

Contents

Preface to the Third Edition	(v)
1. INTRODUCTION TO SOFTWARE ENGINEERING	1-19
1.1 The Envolving Role of Software.....	1
1.1.1 Some Software Failures.....	1
1.1.2 No Software Bullet.....	3
1.2 What is Software Engineering?	4
1.2.1 Definition.....	4
1.2.2 Program Versus Software.....	4
1.2.3 Software Process	6
1.2.4 Software Characteristics	8
1.3 The Changing Nature of Software.....	10
1.4 Software Myths	11
1.5 Some Terminologies	12
1.5.1 Deliverables and Milestones	12
1.5.2 Product and Process	13
1.5.3 Measures, Metrics and Measurement	13
1.5.4 Software Process and Product Metrics	13
1.5.5 Productivity and Effort.....	14
1.5.6 Module and Software Components	14
1.5.7 Generic and Customised Software Products	14
1.6 Role of Management in Software Development	15
1.6.1 The People	15
1.6.2 The Product	15
1.6.3 The Process	15
1.6.4 The Project	16
References	16
Multiple Choice Questions	17
Exercise	19
2. SOFTWARE LIFE CYCLE MODELS	20-39
2.1 Build and Fix Model	20
2.2 The Water Fall Model	21

2.3	The Increment Process Model	23
2.3.1	<i>Iterative Enhancement Model</i>	23
2.3.2	<i>The Rapid Application Development (RAD) Model</i>	24
2.4	Evolutionary Process Models	25
2.4.1	<i>Prototyping Model</i>	26
2.4.2	<i>Spiral Model</i>	26
2.5	The Unified Process	27
2.5.1	<i>The Phases of Unified Process</i>	28
2.5.2	<i>Unified Process Work Products</i>	29
2.6	Selection of a Life Cycle Model	31
2.6.1	<i>Characteristics of Requirements</i>	34
2.6.2	<i>Status of Development Team</i>	35
2.6.3	<i>Involvement of Users</i>	35
2.6.4	<i>Type of Project and Associated Risk</i>	35
	References	36
	Multiple Choice Questions	37
	Exercise	38

3. SOFTWARE REQUIREMENTS: ANALYSIS AND SPECIFICATIONS

40-138

3.1	Requirements Engineering	40
3.1.1	<i>Crucial Process Steps</i>	40
3.1.2	<i>Present State of Practice</i>	42
3.2	Type of Requirements	45
3.2.1	<i>Functional and Non-functional Requirements</i>	45
3.2.2	<i>User and System Requirements</i>	46
3.2.3	<i>Interface Specification</i>	46
3.3	Feasibility Studies	47
3.3.1	<i>Purpose of Feasibility Studies</i>	47
3.3.2	<i>Focus of Feasibility Studies</i>	48
3.4	Requirements Elicitation	48
3.4.1	<i>Interviews</i>	49
3.4.2	<i>Brainstorming Sessions</i>	51
3.4.3	<i>Facilitated Application Specification Technique</i>	51
3.4.4	<i>Quality Function Deployment</i>	53
3.4.5	<i>The Use Case Approach</i>	54
3.5	Requirements Analysis	58
3.5.1	<i>Data Flow Diagrams</i>	60
3.5.2	<i>Data Dictionaries</i>	62
3.5.3	<i>Entity-Relationship Diagrams</i>	63
3.5.4	<i>Software Prototyping</i>	69

3.6	Requirements Documentation	72
	3.4.1 <i>Nature of the SRS</i>	72
	3.4.2 <i>Characteristics of a Good SRS</i>	73
	3.4.3 <i>Organization of the SRS</i>	75
3.7	Requirements Validation	91
	3.7.1 <i>Requirements Reviews</i>	92
3.8	Requirements Management	94
	3.8.1 <i>Enduring and Volatile Requirement</i>	94
	3.8.2 <i>Requirements Management Planning</i>	95
	3.8.3 <i>Requirements Change Management</i>	95
3.9	Student Result Management System—Example	97
	3.9.1 <i>Problem Statement</i>	97
	3.9.2 <i>Context Diagram</i>	98
	3.9.3 <i>Level-n DFD</i>	99
	3.9.4 <i>Entity Relationship Diagram</i>	102
	3.9.5 <i>Use Case Diagram</i>	103
	3.9.6 <i>Use Cases</i>	103
	3.9.7 <i>SRS Document</i>	111
	<i>References</i>	133
	<i>Multiple Choice Questions</i>	135
	<i>Exercise</i>	137

