

# Contents

Foreword	xvii
Preface	xix
<b>1 Introduction</b>	<b>1</b>
1.1 What Is Software Engineering?	5
1.2 Phases in the Development of Software	10
1.3 Maintenance or Evolution	15
1.4 From the Trenches	17
1.4.1 Ariane 5, Flight 501	18
1.4.2 Therac-25	19
1.4.3 The London Ambulance Service	21
1.4.4 Who Counts the Votes?	23
1.5 Software Engineering Ethics	24
1.6 Quo Vadis?	27
1.7 Summary	29
1.8 Further Reading	30
Exercises	31
<b>Part I Software Management</b>	<b>35</b>
<b>2 Introduction to Software Engineering Management</b>	<b>37</b>
2.1 Planning a Software Development Project	40
2.2 Controlling a Software Development Project	43
2.3 Summary	45
Exercises	46
<b>3 The Software Life Cycle Revisited</b>	<b>49</b>
3.1 The Waterfall Model	52
3.2 Agile Methods	54

3.2.1	Prototyping . . . . .	56
3.2.2	Incremental Development . . . . .	60
3.2.3	Rapid Application Development and Dynamic Systems Development Method . . . . .	62
3.2.4	Extreme Programming . . . . .	66
3.3	The Rational Unified Process (RUP) . . . . .	68
3.4	Model-Driven Architecture . . . . .	71
3.5	Intermezzo: Maintenance or Evolution . . . . .	72
3.6	Software Product Lines . . . . .	75
3.7	Process Modeling . . . . .	77
3.8	Summary . . . . .	80
3.9	Further Reading . . . . .	81
	Exercises . . . . .	82
<b>4</b>	<b>Configuration Management . . . . .</b>	<b>85</b>
4.1	Tasks and Responsibilities . . . . .	87
4.2	Configuration Management Plan . . . . .	92
4.3	Summary . . . . .	93
4.4	Further Reading . . . . .	94
	Exercises . . . . .	94
<b>5</b>	<b>People Management and Team Organization . . . . .</b>	<b>97</b>
5.1	People Management . . . . .	99
5.1.1	Coordination Mechanisms . . . . .	101
5.1.2	Management Styles . . . . .	102
5.2	Team Organization . . . . .	104
5.2.1	Hierarchical Organization . . . . .	104
5.2.2	Matrix Organization . . . . .	106
5.2.3	Chief Programmer Team . . . . .	107
5.2.4	SWAT Team . . . . .	107
5.2.5	Agile Team . . . . .	108
5.2.6	Open Source Software Development . . . . .	108
5.2.7	General Principles for Organizing a Team . . . . .	111
5.3	Summary . . . . .	112
5.4	Further Reading . . . . .	113
	Exercises . . . . .	113
<b>6</b>	<b>On Managing Software Quality . . . . .</b>	<b>115</b>
6.1	On Measures and Numbers . . . . .	118
6.2	A Taxonomy of Quality Attributes . . . . .	123
6.3	Perspectives on Quality . . . . .	130
6.4	The Quality System . . . . .	134
6.5	Software Quality Assurance . . . . .	135
6.6	The Capability Maturity Model (CMM) . . . . .	137

6.6.1	Personal Software Process .....	142
6.6.2	BOOTSTRAP and SPICE .....	143
6.6.3	Some Critical Notes .....	143
6.7	Getting Started .....	144
6.8	Summary .....	147
6.9	Further Reading .....	148
	Exercises .....	149
<b>7</b>	<b>Cost Estimation .....</b>	<b>153</b>
7.1	Algorithmic Models .....	158
7.1.1	Walston–Felix .....	160
7.1.2	COCOMO .....	162
7.1.3	Putnam .....	163
7.1.4	Function Point Analysis .....	165
7.1.5	COCOMO 2: Variations on a Theme .....	168
7.1.6	Use-Case Points: Another Variation on a Theme .....	173
7.2	Guidelines for Estimating Cost .....	175
7.3	Distribution of Manpower over Time .....	179
7.4	Agile Cost Estimation .....	183
7.5	Summary .....	184
7.6	Further Reading .....	186
	Exercises .....	187
<b>8</b>	<b>Project Planning and Control .....</b>	<b>189</b>
8.1	A Systems View of Project Control .....	190
8.2	A Taxonomy of Software Development Projects .....	192
8.2.1	Realization Control Situation .....	194
8.2.2	Allocation Control Situation .....	195
8.2.3	Design Control Situation .....	195
8.2.4	Exploration Control Situation .....	196
8.2.5	Summary of Control Situations .....	197
8.3	Risk Management .....	198
8.4	Techniques for Project Planning and Control .....	201
8.5	Summary .....	207
8.6	Further Reading .....	208
	Exercises .....	208
<b>Part II</b>	<b>The Software Life Cycle .....</b>	<b>211</b>
<b>9</b>	<b>Requirements Engineering .....</b>	<b>213</b>
9.1	Requirements Elicitation .....	220
9.1.1	Requirements Engineering Paradigms .....	224

