

Inhalt

Vorwort zur 4. Auflage	XV
1 Einführung	1
1.1 Warum dieses Buch?	2
1.2 Struktur und Aufbau	3
1.3 Dankeschön	5
1.4 Feedback	6
2 Beispiel: Scrumcoaches.com	7
2.1 Das Projekt	8
2.2 Der Entwicklungsprozess	9
2.3 Die Beteiligten	10
2.4 Die Anforderungen	10
2.5 Priorisieren und Schätzen des Product Backlog	12
2.5.1 Priorisieren	12
2.5.2 Schätzen	13
2.6 Sprint-Planung	14
2.6.1 Sprint-Ziel	14
2.6.2 Entwicklungsgeschwindigkeit	15
2.6.3 Analyse der User Stories	15
2.6.4 Design der User Stories	16
2.7 Sprint-Durchführung	16
2.8 Messen des Sprint-Fortschritts	19
2.9 Am Ende des Sprint	20
2.9.1 Sprint-Review	20
2.9.2 Sprint-Retrospektive	21
2.10 Die Arbeit geht weiter	22
2.11 Zusammenfassung	23
2.12 Wie geht es weiter?	23

3	Die Grundlagen von Scrum	24
3.1	Was ist Scrum?	24
3.2	Scrum, ein Framework?	26
3.3	Überblick	27
3.3.1	Scrum-Team	27
3.3.2	Vision und Product Backlog	27
3.3.3	Sprint Planning Meeting	28
3.3.4	Sprints	29
3.3.5	Daily Scrums	29
3.3.6	Sprint-Review	29
3.3.7	Sprint-Retrospektive	30
3.4	Prinzipien	30
3.4.1	Transparenz	30
3.4.2	Beobachten und Anpassen	31
3.4.3	Timeboxing	31
3.4.4	Dinge abschließen	32
3.4.5	Maximierung von Geschäftswert	33
3.4.6	Teams scheitern nicht	34
3.4.7	Ergebnisorientierung	34
3.5	Die Rollen	35
3.5.1	Das Team	36
3.5.2	Der ScrumMaster	37
3.5.2.1	Dienstleistender Anführer und Problembeseitiger	37
3.5.2.2	Scrum implementieren	38
3.5.2.3	Entscheider	38
3.5.2.4	Müssen ScrumMaster programmieren können?	38
3.5.2.5	Product Owner-Coaching	39
3.5.2.6	Belastbare Persönlichkeit	39
3.5.2.7	Scrum in der Organisation verbreiten	39
3.5.3	Der Product Owner	40
3.5.3.1	Den Kunden repräsentieren	41
3.5.3.2	User Stories und Product Backlog	41
3.5.3.3	Mit dem Team durch den Sprint	42
3.5.3.4	Bestimmen, wann was fertig ist	42
3.5.4	Nebenrolle Kunde	42
3.6	Die ideale Arbeitsumgebung	44
3.7	Empirisches Management	44
3.8	Zusammenfassung	46
3.9	Wie geht es weiter?	47

4	User Stories	48
4.1	Was sind User Stories?	49
4.1.1	Story-Karte	50
4.1.2	Konversation	51
4.1.3	Akzeptanzkriterien	51
4.2	Warum User Stories?	52
4.3	User Stories schreiben	53
4.3.1	Die Sprache des Kunden	54
4.3.2	Benutzerrollen	54
4.3.3	User-Story-Muster	56
4.3.4	Epics	56
4.3.5	Themen	58
4.3.6	Wie viel Detail?	59
4.3.7	Keine Technik	60
4.3.8	Keine Benutzeroberfläche	60
4.4	Eigenschaften guter User Stories	60
4.4.1	Independent – unabhängige User Stories	60
4.4.2	Negotiable – verhandelbare User Stories	61
4.4.3	Valuable – wertvolle User Stories	61
4.4.4	Estimatable – schätzbare User Stories	62
4.4.5	Small – kleine User Stories	62
4.4.6	Testable – testbare User Stories	63
4.5	Zusammenfassung	64
4.6	Wie geht es weiter?	64
5	Agiles Schätzen	65
5.1	Was ist agiles Schätzen?	66
5.1.1	Relative Größe statt Dauer	66
5.1.2	Schätzen in Story Points	67
5.1.3	Wo bleibt die Dauer?	68
5.1.4	Argumentationshilfe für Story Points	68
5.2	Schätzen von User Stories	69
5.2.1	Größenordnungen und Punktesequenzen	70
5.2.2	Planungspoker	71
5.2.2.1	Schätzen im Team	74
5.2.2.2	Referenz-Story und Triangularisierung	74
5.2.2.3	Planungspoker funktioniert	76
5.2.3	Wann schätzen?	76
5.3	Zusammenfassung	77
5.4	Wie geht es weiter?	77