4. SOFTWARE PROJECT PLANNING

139-202

4.1	Size Estimation	140
	4.1.1 <i>Lines of Code (LOC)</i>	141
	4.1.2 <i>Function Count</i>	142
4.2	Cost Estimation	150
4.3	Models	150
	4.3.1 <i>Static, Single Variable Models</i>	150
	4.3.2 <i>Static, Multivariable Models</i>	151
4.4	The Constructive Cost Model (COCOMO)	152
	4.4.1 <i>Basic Model</i>	152
	4.4.2 <i>Intermediate Model</i>	155
	4.4.3 <i>Detailed COCOMO Model</i>	157
4.5	COCOMO II	161
	4.5.1 <i>Application Composition Estimation Model</i>	163
	4.5.2 <i>The Early Design Model</i>	167
	4.5.3 <i>Post Architecture Model</i>	173
4.6	The Putnam Resource Allocation Model	181
	4.6.1 <i>The Norden/Rayleigh Curve</i>	182
	4.6.2 <i>Difficulty Metric</i>	184

4.6.3	<i>Productivity Versus Difficulty</i>	186
4.6.4	<i>The Trade-off between Time Versus Cost</i>	187
4.6.5	<i>Development Sub-cycle</i>	188
4.7	<i>Software Risk Management</i>	194
4.7.1	<i>What is Risk?</i>	194
4.7.2	<i>Typical Software Risks</i>	195
4.7.3	<i>Risk Management Activities</i>	196
	<i>References</i>	198
	<i>Multiple Choice Questions</i>	200
	<i>Exercise</i>	

203-249

5. SOFTWARE DESIGN

5.1	<i>What is Design ?</i>	203
5.1.1	<i>Conceptual and Technical Designs</i>	204
5.1.2	<i>Objectives of Design</i>	205
5.1.3	<i>Why Design is Important?</i>	206
5.2	<i>Modularity</i>	206
5.2.1	<i>Module Coupling</i>	206
5.2.2	<i>Module Cohesion</i>	211
5.2.3	<i>Relationship between Cohesion & Coupling</i>	213
5.3	<i>Strategy of Design</i>	214
5.3.1	<i>Bottom-Up Design</i>	214
5.3.2	<i>Top-Down Design</i>	215
5.3.3	<i>Hybrid Design</i>	215
5.4	<i>Function Oriented Design</i>	216
5.4.1	<i>Design Notations</i>	217
5.4.2	<i>Functional Procedure Layers</i>	220
5.5	<i>IEEE Recommended Practice for Software Design Descriptions</i>	221
5.5.1	<i>Scope</i>	221
5.5.2	<i>References</i>	221
5.5.3	<i>Definitions</i>	221
5.5.4	<i>Purpose of an SDD</i>	221
5.5.5	<i>Design Description Information Content</i>	222
5.5.6	<i>Design Description Organisation</i>	223
5.6	<i>Object Oriented Design</i>	225
5.6.1	<i>Basic Concepts</i>	225
5.6.2	<i>Steps to Analyze and Design Object Oriented System</i>	231
5.6.3	<i>Case Study of Library Management System</i>	234
	<i>References</i>	234
	<i>Multiple Choice Questions</i>	247
	<i>Exercise</i>	248