9.1.2	Requirements Elicitation Techniques . . . . .	226
9.1.3	Goals and Viewpoints . . . . .	234
9.1.4	Prioritizing Requirements . . . . .	237
9.1.5	COTS selection . . . . .	239
9.1.6	Crowdsourcing . . . . .	240
9.2	Requirements Documentation and Management . . . . .	241
9.2.1	Requirements Specification . . . . .	241
9.2.2	Requirements Management . . . . .	247
9.3	Requirements Specification Techniques . . . . .	249
9.3.1	Choosing a Notation . . . . .	250
9.3.2	Specifying Non-Functional Requirements . . . . .	252
9.4	Verification and Validation . . . . .	253
9.5	Summary . . . . .	254
9.6	Further Reading . . . . .	255
	Exercises . . . . .	257
<b>10</b>	<b>Modeling . . . . .</b>	<b>261</b>
10.1	Classic Modeling Techniques . . . . .	263
10.1.1	Entity–Relationship Modeling . . . . .	263
10.1.2	Finite State Machines . . . . .	265
10.1.3	Data Flow Diagrams . . . . .	267
10.1.4	CRC Cards . . . . .	267
10.2	On Objects and Related Stuff . . . . .	268
10.3	The Unified Modeling Language . . . . .	274
10.3.1	The Class Diagram . . . . .	276
10.3.2	The State Machine Diagram . . . . .	279
10.3.3	The Sequence Diagram . . . . .	283
10.3.4	The Communication Diagram . . . . .	284
10.3.5	The Component Diagram . . . . .	285
10.3.6	The Use Case . . . . .	286
10.4	Summary . . . . .	287
10.5	Further Reading . . . . .	287
	Exercises . . . . .	288
<b>11</b>	<b>Software Architecture . . . . .</b>	<b>289</b>
11.1	Software Architecture and the Software Life Cycle . . . . .	293
11.2	Architecture Design . . . . .	294
11.3	Architectural Views . . . . .	298
11.4	Architectural Styles . . . . .	303
11.5	Software Architecture Assessment . . . . .	317
11.6	Summary . . . . .	321
11.7	Further Reading . . . . .	322
	Exercises . . . . .	322

<b>12 Software Design</b> .....	<b>325</b>
12.1 Design Considerations .....	329
12.1.1 Abstraction .....	330
12.1.2 Modularity .....	333
12.1.3 Information Hiding .....	336
12.1.4 Complexity .....	337
12.1.5 System Structure .....	344
12.1.6 Object-Oriented Metrics .....	348
12.2 Classical Design Methods .....	351
12.2.1 Functional Decomposition .....	353
12.2.2 Data Flow Design (SA/SD) .....	356
12.2.3 Design Based on Data Structures .....	361
12.3 Object-Oriented Analysis and Design Methods .....	369
12.3.1 The Booch Method .....	376
12.3.2 Fusion .....	377
12.3.3 RUP Revisited .....	379
12.4 How to Select a Design Method .....	380
12.4.1 Design Method Classification .....	381
12.4.2 Object Orientation: Hype or the Answer? .....	382
12.5 Design Patterns .....	385
12.6 Design Documentation .....	389
12.7 Verification and Validation .....	393
12.8 Summary .....	393
12.9 Further Reading .....	398
Exercises .....	399
<b>13 Software Testing</b> .....	<b>405</b>
13.1 Test Objectives .....	410
13.1.1 Test Adequacy Criteria .....	412
13.1.2 Fault Detection Versus Confidence Building .....	413
13.1.3 From Fault Detection to Fault Prevention .....	415
13.2 Testing and the Software Life Cycle .....	417
13.2.1 Requirements Engineering .....	417
13.2.2 Design .....	419
13.2.3 Implementation .....	420
13.2.4 Maintenance .....	420
13.2.5 Test-Driven Development (TDD) .....	421
13.3 Verification and Validation Planning and Documentation .....	422
13.4 Manual Test Techniques .....	425
13.4.1 Reading .....	425
13.4.2 Walkthroughs and Inspections .....	426
13.4.3 Correctness Proofs .....	428
13.4.4 Stepwise Abstraction .....	429

