

Contents

- 1 Introduction** 1
 - 1.1 Consistency, Complexity, and Change 1
 - 1.2 Synopsis 2
- 2 What Is Software Quality, and Why Does it Matter?** 7
 - 2.1 Why Care about Software Quality? 7
 - 2.2 What Drives Software Quality Assurance? 14
 - 2.3 Defining “Software Quality” 16
 - 2.3.1 The Challenge of Defining Quality 16
 - 2.3.2 Quality Models - a Historical Perspective 18
 - 2.4 Key Points 21
- 3 Software Development Processes and Process Improvement** 23
 - 3.1 Process and Process Improvement in Manufacturing 24
 - 3.1.1 The Industrial Revolution 24
 - 3.1.2 Plan Do Check Act 26
 - 3.1.3 Quality-Driven Manufacturing in Japan 27
 - 3.1.4 Total Quality Management 30
 - 3.2 The Software Development Process 31
 - 3.2.1 The Waterfall Model 33
 - 3.2.2 Iterative and Incremental Software Development 35
 - 3.3 Agile Software Development 38
 - 3.3.1 The Principles of Agile Software Development 38
 - 3.3.2 An Example: SCRUM 39
 - 3.3.3 Relation to Total Quality Management 42
 - 3.3.4 Why Not Always Go Agile? 44
 - 3.4 Software Process Improvement - The Capability Maturity Model ... 45
 - 3.5 Key Points 48

4	Managing Requirements and Code	51
4.1	Managing Requirements	51
4.1.1	What is a Requirement?	52
4.1.2	Requirements Elicitation	53
4.1.3	Requirements Documents	56
4.1.4	Security Requirements	59
4.1.5	Tracing Requirements	60
4.1.6	Prioritisation	62
4.1.7	Oversight with Kanban boards	64
4.2	Writing Maintainable Source Code and Handling Change	64
4.2.1	Coding Conventions and Design / Architecture Patterns	65
4.2.2	Collaborative Development and Version Repositories	69
4.3	Key Points	74
5	Planning Activities and Predicting Costs	77
5.1	Planning	78
5.1.1	Program Evaluation and Review Technique (PERT)	78
5.1.2	Gantt Charts	81
5.2	Predicting Costs	82
5.2.1	Base Models	82
5.2.2	Parameter Fitting by Linear Regression	83
5.2.3	COCOMO	84
5.2.4	Planning Poker	90
5.2.5	Uncertainty and Predictive Accuracy	91
5.2.6	Keeping Track of Progress	92
5.3	Key Points	94
6	Testing	95
6.1	The Foundations of Software Testing	95
6.2	White-Box Testing	99
6.2.1	Code coverage	99
6.2.2	White Box Test Generation	101
6.2.3	The Case(s) Against Code Coverage	106
6.2.4	Goto Fail: A Case For Code Coverage	108
6.2.5	An Alternative: Mutation Testing	109
6.3	Black-Box Testing	110
6.3.1	Specification-Based Testing	111
6.3.2	Random Testing	116
6.3.3	Exposing Security Flaws with Fuzz-Testing	123
6.4	Key Points	124
7	Software Inspections, Code Reviews, and Safety Arguments	127
7.1	Formal Inspections	128
7.2	Modern Code Reviews - Reviewing Code During Development	128
7.2.1	Tool-Driven Code Review	129

7.2.2	Pull-Based Development	130
7.2.3	The Impact of MCR on Software Development and Quality .	131
7.3	Code Reviewing Techniques	132
7.3.1	Tool-Driven Code Review	133
7.3.2	Developer-driven Code Reviews	134
7.4	Safety Arguments and Inspections of Safety Requirements	136
7.4.1	Checklists	136
7.4.2	Safety Argumentation and the Goal Structure Notation	138
7.5	Key Points	139
8	Measurement	141
8.1	Measurement Basics	142
8.2	Metrics	147
8.2.1	Size and Complexity	148
8.2.2	Modularity Metrics	153
8.2.3	Maintainability Metrics and the Maintainability Index	158
8.3	Validity and the Use of Goal Question Metric	159
8.3.1	Problems of Validity	159
8.3.2	Goal Question Metric	160
8.4	Key Points	162
9	Conclusions	165
9.1	Topical and Emerging Quality Concerns	165
9.1.1	Autonomy in Socio-Technical Systems	165
9.1.2	Data-Intensive, Untestable Systems	167
9.2	Concluding Remarks	169
	References	171
	Index	179

Software Quality Assurance

Consistency in the Face of Complexity and Change

Walkinshaw, N.

2017, XI, 181 p. 63 illus., 41 illus. in color., Softcover

ISBN: 978-3-319-64821-7