249

6. SOFTWARE METRICS	250-307
6.1 Software Metrics: What & Why ?	251
6.1.1 <i>Definition</i>	252
6.1.2 <i>Areas of Applications</i>	253
6.1.3 <i>Problems During Implementation</i>	253
6.1.4 <i>Categories of Metrics</i>	254
6.2 Token Count	254
6.2.1 <i>Estimated Program Length</i>	256
6.2.2 <i>Potential Volume</i>	258
6.2.3 <i>Estimated Program Level/Difficulty</i>	258
6.2.4 <i>Effort & Time</i>	258
6.2.5 <i>Language Level</i>	259
6.3 Data Structure Metrics	264
6.3.1 <i>The Amount of Data</i>	265
6.3.2 <i>The Usage of Data within a Module</i>	268
6.3.3 <i>Program Weakness</i>	273
6.3.4 <i>The Sharing of Data Among Modules</i>	281
6.4 Information Flow Metrics	283
6.4.1 <i>The Basic Information Flow Model</i>	283
6.4.2 <i>A More Sophisticated Information Flow Model</i>	286
6.5 Object Oriented Metrics	287
6.5.1 <i>Size Metrics</i>	287
6.5.2 <i>Coupling Metrics</i>	289
6.5.3 <i>Cohesion Metrics</i>	291
6.5.4 <i>Inheritance Metrics</i>	292
6.6 Use-Case Oriented Metrics	295
6.6.1 <i>Counting Actors</i>	295
6.6.2 <i>Counting Use Cases</i>	295
6.7 Web Engineering Project Metrics	296
6.7.1 <i>Number of Static Web Pages</i>	296
6.7.2 <i>Number of Dynamic Web Pages</i>	296
6.7.3 <i>Number of Internal Page Links</i>	296
6.7.4 <i>Word Count</i>	297
6.7.5 <i>Web Page Similarity</i>	297
6.7.6 <i>Web Page Search and Retrieval</i>	297
6.7.7 <i>Number of Static Content Objects</i>	297
6.7.8 <i>Number of Dynamic Content Objects</i>	297
6.8 Metrics Analysis	298
6.8.1 <i>Using Statistics for Assessment</i>	298
6.8.2 <i>Problems with Metrics Data</i>	299
6.8.3 <i>The Common Pool of Data</i>	300

6.8.4 A Pattern for Successful Applications	301
References	302
Multiple Choice Questions	304
Exercise	306
7. SOFTWARE RELIABILITY	308-364
7.1 Basic Concepts	308
7.1.1 What is Software Reliability?	309
7.1.2 Software Reliability and Hardware Reliability	310
7.1.3 Failures and Faults	310
7.1.4 Environment	314
7.1.5 Uses of Reliability Studies	316
7.2 Software Quality	317
7.2.1 McCall Software Quality Model	320
7.2.2 Boehm Software Quality Model	325
7.2.3 ISO 9126	326
7.3 Software Reliability Models	328
7.3.1 Basic Execution Time Model	329
7.3.2 Logarithmic Poisson Execution Time Model	334
7.3.3 Calendar Time Component	338
7.3.4 The Jelinski-Moranda Model	343
7.3.5 The Bug Seeding Model	345
7.4 Capability Maturity Model	347
7.4.1 Maturity Levels	347
7.4.2 Key Process Areas	350
7.4.3 Common Features	351
7.5 ISO 9000	352
7.5.1 Mapping ISO 9001 to the CMM	353
7.5.2 Contrasting ISO 9001 and the CMM	356
7.5.3 Conclusion	356
References	357
Multiple Choice Questions	357
Exercise	359
8. SOFTWARE TESTING	365-458
8.1 A Strategic Approach to Software Testing	365
8.1.1 What is Testing?	365
8.1.2 Why should we Test?	366
8.1.3 Who should do the Testing?	367
8.1.4 What should we Test?	367
8.2 Some Terminologies	367
8.2.1 Error, Mistake, Bug, Fault and Failure	368
	369