6	Agiles Planen	78
6.1	Was macht Planung agil?	78
6.2	Velocity	80
6.2.1	Tatsächliche Velocity	80
6.2.2	Angenommene Velocity	81
6.2.2.1	Angenommene Velocity = Tatsächliche Velocity	82
6.2.2.2	Mittlere Velocity	83
6.2.3	Velocity-basierte Planung	84
6.2.4	Nachhaltige Velocity	85
6.3	Agile Planung funktioniert	87
6.3.1	Velocity korrigiert Schätzfehler	87
6.3.2	Neubewertung von User Stories	88
6.3.3	Urlaub, Krankheit und ähnliche Ereignisse	89
6.3.4	Der Plan entsteht	89
6.4	Zusammenfassung	90
6.5	Wie geht es weiter?	90
7	User Stories fürs Product Backlog	91
7.1	Das Product Backlog	91
7.2	Das Product Backlog füllen	93
7.2.1	Anforderungswshops	95
7.2.2	Interviews, Markt-Feedback und Abstimmungsrunden	96
7.2.3	Überarbeitung und Pflege des Product Backlog	97
7.3	User Stories priorisieren	98
7.3.1	Finanzieller Wert	98
7.3.2	Kosten	99
7.3.3	Kundenzufriedenheit nach Kano	100
7.3.4	Risiko	101
7.3.5	Abhängigkeiten	102
7.3.6	Priorisierende Faktoren abwägen	102
7.3.7	MuSCoW-Priorisierung	103
7.4	User Stories schneiden	104
7.4.1	Vertikales Schneiden	104
7.4.2	Schneiden nach Daten	106
7.4.3	Schneiden nach Aufwand	106
7.4.4	Schneiden nach Forschungsanteilen	107
7.4.5	Schneiden nach Qualität	108
7.4.6	Schneiden nach Benutzerrolle	108

7.4.7	Schneiden nach Akzeptanzkriterien	109
7.4.8	Schneiden nach technischer Voraussetzung	110
7.5	Andere Anforderungen	110
7.5.1	Anforderungen umformulieren	111
7.5.2	Constraints	111
7.5.3	Fehler.....	112
7.5.4	Technisches Backlog	113
7.6	Zusammenfassung.....	114
7.7	Wie geht es weiter?	114
8	User Story Mapping	115
8.1	User Story Maps	116
8.2	Eine Story Map erstellen	117
8.2.1	Schritt 1: User Tasks ermitteln	118
8.2.2	Schritt 2: Gruppen bilden – User Activities.....	119
8.2.3	Schritt 3: Ordnung schaffen	119
8.2.4	Schritt 4: User Tasks durchlaufen = Geschichten erzählen	120
8.2.5	Schritt 5: User Stories schreiben.....	121
8.3	Warum Story Mapping?	122
8.3.1	Basis für gute Product Backlogs	122
8.3.2	Kleinstmögliche Releases.....	123
8.3.3	Motivation und Einsicht für alle Stakeholder	123
8.3.4	Lückenlosigkeit	123
8.3.5	Softwarearchitektur	123
8.3.6	Multi-Team-Setups	124
8.4	Von der Story Map zum Product Backlog	124
8.4.1	User Stories schreiben	126
8.4.2	Die Story Map ersetzt das Product Backlog	127
8.5	Zusammenfassung.....	127
8.6	Wie geht es weiter?	128
9	Sprint-Planung	129
9.1	Überblick und Ablauf.....	129
9.2	Beteiligte	130
9.3	Ergebnisse.....	130
9.4	Vorbereitung	133
9.4.1	Sprint Velocity.....	133
9.4.1.1	Anpassen der Velocity	133
9.4.1.2	Bugfixing, Refactoring und andere Aufgaben	134

