

Detailed Contents

Preface

v

Part I Understanding the Realm of Software Engineering

1. What is Software Engineering?	3
1.1 Motivation	3
1.2 Definition of Software Engineering	4
1.3 Characteristics of Software	5
1.4 Problems Confronted by Software Engineering	6
1.4.1 Problem of Change	6
1.4.2 Problem of Complexity	7
1.5 The Software Engineering Response	8
1.6 Challenges with the Response	10
1.7 Grand Challenge	11
1.8 What it is Like to be a Software Engineer?	12
1.8.1 Knowing across Domains	12
1.8.2 Teaming across Cultures	12
1.8.3 Innovating across Technologies	13
2. Evolution of Software Engineering	17
2.1 Motivation	17
2.2 Need to Know History	18
2.3 Evolutionary Trends	19
2.3.1 Programming to Software Engineering	19
2.3.2 Hardware-Software: From Coupling to Congress	20
2.3.3 Advent of High-Level Languages	22
2.3.4 Advent of the Personal Computer	24
2.3.5 Global Software Development	25
2.3.6 Return of Open Source	26
2.4 Milestones in Software Engineering	27
2.5 Towards a Slew of Silver Bullets	28
3. Basic Ideas and First Principles	33
3.1 Motivation	33
3.2 A Word of Caution	34
3.3 Are There Laws of Software Engineering?	34

xii Detailed Contents

3.4	Software Engineering versus Other Engineering Disciplines	36
3.4.1	How an Engineering Approach to Software Helps	38
3.4.2	How an Engineering Approach to Software Hinders	38
3.5	Characterizing Software and Software Engineering	39
3.5.1	No Laws of Software Engineering, Yet	39
3.5.2	Development versus Production	40
3.5.3	Plasticity of Software	40
3.5.4	Macro- and Micro-states	41
3.5.5	Importance of the Human Aspects	42
3.5.6	Concept of Co-evolution	43
3.6	Tying the Threads Together	43

Part II Planning and Managing Software Development

4.	Software Development Methodologies	51
4.1	Motivation	51
4.2	A Method to the Madness	52
4.3	Software Development Life Cycle	53
4.4	Algorithm, Process, and Methodology	55
4.5	Different Development Philosophies	56
4.5.1	Sequential Development	57
4.5.2	Iterative Development	57
4.6	Brief Review of Software Development Methodologies	58
4.6.1	Code-a-Bit-Test-a-Bit	58
4.6.2	Waterfall	58
4.6.3	Rapid Prototyping	59
4.6.4	Iterative and Incremental Development	61
4.6.5	Spiral	64
4.6.6	Extreme Programming and Agile Processes	65
4.7	People and Processes	67
5.	Place of Process in Software Development	72
5.1	Motivation	72
5.2	What is a Process?	73
5.3	Processes and Software Engineering	74
5.4	From Micro to Macro	75
5.5	Personal Software Process	75
5.6	Team Software Process	77
5.7	Unified Software Development Process	78
5.8	Towards Process Improvement and Process Making	80
	<i>Case Study</i>	81

6. Software Estimation	84
6.1 Motivation	84
6.2 What is Estimation?	85
6.3 Science and Art of Software Estimation	85
6.4 Importance of Estimation in Software Development	86
6.4.1 Getting the Work	87
6.4.2 Getting the Work Done	87
6.4.3 Getting the Work Done Well	87
6.5 Why is Good Estimation So Difficult?	88
<i>Case Study</i>	90
6.6 Some Standard Estimation Techniques	91
6.6.1 Estimation by Judgement	93
6.6.2 Estimation by Comparison	95
6.6.3 Estimation by Correlation	96
6.7 Estimating Size	98
6.8 Estimating Effort	99
6.9 Estimating Time	100
6.10 Estimation and Experience	100
7. Role of Metrics in Software Development	109
7.1 Motivation	109
7.2 Need for Measurement	110
7.3 Metrics Go Beyond Mere Measuring	111
7.4 Metrics, Management, and Beyond	112
7.5 Brief Review of Software Metrics	112
7.5.1 Early Perspectives	113
7.5.2 A Maturing Discipline	116
7.5.3 Towards a Deeper Perception	117
7.5.4 Metrics in the New Millennium	123
7.6 Art and Craft of Metrics Making	128
<i>Case Study</i>	129
<i>Shifting Sands of Design</i>	130
<i>Making of the Metric</i>	130
<i>Derivation—First Pass</i>	130
<i>Derivation—Second Pass</i>	132
<i>Back to Preeti</i>	133
<i>An Allied Metric—Whitmire's Volatility Index</i>	134
8. Software Project Management	141
8.1 Motivation	141

