

Inhaltsverzeichnis

1	Einleitung	1
Teil I	Softwareentwicklung und Produktivität	5
2	Professionalisierung als Herausforderung	7
2.1	Wie wird heute Software entwickelt?	8
2.1.1	Aktivitäten der Softwareentwicklung	8
2.1.2	Ergebnisse der Softwareentwicklung	10
2.1.3	Methoden der Softwareentwicklung	12
2.1.4	Die Bedeutung von CMMI und Reifegradmodellen	14
2.2	Professionalität auf dem Prüfstand	17
2.2.1	Was ist Professionalität?	17
2.2.2	Verbreitung methodischer Softwareentwicklung	18
2.2.3	Bescheidene Erfolgsquoten	21
2.2.4	Große Unterschiede – Top und Low Performer	23
2.2.5	Ursachen der geringen Professionalität	25
2.3	Übliche Wege der Professionalisierung	26
2.3.1	Proklamation von Wertesystemen	26
2.3.2	Professionalisierung durch Methoden und Prozesse	27
2.3.3	Professionalisierung als Industrialisierung	27
2.3.4	Vision des Software Engineering	28
3	Die Bedeutung der Produktivität für die Softwareentwicklung	31
3.1	Was verstehen wir unter Produktivität?	31
3.2	Produktivität über alles?	34
3.3	Produktivität und Projekterfolg	36
3.4	Produktivität und Qualität	40
3.5	Produktivität und Aufwandsschätzung	43
3.6	Ohne Produktivität keine Professionalisierung	46

Teil II	Produktivität messen und bewerten	49
4	Function Points und andere Maße für Projektergebnisse	51
4.1	Functional Size Measurement nach ISO 14143	51
4.2	Function Points	52
4.3	Die Functional Size typischer Projekte	55
4.4	Use Case Points	56
4.5	Story Points	57
4.6	Weitere Alternativen	57
4.7	Bei aller Kritik: Function Points sind eine gute Wahl	61
5	Kennzahlen für Produktivität und Qualität	63
5.1	Kennzahlen für die Produktivität	64
5.2	Kennzahlen für die Qualität	65
5.3	Weitere Kennzahlen	68
6	Praxis der Messungen	79
6.1	Bewertung der Functional Size	79
6.2	Messung von Aufwand, Kosten und Zeit	88
6.3	Softwarefehler	91
6.4	Erfahrungsdatenbanken	92
6.5	Statistische Auswertung von Kennzahlen	93
7	Produktivitätsunterschiede	97
7.1	Streuung von Produktivität und Qualität	97
7.2	Herausforderung Messgenauigkeit	100
7.3	Vergleichbar – aber nicht gleich	105
7.4	Benchmarking	108
Teil III	Produktivitätsfaktoren	115
8	Bad Practices – Was bremst uns?	117
8.1	Unschärfe Ziele und Ziellosigkeit	117
8.2	Unrealistische Projekte	118
8.3	Echte Zeitverschwendungen	121
8.4	Ungeeignetes oder unklares Vorgehen	122
8.5	Unklare und instabile Anforderungen	123

8.6	Schlechte Qualität und viele Nacharbeiten	124
8.7	Unerfahrene oder ungeeignete Projektleiter	124
8.8	Fehlende Skills im Team	125
8.9	Wenig Motivation	125
8.10	Chinesenprinzip	126
8.11	Viele 50%-Ressourcen	127
8.12	Zu viel Politik, zu wenig Realismus	127
9	Produktivitätsfaktoren der Softwareentwicklung	129
9.1	Produktivitätsfaktoren im Überblick	129
9.2	Produktivitätsfaktoren und Produktivitätshebel	135
9.3	Welche Produktivitätsfaktoren sind relevant?	137
10	Die acht elementaren Produktivitätsfaktoren	139
10.1	Ein einfaches Modell produktiver Prozesse	139
10.2	Berücksichtigung von Qualität und Rework	142
10.3	Berücksichtigung von Verschwendungen	143
10.4	Die acht Gebote der Produktivität	145
Teil IV	Produktiver werden	147
11	Die Macht der Ziele nutzen	149
11.1	Die Richtungen der Beschleunigung	149
11.2	Klares Projektziel voranstellen	151
11.3	Explizite Projektdurchführungsziele setzen	152
11.4	Ziel-Commitments vereinbaren	153
11.5	Erreichbare Zwischenziele setzen	154
11.6	Zug zum Ziel durch festen Steuerrhythmus	155
11.7	Zielorientierung auch im Kleinen einfordern	157
12	Produktive Hochleistungsteams aufbauen	159
12.1	Die Richtungen der Beschleunigung	159
12.2	Die richtige Teamgröße bestimmen	160
12.3	Teams effektiv organisieren	165
12.4	Den passenden Skillmix finden	167
12.5	Gute Projektleiter als Taktgeber finden	168

12.6	Individuelle Leistungsfähigkeit erkennen	170
12.7	Motivation erzeugen und erhalten	171
12.8	Ein professionelles Wertesystem aufbauen	173
13	Den Kern der richtigen Anforderungen treffen	175
13.1	Die Richtungen der Beschleunigung	175
13.2	Anforderungen iterativ analysieren	177
13.3	Aktive Benutzerbeteiligung einfordern	178
13.4	Prototyping statt Papiertiger	180
13.5	Überproduktion vermeiden	183
14	Vorgehen ohne effektive Methodik abstellen	187
14.1	Die Richtungen der Beschleunigung	187
14.2	Methodisch vorgehen	188
14.3	Best Practices sammeln	190
14.4	Wasserfall vermeiden	193
14.5	Methodendisziplin erreichen	195
14.6	Automatisierungen selektiv einsetzen	196
14.7	Wiederverwendung auf verschiedenen Ebenen nutzen	198
15	Qualität steigern und Rework radikal reduzieren	201
15.1	Die Richtungen der Beschleunigung	201
15.2	Qualitätsbewusstsein durch Qualitätsbegriff schaffen	202
15.3	Anzahl ausgelieferter Fehler reduzieren	203
15.4	Rework-Anteil gering halten	204
15.5	Qualitätssicherung einfach organisieren	205
15.6	Mit Projektaudits Sackgassen erkennen	206
15.7	Codereviews sind effektiver als Tests	208
15.8	Frühes iteratives Testen	209
16	Verschwendungen erkennen und eliminieren	211
16.1	Die Richtungen der Beschleunigung	211
16.2	Angemessenheit zum Prinzip erklären	212
16.3	Pragmatisch dokumentieren	213
16.4	Effektive Meetings durchführen	215
16.5	Organisatorische Verschwendungen vermeiden	216
16.6	Rechtzeitig Hilfe holen	218

17	Projekte richtig in die Umgebung integrieren	219
17.1	Die Richtungen der Beschleunigung	219
17.2	Projekt schützen	220
17.3	Stakeholder aktiv managen	221
17.4	Projekte durch Marketing stärken	222
17.5	Topmanagementfokus nutzen	223
18	Fortschritt, Qualität und Produktivität steuern	225
18.1	Die Richtungen der Beschleunigung	225
18.2	Richtige Rahmenbedingungen schaffen	226
18.3	Projektfortschritt kontrollieren	228
18.4	Qualität messen und bewerten	231
18.5	Produktivität messen und bewerten	232
19	Zusammenfassung und Ausblick	233
Anhang		237
<hr/>		
	Literaturverzeichnis	239
	Index	245