

CONTENTS

List of Figures	xvii
List of Tables	xxi
Preface	xxv

PART I OVERVIEW AND BASICS

1 Overview	3
1.1 Meeting People's Quality Expectations	3
1.2 Book Organization and Chapter Overview	6
1.3 Dependency and Suggested Usage	9
1.4 Reader Preparation and Background Knowledge	11
Problems	13
2 What Is Software Quality?	15
2.1 Quality: Perspectives and Expectations	15
2.2 Quality Frameworks and ISO-9126	18
2.3 Correctness and Defects: Definitions, Properties, and Measurements	20
2.4 A Historical Perspective of Quality	24
2.5 So, What Is Software Quality?	25
Problems	26
	vii

3	Quality Assurance	27
3.1	Classification: QA as Dealing with Defects	27
3.2	Defect Prevention	31
3.2.1	Education and training	31
3.2.2	Formal method	32
3.2.3	Other defect prevention techniques	33
3.3	Defect Reduction	34
3.3.1	Inspection: Direct fault detection and removal	34
3.3.2	Testing: Failure observation and fault removal	35
3.3.3	Other techniques and risk identification	36
3.4	Defect Containment	37
3.4.1	Software fault tolerance	37
3.4.2	Safety assurance and failure containment	38
3.5	Concluding Remarks	38
	Problems	39
4	Quality Assurance in Context	41
4.1	Handling Discovered Defect During QA Activities	41
4.2	QA Activities in Software Processes	43
4.3	Verification and Validation Perspectives	46
4.4	Reconciling the Two Views	49
4.5	Concluding Remarks	51
	Problems	52
5	Quality Engineering	53
5.1	Quality Engineering: Activities and Process	53
5.2	Quality Planning: Goal Setting and Strategy Formation	56
5.3	Quality Assessment and Improvement	59
5.4	Quality Engineering in Software Processes	59
5.5	Concluding Remarks	63
	Problems	64
PART II SOFTWARE TESTING		
6	Testing: Concepts, Issues, and Techniques	67
6.1	Purposes, Activities, Processes, and Context	67
6.2	Questions About Testing	71
6.3	Functional vs. Structural Testing: What to Test?	74
6.4	Coverage-Based vs. Usage-Based Testing: When to Stop Testing?	78
6.5	Concluding Remarks	83
	Problems	84

7	Test Activities, Management, and Automation	85
7.1	Test Planning and Preparation	85
7.1.1	Test planning: Goals, strategies, and techniques	85
7.1.2	Testing models and test cases	86
7.1.3	Test suite preparation and management	88
7.1.4	Preparation of test procedure	89
7.2	Test Execution, Result Checking, and Measurement	90
7.3	Analysis and Follow-up	93
7.4	Activities, People, and Management	95
7.5	Test Automation	97
7.6	Concluding Remarks	100
	Problems	101
8	Coverage and Usage Testing Based on Checklists and Partitions	103
8.1	Checklist-Based Testing and Its Limitations	103
8.2	Testing for Partition Coverage	107
8.2.1	Some motivational examples	107
8.2.2	Partition: Concepts and definitions	108
8.2.3	Testing decisions and predicates for partition coverage	109
8.3	Usage-Based Statistical Testing with Musa's Operational Profiles	111
8.3.1	The cases for usage-based statistical testing	111
8.3.2	Musa OP: Basic ideas	112
8.3.3	Using OPs for statistical testing and other purposes	114
8.4	Constructing Operational Profiles	115
8.4.1	Generic methods and participants	116
8.4.2	OP development procedure: Musa-1	117
8.4.3	OP development procedure: Musa-2	119
8.5	Case Study: OP for the Cartridge Support Software	121
8.5.1	Background and participants	121
8.5.2	OP development in five steps	122
8.5.3	Metrics collection, result validation, and lessons learned	124
8.6	Concluding Remarks	125
	Problems	126
9	Input Domain Partitioning and Boundary Testing	127
9.1	Input Domain Partitioning and Testing	128
9.1.1	Basic concepts, definitions, and terminology	128
9.1.2	Input domain testing for partition and boundary problems	130
9.2	Simple Domain Analysis and the Extreme Point Combination Strategy	132
9.3	Testing Strategies Based on Boundary Analysis	135
9.3.1	Weak $N \times 1$ strategy	135