9.4.2	Story-Auswahl.....	135
9.4.3	Sprint-Länge	135
9.5	Sprint Planning 1	137
9.5.1	Ablauf	137
9.5.2	Sprint-Ziel – warum führen wir den Sprint durch?.....	138
9.5.3	Vorstellung, Analyse und Commitment	138
9.5.4	Fehler und andere technische Aufgaben	140
9.6	Sprint Planning 2.....	141
9.6.1	Ablauf	142
9.6.2	Story-Design.....	142
9.6.3	Tasks schneiden	144
9.6.3.1	Taskgröße.....	145
9.6.3.2	Schneidetechniken.....	145
9.6.3.3	Ungeplante Tasks.....	146
9.6.4	Tasks schätzen?	146
9.6.4.1	Taskschätzungen sind sinnvoll	147
9.6.4.2	Taskschätzungen sind unsinnig	147
9.6.4.3	Keine Empfehlung	148
9.6.5	Das Sprint Backlog	149
9.6.6	Fehler und andere technischen Aufgaben verteilen	150
9.6.7	Was tun, wenn es länger wird?	150
9.7	Abschluss.....	151
9.8	Zusammenfassung.....	152
9.9	Wie geht es weiter?	152
10	Sprint-Durchführung	153
10.1	Die eigentliche Arbeit beginnt	153
10.2	Wer macht was?	155
10.2.1	Das Team	155
10.2.2	Der Product Owner.....	156
10.2.3	Der ScrumMaster	156
10.3	Story für Story Richtung Sprint-Ziel	157
10.3.1	Wie viele User Stories zurzeit?	158
10.3.2	Arbeit an einer User Story	158
10.3.3	Definition of Done	158
10.3.4	Abnahme fertiger User Stories	159
10.3.4.1	Entwicklertest.....	159

10.3.4.2	Akzeptanztest	160
10.3.4.3	QA-Abnahme	160
10.3.4.4	Frühestmögliche Fehlerbehebung.....	161
10.4	Daily Scrum	161
10.4.1	Aktualisierung des Taskboard.....	162
10.4.2	Ein guter Zeitpunkt.....	163
10.4.3	Ein guter Ort.....	164
10.4.4	Wer ist noch dabei?	164
10.4.5	Was macht der ScrumMaster?	165
10.5	Unterbrechungen	165
10.6	Messen und Anpassen.....	167
10.6.1	Bug- und technische Burndown-Charts.....	168
10.6.2	Was tun, wenn es eng wird?	168
10.7	Reguläres Sprint-Ende.....	170
10.8	Sprint-Review.....	171
10.8.1	Vorbereitung	171
10.8.2	Ort, Zeitpunkt und Teilnehmer	171
10.8.3	Ziel.....	171
10.8.4	Ablauf	172
10.9	Das Team organisiert sich.....	172
10.9.1	Verantwortung übernehmen.....	173
10.9.2	Das Team machen lassen und aus Fehlern lernen	173
10.9.3	Den Product Owner einbeziehen.....	174
10.9.4	Software-Pull-Systeme	174
10.9.5	Das Team bei der Arbeit mit Tasks coachen	175
10.9.6	Einzelgespräche	176
10.10	Sprint Best Practices	177
10.10.1	Source Code Management und Story-Branches	177
10.10.2	Kontinuierliches Integrieren	178
10.10.3	Automatisierung	178
10.10.4	Verständlicher Quellcode.....	178
10.10.5	Elektronische Sprint Backlogs und Burndown-Charts	179
10.11	Zusammenfassung.....	179
10.12	Wie geht es weiter?.....	180
11	User Stories Akzeptanztesten.....	181
11.1	Was ist Akzeptanztesten?.....	181

