

# SOFTWARE ENGINEERING WITH JAVA

---

**Stephen R. Schach**  
*Vanderbilt University*

 **Irwin  
McGraw-Hill**

Boston, Massachusetts Burr Ridge, Illinois Dubuque, Iowa  
Madison, Wisconsin New York, New York San Francisco, California St. Louis, Missouri

# CONTENTS

## Prologue 1

### PART 1

## Introduction to the Software Process 3

---

### CHAPTER 1

## Scope of Software Engineering 5

- 1.1 Historical Aspects 6
- 1.2 Economic Aspects 9
- 1.3 Maintenance Aspects 10
- 1.4 Specification and Design Aspects 14
- 1.5 Team Programming Aspects 16
- 1.6 The Object-Oriented Paradigm 17
- 1.7 Terminology 22
- Chapter Review 24
- For Further Reading 25
- Problems 26
- References 27

### CHAPTER 2

## The Software Process and Its Problems 30

- 2.1 Client, Developer, and User 32
- 2.2 Requirements Phase 33
  - 2.2.1 Requirements Phase Testing 34
- 2.3 Specification Phase 35
  - 2.3.1 Specification Phase Testing 36
- 2.4 Planning Phase 36
  - 2.4.1 Planning Phase Testing 37
- 2.5 Design Phase 38
  - 2.5.1 Design Phase Testing 39
- 2.6 Implementation Phase 39
  - 2.6.1 Implementation Phase Testing 39
- 2.7 Integration Phase 40
  - 2.7.1 Integration Phase Testing 40
- 2.8 Maintenance Phase 41
  - 2.8.1 Maintenance Phase Testing 42

- 2.9 Retirement 42
- 2.10 Problems with Software Production:
  - Essence and Accidents 43
  - 2.10.1 Complexity 44
  - 2.10.2 Conformity 46
  - 2.10.3 Changeability 47
  - 2.10.4 Invisibility 48
  - 2.10.5 No Silver Bullet? 49
- Chapter Review 50
- For Further Reading 50
- Problems 51
- References 52

### CHAPTER 3

## Software Life-Cycle Models 53

- 3.1 Build-and-Fix Model 53
- 3.2 Waterfall Model 54
  - 3.2.1 Analysis of the Waterfall Model 57
- 3.3 Rapid Prototyping Model 59
  - 3.3.1 Integrating the Waterfall and Rapid Prototyping Models 61
- 3.4 Incremental Model 61
  - 3.4.1 Analysis of the Incremental Model 63
- 3.5 Spiral Model 66
  - 3.5.1 Analysis of the Spiral Model 70
- 3.6 Comparison of Life-Cycle Models 71
- 3.7 Capability Maturity Model 71
- 3.8 ISO 9000 75
- Chapter Review 76
- For Further Reading 77
- Problems 78
- References 78

### CHAPTER 4

## Stepwise Refinement, CASE, and Other Tools of the Trade 82

- 4.1 Stepwise Refinement 82
  - 4.1.1 Stepwise Refinement Example 83

4.2	Cost–Benefit Analysis	89
4.3	CASE (Computer-Aided Software Engineering)	90
4.3.1	Taxonomy of CASE	90
4.4	Scope of CASE	92
4.5	Software Versions	96
4.5.1	Revisions	96
4.5.2	Variations	97
4.6	Configuration Control	98
4.6.1	Configuration Control during Product Maintenance	100
4.6.2	Baselines	101
4.6.3	Configuration Control during Product Development	101
4.7	Build Tools	102
4.8	Productivity Gains with CASE Technology	103
4.9	Software Metrics	103
	Chapter Review	105
	For Further Reading	105
	Problems	106
	References	108

## **CHAPTER 5** **Testing Principles 110**

5.1	Quality Issues	111
5.1.1	Software Quality Assurance	111
5.1.2	Managerial Independence	112
5.2	Nonexecution-Based Testing	113
5.2.1	Walkthroughs	113
5.2.2	Managing Walkthroughs	114
5.2.3	Inspections	115
5.2.4	Comparison of Inspections and Walkthroughs	117
5.2.5	Metrics for Inspections	118
5.3	Execution-Based Testing	118
5.4	What Should Be Tested?	119
5.4.1	Utility	120
5.4.2	Reliability	120
5.4.3	Robustness	121
5.4.4	Performance	121
5.4.5	Correctness	122
5.5	Testing versus Correctness Proofs	124
5.5.1	Example of a Correctness Proof	124
5.5.2	Correctness Proof Case Study	128