xiv Detailed Contents

8.2	That Elusive Something	142
8.3	Four Ps of Software Development: People, Project, Product, and Process	143
8.4	Project Life Cycle	144
8.5	Principles of Software Project Management	146
8.6	Project Management: Processes Groups and Knowledge Areas	148
8.7	Software Project Management Plan	150
8.8	Team Dynamics	152
8.9	Important Project Management Activities	152
8.9.1	Defining a Task Network	153
8.9.2	Scheduling	153
8.9.3	Earned Value Analysis	154
8.9.4	Error Tracking	154
8.10	Managing versus Leading	154
9.	Human Aspects of Software Development	159
9.1	Motivation	159
9.2	Software for Real Users	161
9.3	Capricious Users	161
	<i>Case Study</i>	163
9.4	Helping Users Know their Needs	165
9.5	Co-evolution: Interaction of the Problem and Solution Domains	166
9.6	Language and Communication	168
9.7	Human-Computer Interaction	169
9.8	Towards Usable Software Systems	169
9.9	The Human Factor	171
10.	Role of Automation in Software Development	176
10.1	Motivation	176
10.2	Computer-Aided Software Engineering (CASE)	177
10.3	The Odyssey of Automation	179
10.4	Automation: Why, How, and What	182
10.4.1	Test Automation	185
10.4.2	Implementation Automation	185
10.4.3	Design Automation	186
10.4.4	Automation of Specification and Analysis	186
10.4.5	Spectrum of Automation	186
10.5	Automating One Aspect of Design: An Example	188
10.5.1	Aptitude Index	189
10.5.2	Requirement Set	190
10.5.3	Concordance Index	190
	<i>Case Study</i>	193

Part III Making Software

11. Understanding Software Architecture	203
11.1 Motivation	203
11.2 Architectural Views of Software	204
11.3 Views and Definitions of Software Architecture	206
11.4 Need for Architecture in Large-Scale Software Systems	207
11.5 How Architecture Differs from Design	209
11.6 Architectural Patterns	210
11.7 Future of Software Architecture	212
<i>Case Study</i>	213
12. Paradigms of Software Development	219
12.1 Motivation	219
12.2 A Cooking Metaphor	220
12.3 Case for Software's Complexity	221
12.4 Strategies for Addressing Complexity in Software Systems	223
12.4.1 Decomposition	223
12.4.2 Abstraction	224
12.4.3 Hierarchies	225
12.5 Different Software Development Paradigms	225
12.5.1 Algorithmic Paradigm	225
12.5.2 Object-Oriented Paradigm	229
12.5.3 Aspect-Oriented Paradigm	231
12.6 Paradigms, Perspectives, and Programming	233
12.7 A Holistic View	234
<i>Case Study</i>	235
13. Languages of Software Development	246
13.1 Motivation	246
13.2 Incremental Approach to Learn Languages	249
13.3 Programming Languages	249
13.3.1 Journey of Programming Languages: Milestones	250
13.3.2 Profusion of Programming Languages	252
13.3.3 Classification of Programming Languages	253
13.3.4 Choice of a Programming Language	255
13.4 Modelling Languages	257
13.4.1 Essence of a Model	257
13.4.2 Unified Modelling Language	260
13.5 Specification Languages	264
13.5.1 Ten Commandments of Formal Methods	265
13.5.2 Simple Example Using Z	268

14. Software Development across Workflows and Phases	279
14.1 Motivation	279
14.2 Dimensionality of Software Development	282
14.3 Phases and Workflows in Perspective	286
14.4 A Model for Software Development	286
14.5 Workflows	287
14.5.1 Requirements	287
14.5.2 Analysis	291
14.5.3 Design	296
14.5.4 Implementation	299
14.5.5 Test	300
14.6 Phases	302
14.6.1 Inception	303
14.6.2 Elaboration	305
14.6.3 Construction	306
14.6.4 Transition	307
14.7 Workflows across Phases	308
15. Building a Software System: An Extended Case Study	317
15.1 Motivation	317
15.2 Example System: An Overview	318
15.3 Requirements	319
15.4 Analysis	325
15.5 Design	328
15.6 Implementation	337
15.7 Testing	353
15.8 Phase Milestones	354
15.9 Limitations of Case Study	354
16. Tricks of the Trade	357
16.1 Motivation	357
16.2 Refactor, Reuse, Refine	358
16.3 Refactor	359
16.4 Reuse	360
16.5 Refine	365
16.6 Structured Analysis and Data Dictionary	365
16.7 Modular Design	366
16.8 Transform and Transaction Mapping	367
16.9 Real-Time Software Design	367
16.9.1 Real-Time Executive	368