8.2.2	<i>Test, Test Case and Test Suite</i>	369
8.2.3	<i>Verification and Validation</i>	370
8.2.4	<i>Alpha, Beta and Acceptance Testing</i>	370
8.3	Functional Testing	371
8.3.1	<i>Boundary Value Analysis</i>	372
8.3.2	<i>Equivalence Class Testing</i>	390
8.3.3	<i>Decision Table Based Testing</i>	395
8.3.4	<i>Cause Effect Graphing Technique</i>	402
8.3.5	<i>Special Value Testing</i>	406
8.4	Structural Testing	406
8.4.1	<i>Path Testing</i>	407
8.4.2	<i>Cyclomatic Complexity</i>	422
8.4.3	<i>Graph Matrices</i>	426
8.4.4	<i>Data Flow Testing</i>	430
8.4.5	<i>Mutation Testing</i>	430
8.5	Levels of Testing	435
8.5.1	<i>Unit Testing</i>	436
8.5.2	<i>Integration Testing</i>	437
8.5.3	<i>System Testing</i>	439
8.6	Validation Testing	440
8.7	The Art of Debugging	441
8.7.1	<i>Debugging Techniques</i>	442
8.7.2	<i>Debugging Approaches</i>	442
8.7.3	<i>Debugging Tools</i>	446
8.8	Testing Tools	446
8.8.1	<i>Static Testing Tools</i>	447
8.8.2	<i>Dynamic Testing Tools</i>	448
8.8.3	<i>Characteristics of Modern Tools</i>	450
	<i>References</i>	450
	<i>Multiple Choice Questions</i>	451
	<i>Exercise</i>	455

9. SOFTWARE MAINTENANCE

459-491

9.1	What is Software Maintenance ?	459
9.1.1	<i>Categories of Maintenance</i>	459
9.1.2	<i>Problems During Maintenance</i>	461
9.1.3	<i>Maintenance is Manageable</i>	462
9.1.4	<i>Potential Solutions to Maintenance Problems</i>	463
9.2	The Maintenance Process	464
9.2.1	<i>Program Understanding</i>	464
9.2.2	<i>Generating Particular Maintenance Proposal</i>	465
9.2.3	<i>Ripple Effect</i>	465

9.2.4	<i>Modified Program Testing</i>	465
9.2.5	<i>Maintainability</i>	465
9.3	<i>Maintenance Models</i>	466
9.3.1	<i>Quick-fix Model</i>	466
9.3.2	<i>Iterative Enhancement Model</i>	466
9.3.3	<i>Reuse Oriented Model</i>	467
9.3.4	<i>Boehm's Model</i>	467
9.3.5	<i>Taute Maintenance Model</i>	468
9.4	<i>Estimation of Maintenance Costs</i>	469
9.4.1	<i>Belady and Lehman Model</i>	470
9.4.2	<i>Boehm Model</i>	471
9.5	<i>Regression Testing</i>	473
9.5.1	<i>Development Testing Versus Regression Testing</i>	473
9.5.2	<i>Regression Test Selection</i>	474
9.5.3	<i>Selective Retest Techniques</i>	475
9.6	<i>Reverse Engineering</i>	476
9.6.1	<i>Scope and Tasks</i>	476
9.6.2	<i>Levels of Reverse Engineering</i>	477
9.6.3	<i>Reverse Engineering Tools</i>	479
9.7	<i>Software Re-engineering</i>	480
9.7.1	<i>Source Code Translation</i>	482
9.7.2	<i>Program Restructuring</i>	482
9.8	<i>Configuration Management</i>	483
9.8.1	<i>Configuration Management Activities</i>	483
9.8.2	<i>Software Versions</i>	484
9.8.3	<i>Change Control Process</i>	484
9.9	<i>Documentation</i>	485
9.9.1	<i>User Documentation</i>	485
9.9.2	<i>System Documentation</i>	486
9.9.3	<i>Other Classification Schemes</i>	486
	<i>References</i>	487
	<i>Multiple Choice Questions</i>	488
	<i>Exercise</i>	490
10.	SOFTWARE CERTIFICATION	492-507
10.1	<i>Requirement of Certification</i>	492
10.2	<i>Types of Certification</i>	493
10.2.1	<i>Certification of Persons</i>	493
10.2.2	<i>Certification of Processes</i>	494
10.2.3	<i>Certification of Products</i>	494
10.3	<i>Third Party Certification for Component Based Software Engineering</i>	495