5.5.3	Correctness Proofs and Software Engineering	129
5.6	Who Should Perform Execution-Based Testing?	131
5.7	When Testing Stops	133
	Chapter Review	134
	For Further Reading	134
	Problems	135
	References	137

## **CHAPTER 6** **Introduction to Objects 140**

6.1	What Is a Module?	140
6.2	Cohesion	144
6.2.1	Coincidental Cohesion	145
6.2.2	Logical Cohesion	145
6.2.3	Temporal Cohesion	146
6.2.4	Procedural Cohesion	147
6.2.5	Communicational Cohesion	148
6.2.6	Informational Cohesion	148
6.2.7	Functional Cohesion	149
6.2.8	Cohesion Example	150
6.3	Coupling	151
6.3.1	Content Coupling	151
6.3.2	Common Coupling	151
6.3.3	Control Coupling	154
6.3.4	Stamp Coupling	154
6.3.5	Data Coupling	155
6.3.6	Coupling Example	156
6.4	Data Encapsulation	157
6.4.1	Data Encapsulation and Product Development	161
6.4.2	Data Encapsulation and Product Maintenance	163
6.5	Abstract Data Types	166
6.6	Information Hiding	168
6.7	Objects	171
6.8	Polymorphism and Dynamic Binding	175
6.9	Cohesion and Coupling of Objects	177
6.10	Reuse	178
6.10.1	Impediments to Reuse	179
6.11	Reuse Case Studies	180
6.11.1	Raytheon Missile Systems Division	180
6.11.2	Toshiba Software Factory	182
6.11.3	NASA Software	183

6.11.4	GTE Data Services	184
6.11.5	Hewlett-Packard	184
6.12	Reuse and Maintenance	185
6.13	Objects and Productivity	186
	Chapter Review	188
	For Further Reading	188
	Problems	189
	References	191

## **PART 2**

### **The Phases of the Software Process 195**

---

#### **CHAPTER 7**

### **Requirements Phase 197**

7.1	Requirements Analysis Techniques	198
7.2	Rapid Prototyping	199
7.3	Human Factors	201
7.4	Rapid Prototyping as a Specification Technique	203
7.5	Reusing the Rapid Prototype	205
7.6	Other Uses of Rapid Prototyping	207
7.7	Management Implications of the Rapid Prototyping Model	208
7.8	Experiences with Rapid Prototyping	209
7.9	Joint Application Design	211
7.10	Comparison of Requirements Analysis Techniques	211
7.11	Testing during the Requirements Phase	212
7.12	CASE Tools for the Requirements Phase	212
7.13	Metrics for the Requirements Phase	213
7.14	MSG Case Study: Requirements Phase	214
7.15	MSG Case Study: Rapid Prototype	216
	Chapter Review	217
	For Further Reading	218
	Problems	219
	References	220

#### **CHAPTER 8**

### **Specification Phase 222**

8.1	The Specification Document	222
-----	----------------------------	-----

8.2	Informal Specifications	224
8.2.1	Case Study: Text Processing	225
8.3	Structured Systems Analysis	226
8.3.1	Sally's Software Shop	226
8.4	Other Semiformal Techniques	234
8.5	Entity-Relationship Modeling	235
8.6	Finite State Machines	237
8.6.1	Elevator Problem: Finite State Machines	239
8.7	Petri Nets	244
8.7.1	Elevator Problem: Petri Nets	247
8.8	Z	250
8.8.1	Elevator Problem: Z	251
8.8.2	Analysis of Z	253
8.9	Other Formal Techniques	255
8.10	Comparison of Specification Techniques	256
8.11	Testing during the Specification Phase	256
8.12	CASE Tools for the Specification Phase	257
8.13	Metrics for the Specification Phase	258
8.14	MSG Case Study: Structured Systems Analysis	258
	Chapter Review	260
	For Further Reading	261
	Problems	262
	References	264