9.3.2	Weak 1×1 strategy	139
9.4	Other Boundary Test Strategies and Applications	140
9.4.1	Strong and approximate strategies	140
9.4.2	Other types of boundaries and extensions	141
9.4.3	Queuing testing as boundary testing	142
9.5	Concluding Remarks	144
	Problems	145
10	Coverage and Usage Testing Based on Finite-State Machines and Markov Chains	147
10.1	Finite-State Machines and Testing	148
10.1.1	Overcoming limitations of simple processing models	148
10.1.2	FSMs: Basic concepts and examples	149
10.1.3	Representations of FSMs	151
10.2	FSM Testing: State and Transition Coverage	153
10.2.1	Some typical problems with systems modeled by FSMs	153
10.2.2	Model construction and validation	154
10.2.3	Testing for correct states and transitions	155
10.2.4	Applications and limitations	156
10.3	Case Study: FSM-Based Testing of Web-Based Applications	157
10.3.1	Characteristics of web-based applications	157
10.3.2	What to test: Characteristics of web problems	158
10.3.3	FSMs for web testing	159
10.4	Markov Chains and Unified Markov Models for Testing	160
10.4.1	Markov chains and operational profiles	161
10.4.2	From individual Markov chains to unified Markov models	162
10.4.3	UMM construction	164
10.5	Using UMMs for Usage-Based Statistical Testing	164
10.5.1	Testing based on usage frequencies in UMMs	164
10.5.2	Testing based on other criteria and UMM hierarchies	165
10.5.3	Implementation, application, and other issues	166
10.6	Case Study Continued: Testing Based on Web Usages	167
10.6.1	Usage-based web testing: Motivations and basic approach	167
10.6.2	Constructing UMMs for statistical web testing	168
10.6.3	Statistical web testing: Details and examples	169
10.7	Concluding Remarks	171
	Problems	172
11	Control Flow, Data Dependency, and Interaction Testing	175
11.1	Basic Control Flow Testing	176
11.1.1	General concepts	176

11.1.2	Model construction	178
11.1.3	Path selection	180
11.1.4	Path sensitization and other activities	181
11.2	Loop Testing, CFT Usage, and Other Issues	182
11.2.1	Different types of loops and corresponding CFGs	182
11.2.2	Loop testing: Difficulties and a heuristic strategy	184
11.2.3	CFT Usage and Other Issues	186
11.3	Data Dependency and Data Flow Testing	186
11.3.1	Basic concepts: Operations on data and data dependencies	187
11.3.2	Basics of DFT and DDG	188
11.3.3	DDG elements and characteristics	189
11.3.4	Information sources and generic procedure for DDG construction	191
11.3.5	Building DDG indirectly	192
11.3.6	Dealing with loops	194
11.4	DFT: Coverage and Applications	195
11.4.1	Achieving slice and other coverage	195
11.4.2	DFT: Applications and other issues	198
11.4.3	DFT application in synchronization testing	199
11.5	Concluding Remarks	200
	Problems	200
12	Testing Techniques: Adaptation, Specialization, and Integration	203
12.1	Testing Sub-Phases and Applicable Testing Techniques	203
12.2	Specialized Test Tasks and Techniques	210
12.3	Test Integration	214
12.4	Case Study: Hierarchical Web Testing	214
12.5	Concluding Remarks	217
	Problems	219
PART III QUALITY ASSURANCE BEYOND TESTING		
13	Defect Prevention and Process Improvement	223
13.1	Basic Concepts and Generic Approaches	223
13.2	Root Cause Analysis for Defect Prevention	224
13.3	Education and Training for Defect Prevention	225
13.4	Other Techniques for Defect Prevention	228
13.4.1	Analysis and modeling for defect prevention	228
13.4.2	Technologies, standards, and methodologies for defect prevention	229
13.4.3	Software tools to block defect injection	230
13.5	Focusing on Software Processes	231
13.5.1	Process selection, definition, and conformance	231
13.5.2	Process maturity	232