Part IV Testing, Maintaining, and Modifying Software Systems

17. Software Testing, Reliability, and Quality	375
17.1 Motivation	375
17.2 Some Testing Terms	376
17.3 Some Testing Tenets	378
17.4 Two Testing Philosophies	379
17.4.1 Black-Box Testing	379
17.4.2 White-Box Testing	381
17.5 Different Types of Testing	383
17.5.1 Unit Testing	383
17.5.2 Integration Testing	384
17.5.3 Regression Testing	387
17.5.4 Performance Testing	387
17.5.5 Stress Testing	388
17.5.6 User-Acceptance Testing	388
17.5.7 Validation Testing	389
17.6 Inspections, Walkthroughs, and Reviews	389
17.7 Designing Test Cases	390
<i>Case Study</i>	391
17.8 Debugging Techniques	392
17.8.1 Debugging by Brute Force	393
17.8.2 Debugging by Induction	393
17.8.3 Debugging by Deduction	393
17.8.4 Debugging by Backtracking	394
17.9 Test Automation	394
17.10 Basic Ideas of Software Reliability	395
17.10.1 Difference between Software and Hardware Reliability	396
17.10.2 Some Useful Software Reliability Relations	397
17.11 Towards Software Quality	398
17.11.1 ISO 9000 Series of Standards	399
17.11.2 Capability Maturity Model	399
17.11.3 Six Sigma	400
18. Towards Software Evolution	411
18.1 Motivation	411
18.2 Life after the Life Cycle	411
18.3 Maintenance and Modification	412
18.4 Software Entropy	413
18.5 Software Evolution	415

Part V Latest Trends of Software Development

19. Software Engineering and the World Wide Web	423
19.1 Motivation	423
19.2 Internet and the WWW	425
19.3 Software Applications: Before and After the Web	430
19.4 Architecture of Web-Based Software Systems	431
19.5 Software Systems on the Web: Salient Features	432
19.6 Web as a Software Development Medium	433
20. Towards Enterprise Software Development	438
20.1 Motivation	438
20.2 How is Enterprise Software Development Different?	440
20.3 Importance of Enterprise Software	443
20.4 Challenges Unique To Enterprise Software Development	443
20.5 Enterprise-Oriented Software Engineering	445
20.5.1 Identifying and Understanding Stakeholders' Needs	446
20.5.2 Choice of a Methodology	447
20.5.3 User Involvement and Feedback	448
20.5.4 Continual Development	449
<i>Case Study</i>	450
21. Global Software Development	456
21.1 Motivation	456
21.2 What is So Special about Global Software Development?	457
21.3 Genesis of Global Software Development	458
21.4 Distributed Teams and Remote Customers	459
21.5 Outsourcing: A Quick Reflection	460
21.6 Global Software Engineer	461
22. Open Source Software Development	466
22.1 Motivation	466
22.2 What is Open Source Software?	467
22.3 Evolution of Open Source Software	468
22.3.1 From Free to Proprietary	468
22.3.2 Open Source Response	469
22.3.3 Spread of the Mantra	470
22.3.4 Open Source as an Institution	471
22.4 Range and Limitations of Open Source Software	471
22.5 Open Source Software and the Professional Software Engineer	473

23. Future of Software Development	478
23.1 Motivation	478
23.2 Evolving Trends in Software Development	479
23.2.1 Understanding of Software Engineering	479
23.2.2 Planning and Managing Software Development	480
23.2.3 Designing and Building Software Systems	480
23.2.4 Testing, Maintenance, and Modifications	481
23.2.5 What will be the Next Big Thing?	481
23.3 Software Engineer's Survival Toolkit	483
23.3.1 Virtuosity with at least One Programming Language	483
23.3.2 In-depth Experience with at least One Development Methodology	484
23.3.3 Detailed Understanding of at least One Application Domain	484
23.3.4 Sense of History	485
<i>Index</i>	<i>489</i>