#### **CHAPTER 9**

### **Object-Oriented Analysis Phase 268**

9.1	Object-Oriented versus Structured Paradigm	268
9.2	Object-Oriented Analysis	270
9.3	Elevator Problem: Object-Oriented Analysis	272
9.3.1	Class Modeling	272
9.3.2	Dynamic Modeling	275
9.3.3	Functional Modeling	278
9.4	Object-Oriented Life-Cycle Models	280
9.5	CASE Tools for the Object-Oriented Analysis Phase	282
9.6	MSG Case Study: Object-Oriented Analysis	283

Chapter Review 286  
 For Further Reading 286  
 Problems 288  
 References 289

## **CHAPTER 10** **Planning Phase 291**

10.1 Estimating Duration and Cost 291  
 10.1.1 Metrics for the Size of a Product 293  
 10.1.2 Techniques of Cost Estimation 297  
 10.1.3 Intermediate COCOMO 299  
 10.1.4 Tracking Duration and Cost Estimates 303  
 10.2 Components of a Software Project Management Plan 303  
 10.3 Software Project Management Plan Framework 305  
 10.4 IEEE Software Project Management Plan 305  
 10.5 Planning of Testing 308  
 10.6 Planning of Object-Oriented Projects 310  
 10.7 Training Requirements 310  
 10.8 Documentation Standards 311  
 10.9 CASE Tools for the Planning Phase 312  
 10.10 Testing during the Planning Phase 315  
 10.11 MSG Case Study: Planning Phase 315  
 Chapter Review 315  
 For Further Reading 316  
 Problems 317  
 References 318

## **CHAPTER 11** **Design Phase 322**

11.1 Design and Abstraction 322  
 11.2 Action-Oriented Design 324  
 11.3 Data Flow Analysis 324  
 11.3.1 Data Flow Analysis Example 325  
 11.3.2 Extensions 329  
 11.4 Transaction Analysis 329  
 11.5 Data-Oriented Design 332  
 11.6 Jackson System Development 333  
 11.6.1 Overview of Jackson System Development 333

11.6.2 Why Jackson System Development Is Presented in This Chapter 335  
 11.6.3 Elevator Problem: Jackson System Development 336  
 11.6.4 Analysis of Jackson System Development 344  
 11.7 Techniques of Jackson, Warnier, and Orr 345  
 11.8 Object-Oriented Design 346  
 11.8.1 Elevator Problem: Object-Oriented Design 347  
 11.9 Detailed Design 350  
 11.10 Comparison of Action-, Data-, and Object-Oriented Design 352  
 11.11 Difficulties Associated with Real-Time Systems 353  
 11.12 Real-Time Design Techniques 354  
 11.13 Testing during the Design Phase 355  
 11.14 CASE Tools for the Design Phase 356  
 11.15 Metrics for the Design Phase 357  
 11.16 MSG Case Study: Object-Oriented Design 358  
 Chapter Review 359  
 For Further Reading 361  
 Problems 363  
 References 364

## **CHAPTER 12** **Implementation Phase 368**

12.1 Choice of Programming Language 368  
 12.2 Fourth Generation Languages 372  
 12.3 Structured Programming 375  
 12.3.1 History of Structured Programming 375  
 12.3.2 Why the `goto` Statement Is Considered Harmful 377  
 12.4 Good Programming Practice 378  
 12.5 Coding Standards 383  
 12.6 Team Organization 385  
 12.7 Democratic Team Approach 387  
 12.7.1 Analysis of the Democratic Team Approach 388  
 12.8 Classical Chief Programmer Team Approach 388  
 12.8.1 The *New York Times* Project 390

