

# Contents

<b>Preface</b> .....	xi
<b>1. The Software Problem</b> .....	1
1.1 Cost, Schedule, and Quality .....	2
1.2 Scale and Change .....	5
1.3 Summary .....	7
Self Assessment Exercises .....	8
<b>2. Software Processes</b> .....	9
2.1 Process and Project .....	10
2.2 Component Software Processes .....	11
2.3 Software Development Process Models .....	13
2.3.1 Waterfall Model .....	14
2.3.2 Prototyping .....	17
2.3.3 Iterative Development .....	19
2.3.4 Rational Unified Process .....	22
2.3.5 Timeboxing Model .....	25
2.3.6 Extreme Programming and Agile Processes .....	28
2.3.7 Using Process Models in a Project .....	30
2.4 Project Management Process .....	32
2.5 Summary .....	34
Self Assessment Exercises .....	36
<b>3. Software Requirements Analysis and Specification</b> .....	37
3.1 Value of a Good SRS .....	38
3.2 Requirement Process .....	39

3.3	Requirements Specification	41
3.3.1	Desirable Characteristics of an SRS	41
3.3.2	Components of an SRS	43
3.3.3	Structure of a Requirements Document	46
3.4	Functional Specification with Use Cases	49
3.4.1	Basics	49
3.4.2	Examples	52
3.4.3	Extensions	54
3.4.4	Developing Use Cases	56
3.5	Other Approaches for Analysis	58
3.5.1	Data Flow Diagrams	59
3.5.2	ER Diagrams	61
3.6	Validation	63
3.7	Summary	66
	Self Assessment Exercises	67
<b>4.</b>	<b>Planning a Software Project</b>	<b>69</b>
4.1	Effort Estimation	70
4.1.1	Top-Down Estimation Approach	71
4.1.2	Bottom-Up Estimation Approach	74
4.2	Project Schedule and Staffing	76
4.3	Quality Planning	78
4.4	Risk Management Planning	80
4.4.1	Risk Management Concepts	81
4.4.2	Risk Assessment	81
4.4.3	Risk Control	83
4.4.4	A Practical Risk Management Planning Approach	84
4.5	Project Monitoring Plan	86
4.5.1	Measurements	86
4.5.2	Project Monitoring and Tracking	87
4.6	Detailed Scheduling	88
4.7	Summary	91
	Self Assessment Exercises	92
<b>5.</b>	<b>Software Architecture</b>	<b>95</b>
5.1	Role of Software Architecture	96
5.2	Architecture Views	98
5.3	Component and Connector View	101
5.3.1	Components	102
5.3.2	Connectors	103
5.3.3	An Example	104
5.4	Architecture Styles for C&C View	108

---

5.4.1	Pipe and Filter	108
5.4.2	Shared-Data Style	110
5.4.3	Client-Server Style	112
5.4.4	Some Other Styles	113
5.5	Documenting Architecture Design	114
5.6	Evaluating Architectures	118
5.7	Summary	119
	Self Assessment Exercises	120
<b>6.</b>	<b>Design</b>	<b>121</b>
6.1	Design Concepts	122
6.1.1	Coupling	123
6.1.2	Cohesion	126
6.1.3	The Open-Closed Principle	129
6.2	Function-Oriented Design	131
6.2.1	Structure Charts	132
6.2.2	Structured Design Methodology	134
6.2.3	An Example	139
6.3	Object-Oriented Design	142
6.3.1	OO Concepts	143
6.3.2	Unified Modeling Language (UML)	147
6.3.3	A Design Methodology	156
6.3.4	Examples	162
6.4	Detailed Design	168
6.4.1	Logic/Algorithm Design	169
6.4.2	State Modeling of Classes	170
6.5	Verification	171
6.6	Metrics	172
6.6.1	Complexity Metrics for Function-Oriented Design	172
6.6.2	Complexity Metrics for OO Design	175
6.7	Summary	176
	Self Assessment Exercises	178
<b>7.</b>	<b>Coding and Unit Testing</b>	<b>179</b>
7.1	Programming Principles and Guidelines	180
7.1.1	Structured Programming	181
7.1.2	Information Hiding	184
7.1.3	Some Programming Practices	185
7.1.4	Coding Standards	189
7.2	Incrementally Developing Code	192
7.2.1	An Incremental Coding Process	192
7.2.2	Test Driven Development	193

---

7.2.3	Pair Programming	195
7.3	Managing Evolving Code	196
7.3.1	Source Code Control and Build	196
7.3.2	Refactoring	198
7.4	Unit Testing	202
7.4.1	Testing Procedural Units	203
7.4.2	Unit Testing of Classes	205
7.5	Code Inspection	206
7.5.1	Planning	209
7.5.2	Self-Review	209
7.5.3	Group Review Meeting	210
7.6	Metrics	212
7.6.1	Size Measures	212
7.6.2	Complexity Metrics	214
7.7	Summary	219
	Self Assessment Exercises	220
<b>8.</b>	<b>Testing</b>	<b>223</b>
8.1	Testing Concepts	224
8.1.1	Error, Fault, and Failure	224
8.1.2	Test case, Test Suite, and Test Harness	225
8.1.3	Psychology of Testing	226
8.1.4	Levels of Testing	227
8.2	Testing Process	229
8.2.1	Test Plan	229
8.2.2	Test Case Design	231
8.2.3	Test Case Execution	232
8.3	Black-Box Testing	234
8.3.1	Equivalence Class Partitioning	235
8.3.2	Boundary Value Analysis	237
8.3.3	Pair-wise Testing	238
8.3.4	Special Cases	241
8.3.5	State-Based Testing	242
8.4	White-Box Testing	245
8.4.1	Control Flow-Based Criteria	246
8.4.2	Test Case Generation and Tool Support	249
8.5	Metrics	250
8.5.1	Coverage Analysis	250
8.5.2	Reliability	251
8.5.3	Defect Removal Efficiency	252
8.6	Summary	253

---

Self Assessment Exercises . . . . .	254
<b>Bibliography</b> . . . . .	257
<b>Index</b> . . . . .	263