Abstraction	395
13.2	Action-Oriented Design	396
13.3	Data Flow Analysis	397
13.3.1	Data Flow Analysis Example	398
13.3.2	Extensions	402
13.4	Transaction Analysis	403
13.5	Data-Oriented Design	406
13.6	Object-Oriented Design	406
13.7	Elevator Problem: Object-Oriented Design	407
13.8	Formal Techniques for Detailed Design	415
13.9	Real-Time Design Techniques	416
13.10	Testing during the Design Phase	418
13.11	CASE Tools for the Design Phase	418
13.12	Metrics for the Design Phase	419
13.13	Air Gourmet Case Study: Object-Oriented Design	420
13.14	Challenges of the Design Phase	429
	Chapter Review	429
	For Further Reading	430
	Problems	431
	References	431

Chapter 14 **Implementation Phase 434**

14.1	Choice of Programming Language	434
14.2	Fourth-Generation Languages	437
14.3	Good Programming Practice	440
14.4	Coding Standards	445
14.5	Module Reuse	446

14.6	Module Test Case Selection	447	15.1.2	Bottom-up Implementation and Integration	477
14.6.1	Testing to Specifications versus Testing to Code	447	15.1.3	Sandwich Implementation and Integration	478
14.6.2	Feasibility of Testing to Specifications	447	15.1.4	Implementation and Integration of Object-Oriented Products	480
14.6.3	Feasibility of Testing to Code	448	15.1.5	Management Issues during the Implementation and Integration Phase	480
14.7	Black-Box Module-Testing Techniques	451	15.2	Testing during the Implementation and Integration Phase	481
14.7.1	Equivalence Testing and Boundary Value Analysis	451	15.3	Integration Testing of Graphical User Interfaces	481
14.7.2	Functional Testing	452	15.4	Product Testing	482
14.8	Glass-Box Module-Testing Techniques	454	15.5	Acceptance Testing	483
14.8.1	Structural Testing: Statement, Branch, and Path Coverage	454	15.6	CASE Tools for the Implementation and Integration Phase	484
14.8.2	Complexity Metrics	456	15.6	CASE Tools for the Complete Software Process	484
14.9	Code Walkthroughs and Inspections	458	15.8	Integrated Environments	485
14.10	Comparison of Module-Testing Techniques	458	15.9	Environments for Business Applications	486
14.11	Cleanroom	459	15.10	Public Tool Infrastructures	487
14.12	Potential Problems When Testing Objects	460	15.11	Potential Problems with Environments	487
14.13	Management Aspects of Module Testing	463	15.12	Metrics for the Implementation and Integration Phase	488
14.14	When to Rewrite Rather than Debug a Module	463	15.13	Air Gourmet Case Study: Implementation and Integration Phase	488
14.15	CASE Tools for the Implementation Phase	465	15.14	Challenges of the Implementation and Integration Phase	489
14.16	Air Gourmet Case Study: Black-Box Test Cases	465	Chapter Review	489	
14.17	Challenges of the Implementation Phase	467	For Further Reading	490	
Chapter Review	467		Problems	490	
For Further Reading	468		References	492	
Problems	469				
References	470				

Chapter 15

Implementation and Integration Phase 474

15.1	Introduction to Implementation and Integration	474
15.1.1	Top-down Implementation and Integration	475

Chapter 16

Maintenance Phase 493

16.1	Why Maintenance Is Necessary	493
16.2	What Is Required of Maintenance Programmers	494
16.3	Maintenance Case Study	497
16.4	Management of Maintenance	498
16.4.1	Fault Reports	498
16.4.2	Authorizing Changes to the Product	499

16.4.3	Ensuring Maintainability	500
16.4.4	Problem of Repeated Maintenance	500
16.5	Maintenance of Object-Oriented Software	501
16.6	Maintenance Skills versus Development Skills	504
16.7	Reverse Engineering	505
16.8	Testing during the Maintenance Phase	506
16.9	CASE Tools for the Maintenance Phase	507
16.10	Metrics for the Maintenance Phase	507
16.11	Air Gourmet Case Study: Maintenance Phase	508
16.12	Challenges of the Maintenance Phase	508
	Chapter Review	509
	For Further Reading	509
	Problems	510
	References	511

Appendix A

**Broadlands Area Children's
Hospital 513**

Appendix B

**Software Engineering
Resources 518**

Appendix C

**Air Gourmet Case Study:
C Rapid Prototype 520**

Appendix D

**Air Gourmet Case Study:
Java Rapid Prototype 521**

Appendix E

**Air Gourmet Case Study:
Structured Systems Analysis 522**

Appendix F

**Air Gourmet Case Study:
Software Project Management
Plan 529**

Appendix G

**Air Gourmet Case Study:
Object-Oriented Analysis 534**

Appendix H

**Air Gourmet Case Study:
Design for C++ Implementation 535**

Appendix I

**Air Gourmet Case Study:
Design for Java Implementation 560**

Appendix J

**Air Gourmet Case Study:
Black-Box Test Cases 582**

Appendix K

**Air Gourmet Case Study:
C++ Source Code 588**

Appendix L

**Air Gourmet Case Study:
Java Source Code 589**

Bibliography 590

Author Index 617

Subject Index 623