Inhaltsverzeichnis

1 Einleitung 1

1.1 Softwarequalität betrifft viele 1
1.2 Für wen dieses Buch gemacht ist 1
1.3 Was Sie von diesem Buch erwarten können 2
1.4 Das Abenteuer von Q 3
1.5 Themen und Anspruch 3
1.5.1 Themenauswahl und Gewichtung 4
1.5.2 Die Reihenfolge der Themen 5
1.6 Bedeutung von Softwarequalität 6
1.7 Wie Q zur Softwarequalität kam 8

2 Grundkonzepte 11

2.1 Qualitätsorganisation und Terminologie 12
2.2 Kosten und Nutzen von Softwarequalität 16
2.3 Qualitätsbeauftragte 18
2.4 Eine Vision: Total Quality Management 22
2.5 Grundbegriffe des Testens 23
2.6 Normen und Standards 28
2.7 Qualitätsaspekte, -anforderungen und Qualitätsmodelle 30

3 Erfahrungen systematisch nutzen 39

3.1 Qualitätsnetzwerke und Qualitätszirkel 40
3.2 Leichtgewichtige Dokumentation von Erfahrungen 42
3.3 Organisation der Erfahrungsverwaltung 45
3.4 Herausforderungen und Chancen für Erfahrungsnutzung 47
3.5 Networking in Organisationen und auf Tagungen 49

4 Messen von Softwarequalität 51
4.1 Wozu messen und konkretisieren? 51
4.2 Softwaremetriken 54
4.2.1 Grundlagen 54
4.2.2 Was Softwaremetriken messen 55
4.2.3 Bezug zwischen Metrik und Qualitätsaspekt 56
4.2.4 Skalen für die Resultate der Metriken 57
4.3 Diskussion bekannter Softwaremetriken 58
4.3.1 Lines of code: Der Teufel steckt im Detail 58
4.3.2 Zyklomatische Komplexität von McCabe 61
4.3.3 Halstead Software Science 66
4.3.4 Weitere Metriken: ein Ausblick 68
4.4 Metriken nach Maß: GQM 69
4.4.1 Von Zielen zu Fragen zu Metriken – und zurück 69
4.4.2 Zielorientiertes Messen und Bewerten 71
4.4.3 Zielfacetten schärfen den Blick 73
4.4.4 Messung vorbereiten mit Abstraction Sheets 74
4.4.5 Besonderheiten bei Messung und Auswertung 78
4.5 Projektfortschritt messen mit Quality Gates 79

5 Systematisches Testen 83
5.1 Vorüberlegungen 83
5.1.1 Testvorbereitung 83
5.1.2 Vollständig testen? 85
5.1.3 Woraus ein Testfall besteht 86
5.1.4 Testfälle dokumentieren 87
5.1.5 Testfälle ermitteln: eine Strategie 88
5.1.6 Hintergrund von Fehlern 90
5.1.7 Übersicht: Black-Box-Test und Glass-Box-Test 91
5.2 Black-Box-Tests aus der Spezifikation 92
5.2.1 Minimalforderung und Effizienzprinzip 93
5.2.2 Äquivalenzklassenmethode 94
5.2.3 Grenzwertanalyse 96
5.2.4 Spezifikationsabdeckung optimieren 96
5.2.5 Klassifikationsbaummethode 99
5.2.6 Testablauf dokumentieren 101
5.3 Sollwerte aus der Spezifikation 102

5.4 Glass-Box: Testen nach der Codestruktur 104
5.4.1 Maße für Codeüberdeckung 105
5.4.2 Interpretation von Überdeckungsmaßen 107
5.4.3 Objektorientierung und Glass-Box-Test 108
5.5 Testfälle für spezielle Qualitätsaspekte 109
5.5.1 Testfälle in Form von Code 112
5.5.2 Granularität und Reihenfolge von Prüflingen 113
5.5.3 Stresstest, Recovery und Security Tests 115
5.6 Hilfsmittel und Werkzeuge für das Testen 115
5.6.1 Debuggen ist nicht Testen 115
5.6.2 Standardhilfsmittel: Testrahmen 116
5.6.3 Werkzeuge für Glass-Box-Test 116
5.6.4 Sonstige Hilfsmittel und Werkzeuge 118
5.7 Testen von grafischen Oberflächen 118
5.7.1 Sackgasse: System als Ganzes 119
5.7.2 Capture/Replay-Tools 120

6 Usability Engineering 123
6.1 Software und Bedienbarkeit 124
6.2 Usability als Qualitätsaspekt 124
6.2.1 Gute Bedienoberflächen und Qualitätsaspekte 125
6.2.2 Usability definiert sich über Anforderungen 127
6.3 Aspekte der Benutzerfreundlichkeit nach ISO 9241 129
6.4 Bedienbarkeit messen 131
6.5 Konstruktives Usability Engineering 131
6.5.1 Aufgaben im Usability Engineering 132
6.5.2 Kernaufgaben in der Anforderungsklärung 133
6.5.3 Aktivitäten in Entwurf und Entwicklung 135
6.5.4 Acht Goldene Regeln nach Shneiderman 136
6.6 Experten-Evaluationen 137

7 Reviews und Inspektionen 141
7.1 Rollen und Ablauf 142
7.2 Hilfsmittel 147
7.3 Aufwand und Nutzen 151
7.4 Varianten von Reviews 153

8 Formale Verfahren 155
8.1 Prädikatenkalkül und formale Beweise 156
8.1.1 Grundvorgehen und Basiselemente 157
8.1.2 Voraussetzungen aus Anforderung ableiten 158
8.1.3 Verzweigung als Anweisungsart 160
8.1.4 Schleifeninvarianten 161
8.2 Verschiedene Spezifikationsstile 163
8.3 Spezifizieren und Beweisen mit Modellen 166
8.3.1 Ampelanlage als Petrinetz-Beispiel 167
8.3.2 Beweise auf Petrinetzen 170
8.4 Diskussion formaler Techniken 172

9 Konstruktive Qualitätssicherung 173
9.1 Analytisch, organisatorisch, konstruktiv 173
9.2 Maßnahmen, bevor ein Problem auftritt 175
9.2.1 Bewährte Verfahren 176
9.2.2 Bewährte Bestandteile 177
9.2.3 Bewährte Strukturen 180
9.3 Beispiel Cleanroom: Fehler vermeiden 180
9.4 Agile Softwareentwicklung und Qualität 182
9.4.1 Extrem kurze Einführung in eXtreme Programming 182
9.4.2 Die Rolle der Softwarequalität in XP 186
9.4.3 Terminologie und Testarten 187
9.4.4 Testautomatisierung ist unverzichtbar 188
9.4.5 Testcode ist seltener fehlerhaft 189
9.5 Test First: Testen vor Codieren 190
9.5.1 Unterschiede durch Test First 192
9.5.2 Besserer Produktionscode durch Test First 192

10 Das Abenteuer geht weiter 195
10.1 Rückblick 195
10.2 Was es noch zu erkunden gibt 197
10.3 Wann man aufhören soll 197

Literaturverzeichnis 201
Abkürzungen 207
Index 209