- 12.8.2 Impracticality of the Classical Chief Programmer Team Approach 391
  - 12.9 Beyond Chief Programmer and Democratic Teams 392
  - 12.10 Portability 396
    - 12.10.1 Hardware Incompatibilities 396
    - 12.10.2 Operating System Incompatibilities 398
    - 12.10.3 Numerical Software Incompatibilities 398
    - 12.10.4 Compiler Incompatibilities 399
  - 12.11 Why Portability? 402
  - 12.12 Techniques for Achieving Portability 404
    - 12.12.1 Portable System Software 404
    - 12.12.2 Portable Application Software 405
    - 12.12.3 Portable Data 406
  - 12.13 Module Reuse 407
  - 12.14 Module Test Case Selection 407
    - 12.14.1 Testing to Specifications versus Testing to Code 408
    - 12.14.2 Feasibility of Testing to Specifications 408
    - 12.14.3 Feasibility of Testing to Code 409
  - 12.15 Black-Box Module-Testing Techniques 411
    - 12.15.1 Equivalence Testing and Boundary Value Analysis 411
    - 12.15.2 Functional Testing 413
  - 12.16 Glass-Box Module-Testing Techniques 414
    - 12.16.1 Structural Testing: Statement, Branch, and Path Coverage 414
    - 12.16.2 Complexity Metrics 415
  - 12.17 Code Walkthroughs and Inspections 418
  - 12.18 Comparison of Module-Testing Techniques 418
  - 12.19 Cleanroom 419
  - 12.20 Testing Objects 420
  - 12.21 Management Aspects of Module-Testing 423
    - 12.21.1 When to Rewrite Rather Than Debug a Module 424
  - 12.22 Testing Distributed Software 425
  - 12.23 Testing Real-Time Software 427
  - 12.24 CASE Tools for the Implementation Phase 429
  - 12.25 MSG Case Study: Black-Box Test Cases 429
  - Chapter Review 431
  - For Further Reading 431
  - Problems 433
  - References 435
- CHAPTER 13**  
**Implementation and Integration Phase 441**
- 13.1 Implementation and Integration 441
    - 13.1.1 Top-Down Implementation and Integration 442
    - 13.1.2 Bottom-Up Implementation and Integration 444
    - 13.1.3 Sandwich Implementation and Integration 445
    - 13.1.4 Implementation and Integration of Object-Oriented Products 446
    - 13.1.5 Management Issues during the Implementation and Integration Phase 446
  - 13.2 Testing during the Implementation and Integration Phase 447
  - 13.3 Integration Testing of Graphical User Interfaces 447
  - 13.4 Product Testing 448
  - 13.5 Acceptance Testing 449
  - 13.6 CASE Tools for the Implementation and Integration Phase 450
  - 13.7 CASE Tools for the Complete Software Process 451
  - 13.8 Language-Centered Environments 451
  - 13.9 Structure-Oriented Environments 452
  - 13.10 Toolkit Environments 452
  - 13.11 Integrated Environments 452
    - 13.11.1 Process Integration 453
    - 13.11.2 Tool Integration 454
    - 13.11.3 Other Forms of Integration 456
  - 13.12 Environments for Business Applications 456
  - 13.13 Public Tool Infrastructures 457
  - 13.14 Comparison of Environment Types 458
  - 13.15 Metrics for the Implementation and Integration Phase 458

- 13.16 MSG Case Study: Implementation and Integration Phase 459
- Chapter Review 460
- For Further Reading 460
- Problems 461
- References 462

## **CHAPTER 14**

### **Maintenance Phase 465**

- 14.1 Why Maintenance Is Necessary 465
- 14.2 What Is Required of Maintenance Programmers 466
- 14.3 Maintenance Case Study 468
- 14.4 Management of Maintenance 470
  - 14.4.1 Fault Reports 470
  - 14.4.2 Authorizing Changes to the Product 471
  - 14.4.3 Ensuring Maintainability 472
  - 14.4.4 Problem of Repeated Maintenance 472
- 14.5 Maintenance of Object-Oriented Software 473
- 14.6 Maintenance Skills versus Development Skills 476
- 14.7 Reverse Engineering 476
- 14.8 Testing during the Maintenance Phase 477
- 14.9 CASE Tools for the Maintenance Phase 478
- 14.10 Metrics for the Maintenance Phase 479
- Chapter Review 479
- For Further Reading 480
- Problems 480
- References 481

### **Epilogue 483**

## **Appendices**

---

### **APPENDIX A**

**Osbert Oglesby—Art Dealer 491**

### **APPENDIX B**

**Software Engineering Resources 494**

### **APPENDIX C**

**MSG Case Study: Rapid Prototype 496**

### **APPENDIX D**

**MSG Case Study: Structured Systems Analysis 509**

### **APPENDIX E**

**MSG Case Study: Object-Oriented Analysis 513**

### **APPENDIX F**

**MSG Case Study: Software Project Management Plan 514**

### **APPENDIX G**

**MSG Case Study: Design 519**

### **APPENDIX H**

**MSG Case Study: Black-Box Test Cases 539**

### **APPENDIX I**

**MSG Case Study: Source Code 542**

**Bibliography 581**

**Author Index 605**

**Subject Index 608**