13.5	Coverage-Based Test Techniques . . . . .	430
13.5.1	Control-Flow Coverage . . . . .	431
13.5.2	Data Flow Coverage . . . . .	433
13.5.3	Coverage-Based Testing of Requirements Specifications . . . . .	435
13.6	Fault-Based Test Techniques . . . . .	437
13.6.1	Error Seeding . . . . .	437
13.6.2	Mutation Testing . . . . .	438
13.7	Error-Based Test Techniques . . . . .	440
13.8	Comparison of Test Techniques . . . . .	441
13.8.1	Comparison of Test Adequacy Criteria . . . . .	442
13.8.2	Properties of Test Adequacy Criteria . . . . .	443
13.8.3	Experimental Results . . . . .	446
13.9	Test Stages . . . . .	448
13.10	Estimating Software Reliability . . . . .	450
13.11	Summary . . . . .	457
13.12	Further Reading . . . . .	458
	Exercises . . . . .	459
<b>14</b>	<b>Software Maintenance . . . . .</b>	<b>465</b>
14.1	Maintenance Categories Revisited . . . . .	468
14.2	Major Causes of Maintenance Problems . . . . .	471
14.3	Reverse Engineering and Refactoring . . . . .	475
14.3.1	Refactoring . . . . .	478
14.3.2	Inherent Limitations . . . . .	480
14.3.3	Tools . . . . .	484
14.4	Software Evolution Revisited . . . . .	486
14.5	Organizational and Managerial Issues . . . . .	488
14.5.1	Organization of Maintenance Activities . . . . .	488
14.5.2	Software Maintenance from a Service Perspective . . . . .	492
14.5.3	Control of Maintenance Tasks . . . . .	497
14.5.4	Quality Issues . . . . .	500
14.6	Summary . . . . .	501
14.7	Further Reading . . . . .	502
	Exercises . . . . .	504
<b>15</b>	<b>Software Tools . . . . .</b>	<b>507</b>
15.1	Toolkits . . . . .	511
15.2	Language-Centered Environments . . . . .	513
15.3	Integrated Environments and WorkBenches . . . . .	514
15.3.1	Analyst WorkBenches . . . . .	515
15.3.2	Programmer WorkBenches . . . . .	516
15.3.3	Management WorkBenches . . . . .	520
15.3.4	Integrated Project Support Environments . . . . .	520

15.4	Process-Centered Environments . . . . .	521
15.5	Summary . . . . .	522
15.6	Further Reading . . . . .	524
	Exercises . . . . .	525
<b>Part III Advanced Topics</b>		<b>527</b>
<b>16</b>	<b>User Interface Design . . . . .</b>	<b>529</b>
16.1	Where Is the User Interface? . . . . .	532
16.2	What Is the User Interface? . . . . .	536
16.3	Human Factors in Human–Computer Interaction . . . . .	537
	16.3.1 Humanities . . . . .	537
	16.3.2 Artistic Design . . . . .	538
	16.3.3 Ergonomics . . . . .	539
16.4	The Role of Models in Human–Computer Interaction . . . . .	540
	16.4.1 A Model of Human Information Processing . . . . .	542
	16.4.2 Mental Models of Information Systems . . . . .	544
	16.4.3 Conceptual Models in User Interface Design . . . . .	547
16.5	The Design of Interactive Systems . . . . .	549
	16.5.1 Design as an Activity Structure . . . . .	550
	16.5.2 Design as Multi-Disciplinary Collaboration . . . . .	552
16.6	Task Analysis . . . . .	553
	16.6.1 Task Analysis in HCI Design . . . . .	554
	16.6.2 Analysis Approaches for Collaborative Work . . . . .	556
	16.6.3 Sources of Knowledge and Collection Methods . . . . .	557
	16.6.4 An Integrated Approach to Task Analysis: GTA . . . . .	558
16.7	Specification of the User Interface Details . . . . .	559
	16.7.1 Dialog . . . . .	560
	16.7.2 Representation . . . . .	561
16.8	Evaluation . . . . .	562
	16.8.1 Evaluation of Analysis Decisions . . . . .	562
	16.8.2 Evaluation of UVM Specifications . . . . .	563
	16.8.3 Evaluation of Prototypes . . . . .	566
16.9	Summary . . . . .	567
16.10	Further Reading . . . . .	568
	Exercises . . . . .	569
<b>17</b>	<b>Software Reusability . . . . .</b>	<b>571</b>
17.1	Reuse Dimensions . . . . .	574
17.2	Reuse of Intermediate Products . . . . .	576
	17.2.1 Libraries of Software Components . . . . .	576
	17.2.2 Templates . . . . .	580