13.5.3	Process and quality improvement	233
13.6	Concluding Remarks	234
	Problems	235
14	Software Inspection	237
14.1	Basic Concepts and Generic Process	237
14.2	Fagan inspection	239
14.3	Other Inspections and Related Activities	242
14.3.1	Inspections of reduced scope or team size	242
14.3.2	Inspections of enlarged scope or team size	243
14.3.3	Informal desk checks, reviews, and walkthroughs	244
14.3.4	Code reading	244
14.3.5	Other formal reviews and static analyses	246
14.4	Defect Detection Techniques, Tool/Process Support, and Effectiveness	247
14.5	Concluding Remarks	249
	Problems	250
15	Formal Verification	251
15.1	Basic Concepts: Formal Verification and Formal Specification	251
15.2	Formal Verification: Axiomatic Approach	254
15.2.1	Formal logic specifications	254
15.2.2	Axioms	255
15.2.3	Axiomatic proofs and a comprehensive example	257
15.3	Other Approaches	259
15.3.1	Weakest pre-conditions and backward chaining	260
15.3.2	Functional approach and symbolic execution	260
15.3.3	Seeking alternatives: Model checking and other approaches	261
15.4	Applications, Effectiveness, and Integration Issues	263
15.5	Concluding Remarks	265
	Problems	266
16	Fault Tolerance and Failure Containment	267
16.1	Basic Ideas and Concepts	267
16.2	Fault Tolerance with Recovery Blocks	270
16.3	Fault Tolerance with N-Version Programming	272
16.3.1	NVP: Basic technique and implementation	272
16.3.2	Ensuring version independence	273
16.3.3	Applying NVP ideas in other QA activities	274
16.4	Failure Containment: Safety Assurance and Damage Control	275
16.4.1	Hazard analysis using fault-trees and event-trees	275
16.4.2	Hazard resolution for accident prevention	278

16.4.3 Accident analysis and post-accident damage control	278
16.5 Application in Heterogeneous Systems	279
16.5.1 Modeling and analyzing heterogeneous systems	279
16.5.2 Prescriptive specifications for safety	281
16.6 Concluding Remarks	282
Problems	282
17 Comparing Quality Assurance Techniques and Activities	285
17.1 General Questions: Cost, Benefit, and Environment	285
17.2 Applicability to Different Environments	289
17.3 Effectiveness Comparison	291
17.3.1 Defect perspective	291
17.3.2 Problem types	292
17.3.3 Defect level and pervasiveness	293
17.3.4 Result interpretation and constructive information	294
17.4 Cost Comparison	295
17.5 Comparison Summary and Recommendations	297
Problems	298
PART IV QUANTIFIABLE QUALITY IMPROVEMENT	
18 Feedback Loop and Activities for Quantifiable Quality Improvement	303
18.1 QA Monitoring and Measurement	304
18.1.1 Direct vs. indirect quality measurements	304
18.1.2 Direct quality measurements: Result and defect measurements	306
18.1.3 Indirect quality measurements: Environmental, product internal, and activity measurements	306
18.2 Immediate Follow-up Actions and Feedback	308
18.3 Analyses and Follow-up Actions	309
18.3.1 Analyses for product release decisions	309
18.3.2 Analyses for other project management decisions	311
18.3.3 Other feedback and follow-up actions	312
18.4 Implementation, Integration, and Tool Support	313
18.4.1 Feedback loop: Implementation and integration	314
18.4.2 A refined quality engineering process	316
18.4.3 Tool support: Strategy, implementation, and integration	317
18.5 Concluding Remarks	320
Problems	320
19 Quality Models and Measurements	323
19.1 Models for Quality Assessment	323