11.1.1	Akzeptanzkriterien	182
11.1.1.1	Akzeptanzkriterien sind Erwartungen	182
11.1.1.2	Akzeptanzkriterien sind Geschäftsregeln	183
11.1.2	Akzeptanztests	183
11.1.3	Akzeptanztesten	184
11.2	Akzeptanzkriterien schreiben	184
11.2.1	Vom Akzeptanzkriterium zum Akzeptanztest	185
11.2.2	Merkmale guter Akzeptanzkriterien	186
11.2.3	Akzeptanzkriterien auch für Epics?	187
11.3	Beispiel: Suche nach Coaches	188
11.4	Kleine Bausteine: Auf dem Weg zur DSL.....	189
11.5	Akzeptanztesten während des Sprint	190
11.6	Die hohe Schule: Akzeptanztest-getriebene Entwicklung.....	192
11.6.1	ATDD-Beispiel: Suche nach Coaches	193
11.6.2	Product Owner love writing Tests?	194
11.6.2.1	Alternative JCriteria	194
11.7	Lohnt sich das Ganze?.....	195
11.8	Zusammenfassung.....	196
11.9	Wie geht es weiter?	196
12	Sprint-Retrospektive	197
12.1	Nach dem Sprint ist vor dem Sprint	198
12.2	Ablauf von Retrospektiven	198
12.3	Retrospektiven vorbereiten	200
12.4	Retrospektiven leiten	200
12.5	Agenda und Check-in	201
12.6	Phase 1: Daten sammeln	202
12.6.1	Erstellung einer Timeline.....	203
12.6.2	Erweiterung der Timeline um Energiepunkte	204
12.7	Phase 2: Einsichten generieren.....	204
12.7.1	Positiv/Delta-Liste.....	205
12.7.2	Warum-Fragen	205
12.8	Phase 3: Entscheiden, was zu tun ist	206
12.9	Phase 4: Ziele formulieren und Aktionen planen	207
12.10	Abschluss.....	208
12.11	Themenorientierte Retrospektiven	208
12.12	Zusammenfassung.....	210
12.13	Wie geht es weiter?	210

13	Agile Releaseplanung	211
13.1	Releaseplanung	211
13.1.1	Themen-Releases	211
13.1.2	Datum-Releases	212
13.1.3	Releaseplanungs-Workshop	213
13.1.4	Was macht die Planung agil?	213
13.2	Planungs-Velocity	214
13.2.1	Durchführung von Test-Sprints	214
13.2.2	Historische Daten	214
13.2.3	Das Team bestimmen lassen	215
13.2.4	Auswahl eines Verfahrens	215
13.3	Der Releaseplan	216
13.4	Sichere Planung	217
13.4.1	Sichere Velocity	217
13.4.2	Sicherheit durch weniger wichtige User Stories	218
13.5	Monitoring und Aktualisierung	219
13.6	Zusammenfassung	220
13.7	Wie geht es weiter?	220
14	Mobiles Arbeiten	221
14.1	Herausforderungen	221
14.2	Start ins mobile Arbeiten	223
14.3	Mobiles Arbeiten in Scrum	224
14.3.1	Werkzeuge	225
14.3.1.1	Das digitale Whiteboard	225
14.3.1.2	Das digitale Taskboard	226
14.3.1.3	Mobiles Pair Programming	226
14.3.2	Meetings	226
14.3.2.1	Mobiler Start-Workshop	227
14.3.2.2	Mobiles Daily Scrum	227
14.3.2.3	Mobile Sprint-Planung	228
14.3.2.4	Mobile Retrospektive	228
14.3.2.5	Beispiel für ein Retrospektive Board	229
14.3.2.6	Mobiles Review	233
14.4	Die Zukunft: hybrides Arbeiten	233
14.4.1	Hybrid starten	234
14.4.2	Hybrid im Alltag	235
14.5	Zusammenfassung	235
14.6	Wie geht es weiter?	236

15	Verticals – SCRUM@OTTO	237
15.1	Warum ich über diese Geschichte schreibe	237
15.2	Die Vorgeschichte	239
15.3	Das Lhotse-Projekt – Zahlen, Daten, Fakten	240
15.4	Das Team – Menschen im Mittelpunkt.....	241
15.5	Triaden – die Führung eines Teams	243
15.6	Die Triade – Rollenbeschreibungen	243
15.6.1	Der Projektmanager – Project-Lead.....	244
15.6.2	Der Produktmanager – Business-Designer	244
15.6.3	Der Team-Architekt – Technical-Designer	245
15.7	Die TD-Runde	246
15.8	Die Otto-Architektur in Vertikalen.....	248
15.8.1	Warum die klassische IT versagt.....	248
15.8.2	Warum vertikale Schnitte helfen	251
15.8.3	Was eine Vertikale ist	252
15.8.4	Wie vertikale Schnitte gefunden werden können	253
15.9	Makro- und Mikroarchitektur	256
15.9.1	Makroarchitektur	256
15.9.2	Mikroarchitektur.....	257
15.10	Werte und Leitplanken statt Richtlinien und Governance	257
15.11	Das klassische Management in der agiler werdenden Organisation.....	258
15.12	Scrum@Otto – 100 Sprints später	259
15.13	Fazit	262
	Glossar	263
	Literatur	271
	Stichwortverzeichnis.....	273