17.2.3	Reuse of Architecture . . . . .	581
17.2.4	Application Generators and Fourth-Generation Languages . . . . .	581
17.3	Reuse and the Software Life Cycle . . . . .	582
17.4	Reuse Tools and Techniques . . . . .	585
17.4.1	From Module Interconnection Language to Architecture Description Language . . . . .	586
17.4.2	Middleware . . . . .	588
17.5	Perspectives of Software Reuse . . . . .	591
17.6	Non-Technical Aspects of Software Reuse . . . . .	594
17.6.1	Economics . . . . .	596
17.6.2	Management . . . . .	597
17.6.3	Psychology of Programmers . . . . .	598
17.7	Summary . . . . .	599
17.8	Further Reading . . . . .	601
	Exercises . . . . .	601
<b>18</b>	<b>Component-Based Software Engineering . . . . .</b>	<b>605</b>
18.1	Why Component-Based Software Engineering? . . . . .	607
18.2	Component Models and Components . . . . .	608
18.2.1	Component Forms in Component Models . . . . .	610
18.2.2	Architecture and Component Models . . . . .	614
18.3	Component-Based Development Process and Component Life Cycle . . . . .	619
18.3.1	Component-Based System Development Process . . . . .	620
18.3.2	Component Assessment . . . . .	622
18.3.3	Component Development Process . . . . .	623
18.4	Architectural Approaches in Component-Based Development . . . . .	625
18.4.1	Architecture-Driven Component Development . . . . .	626
18.4.2	Product-Line Development . . . . .	626
18.4.3	COTS-Based Development . . . . .	627
18.4.4	Selecting an Approach . . . . .	627
18.5	Summary . . . . .	628
18.6	Further Reading . . . . .	628
	Exercises . . . . .	629
<b>19</b>	<b>Service Orientation . . . . .</b>	<b>631</b>
19.1	Services, Service Descriptions, and Service Communication . . . . .	634
19.2	Service-Oriented Architecture (SOA) . . . . .	639
19.3	Web Services . . . . .	641
19.3.1	Extensible Markup Language (XML) . . . . .	643
19.3.2	Simple Object Access Protocol (SOAP) . . . . .	644
19.3.3	Web Services Description Language (WSDL) . . . . .	644
19.3.4	Universal Description, Discovery, and Integration (UDDI) . . . . .	646

19.3.5	Business Process Execution Language for Web Services (BPEL4WS) .....	647
19.4	Service-Oriented Software Engineering .....	650
19.5	Summary .....	652
19.6	Further Reading .....	652
	Exercises .....	653
<b>20</b>	<b>Global Software Development</b> .....	<b>655</b>
20.1	Challenges of Global System Development .....	657
20.2	How to Overcome Distance .....	664
20.2.1	Common Ground .....	664
20.2.2	Coupling of Work .....	666
20.2.3	Collaboration Readiness .....	666
20.2.4	Technology Readiness .....	666
20.2.5	Organizing Work in Global Software Development .....	668
20.3	Summary .....	670
20.4	Further Reading .....	670
	Exercises .....	671
	<b>Bibliography</b> .....	<b>673</b>
	<b>Index</b> .....	<b>705</b>