19.2	Generalized Models	324
19.3	Product-Specific Models	327
19.4	Model Comparison and Interconnections	328
19.5	Data Requirements and Measurement	330
19.6	Selecting Measurements and Models	333
19.7	Concluding Remarks	335
	Problems	337
20	Defect Classification and Analysis	339
20.1	General Types of Defect Analyses	339
20.1.1	Defect distribution analysis	340
20.1.2	Defect trend analysis and defect dynamics model	343
20.1.3	Defect causal analysis	344
20.2	Defect Classification and ODC	345
20.2.1	ODC concepts	345
20.2.2	Defect classification using ODC: A comprehensive example	346
20.2.3	Adapting ODC to analyze web errors	347
20.3	Defect Analysis for Classified Data	348
20.3.1	One-way analysis: Analyzing a single defect attribute	348
20.3.2	Two-way and multi-way analysis: Examining cross-interactions	349
20.4	Concluding Remarks	350
	Problems	351
21	Risk Identification for Quantifiable Quality Improvement	353
21.1	Basic Ideas and Concepts	353
21.2	Traditional Statistical Analysis Techniques	355
21.3	New Techniques for Risk Identification	356
21.3.1	Principal component and discriminant analyses	356
21.3.2	Artificial neural networks and learning algorithms	358
21.3.3	Data partitions and tree-based modeling	359
21.3.4	Pattern matching and optimal set reduction	362
21.4	Comparisons and Integration	362
21.5	Risk Identification for Classified Defect Data	365
21.6	Concluding Remarks	368
	Problems	369
22	Software Reliability Engineering	371
22.1	SRE: Basic Concepts and General Approaches	371
22.2	Large Software Systems and Reliability Analyses	372
22.3	Reliability Snapshots Using IDRM	374
22.4	Longer-Term Reliability Analyses Using SRGMs	377

22.5 TBRMs for Reliability Analysis and Improvement	380
22.5.1 Constructing and using TBRMs	381
22.5.2 TBRM Applications	382
22.5.3 TBRM's impacts on reliability improvement	384
22.6 Implementation and Software Tool Support	385
22.7 SRE: Summary and Perspectives	386
Problems	387
Bibliography	389
Index	403

LIST OF FIGURES

1.1	Scope and content hierarchy: Testing, quality assurance (QA), and software quality engineering	6
1.2	Chapter and PART dependency diagram	10
2.1	Defect related concepts and relations	21
3.1	Generic ways to deal with defects	30
4.1	QA activities in the waterfall process	45
4.2	Verification and validation activities associated with the V-Model	49
5.1	Quality engineering process	54
5.2	Quality engineering in the waterfall process	61
5.3	Quality engineering effort profile: The share of different activities as part of the total effort	63
6.1	Generic testing process	69
7.1	Test coverage analysis with S-TCAT	100
8.1	An operational profile (OP) of requested file types for the SMU/SEAS web site	113
8.2	A tree-structured or graphical operational profile	121
9.1	1-dimensional domain testing with EPC strategy	133

9.2	2-dimensional domain testing with EPC strategy	134
9.3	1-dimensional domain testing with weak $N \times 1$ strategy	137
9.4	2-dimensional domain testing with weak $N \times 1$ strategy for the boundary between C0 and C2	138
9.5	2-dimensional boundary tilt detection by the weak $N \times 1$ strategy	138
9.6	2-dimensional domain testing with weak 1×1 strategy for the boundary between C0 and C6	139
9.7	2-dimensional boundary tilt detection by the weak 1×1 strategy	140
10.1	An example finite-state machine (FSM) for call processing	151
10.2	Multi-layered web applications	158
10.3	Example Markov chain for call processing FSM in Figure 10.1	162
10.4	Example UMM (unified Markov model): Expanding state E of the top-level UMM in Figure 10.3 into a lower-level UMM	163
10.5	Sample entries in an access log	168
10.6	Top-level UMM for SMU/SEAS	170
11.1	A sample control flow graph (CFG)	177
11.2	A sample program and its control flow graph (CFG)	179
11.3	Control flow graphs (CFGs) for “for” and “while” loops	183
11.4	Data dependency graph (DDG) element: An example of data definition through assignment	188
11.5	DDG element: An example of data selector node	190
11.6	A sample data flow graph (DDG)	192
11.7	Data selectors for multiple variables in branches	194
11.8	Three data slices for the DDG in Figure 11.6 and their sensitization	195
11.9	Combination of independent data selectors and related slices	196
11.10	Combination of nested data selectors and related slices	197
12.1	Testing sub-phases associated with the V-Model	204
12.2	Hierarchical implementation of an integrated web testing strategy	218
14.1	Generic inspection process	238
14.2	A program segment (left) and its permutation (right)	245
15.1	A program segment with its formal specification	258
16.1	Fault tolerance with recovery blocks	270

16.2	Fault tolerance with NVP	272
16.3	Fault-tree analysis (FTA) for an automobile accident	276
16.4	Event-tree analysis (ETA) for an automobile accident	277
16.5	Two-frame model for a CCSCS	280
16.6	Prescription monitor for safety assurance	281
18.1	Refined quality engineering process: Measurement, analysis, and feedback for quantifiable quality improvement	304
18.2	Further refined quality engineering process with detailed measurement sources and feedback paths	315
18.3	Tools for quality measurement, analysis, and feedback	319
19.1	Classification of quality assessment models	324
19.2	Effort or defect profile in the Putnam Model	326
19.3	Relating measurements to quality assessment models	332
19.4	A fitted SRGM for an IBM product	335
19.5	A tree-based reliability model (TBRM) for an IBM product	336
20.1	One-way analysis of defect impact for an IBM product	349
20.2	Error (type E) and hit profiles for SMU/SEAS	350
21.1	Processing model of a neuron	358
21.2	Backward propagation algorithm for artificial neural networks	359
21.3	Algorithm for tree-based model construction	360
21.4	Tree-based defect model for a commercial product	361
21.5	Algorithm for optimal set reduction	362
21.6	Example hierarchy for optimal set reduction	363
21.7	Predictions of defect impact for an IBM product	366
21.8	Defect impact distributions for an IBM product	367
22.1	Measured runs (per day) for products D	374
22.2	Measured transactions (per run) for products E	375
22.3	SRGMs for test run indexed failures for product D	380
22.4	TBRM1 for product D	383
22.5	TBRM2 for product D	383
22.6	Comparing failure arrivals for products A, B, C, and D	384

LIST OF TABLES

2.1	Correctness-centered properties according to quality views and attributes	23
4.1	QA activities: Mapping from defect-centered (DC) view to verification and validation (V&V) view	51
7.1	A template for test execution measurements	93
8.1	A high-level functional checklist for some relational database products	105
8.2	A template for a two-dimensional checklist by combining a standards checklist and a component checklist	106
8.3	Sample test cases for the program solving the equation $ax^2 + bx + c = 0$	108
8.4	Usage frequencies (hits) and probabilities (% of total) for different file types for SMU/SEAS	112
8.5	A sample customer profile	118
8.6	A sample user profile	119
8.7	CSS user profile	123
8.8	CSS OP: CSS functions classified according to usage probabilities	124
10.1	An example finite-state machine (FSM) for call processing in tabular representation	152
10.2	Top entry pages to SMU/SEAS	170

12.1	Comparison of key characteristics and applicable testing techniques for different testing sub-phases	209
13.1	Distribution of modules of different maturity for an IBM product	227
13.2	Process maturity levels in CMM	233
15.1	Example symbolic execution traces	261
17.1	Objects of QA alternatives	289
17.2	Development activities where different QA alternatives are applicable	290
17.3	Required expertise and background knowledge for people to perform different QA alternatives	291
17.4	Defect observed and dealt with by different QA alternatives	292
17.5	Main problem types dealt with by different QA alternatives	292
17.6	Defect levels where different QA alternatives are suitable	294
17.7	Ease of result interpretation for different QA alternatives and amount of constructive information/measurements	295
17.8	Cost comparison for different QA alternatives	297
17.9	General comparison for different QA alternatives	298
19.1	A segmented model for reliability level estimation	326
19.2	DRM (defect removal model): defect distribution for previous releases of a product	327
19.3	High-defect modules for two products identified by tree-based modeling	329
19.4	Summary of quality assessment models and their applications	329
19.5	Summary of measurements required by different quality models	331
19.6	Data attributes used in Figure 19.5	336
20.1	Common error types and error distribution for SMU/SEAS	341
20.2	Characterizing web errors by file types	342
20.3	Distribution of DF for a commercial product LS	342
20.4	Distribution of DF for a commercial product NS	343
20.5	A sample defect dynamics model	344
20.6	Some defect attributes and values for an IBM product	347
20.7	Two-way analysis results: Interaction between impact and severity	351
21.1	Principal components for a commercial product	357
21.2	Predicting defects using artificial neural networks	359

21.3	Characterizing high-defect modules for a commercial product	361
21.4	Comparison of risk identification techniques	364
22.1	Estimated reliability (\hat{R}) and failure rate ($\hat{\lambda}$) for successive time segments	376
22.2	Daily error rate (or failure rate) for SMU/SEAS	377
22.3	Comparing purification levels for products A, B, C, and D	384