

Bernd Oestereich · Christian Weiss

APM – Agiles Projekt- management

Erfolgreiches Timeboxing für IT-Projekte

→ Unter Mitarbeit von Oliver F. Lehmann und Uwe Vigerschow

oose.
Innovative Informatik

dpunkt.verlag

Vorwort

Projektmanagement ist der branchen- und technologieübergreifende Erfolgsfaktor für die Entwicklung von Systemen und Produkten. Von kleinen Projektarbeiten in allgemeinbildenden Schulen bis zu den größten Vorhaben der Menschheit – es wimmelt in unserer Welt von Projekten. Projektmanagementfähigkeiten haben also beinahe grundlegende gesellschaftliche Bedeutung.

Die vordergründige Aufmerksamkeit bei Projekten richtet sich fast immer auf den inhaltlichen, wirtschaftlichen und terminlichen Erfolg. Die Projektleitung ist dafür verantwortlich und steht somit unter besonderem Druck.

Verantwortungsbewusste junge oder neue Projektleiter und -leiterinnen blicken häufig mit Ehrfurcht auf KollegInnen mit viel Erfahrung aus großen Projekten. Die Expertise und Erfahrung erfolgreicher ProjektleiterInnen lassen sich ganz sicher auch nicht in einem oder vielen Büchern vermitteln. Dafür ist die Aufgabe zu komplex. Andererseits kochen auch die Topleute nur mit Wasser. An Ratschlägen, Methoden, Anleitungen, Büchern und Seminaren mangelt es auch nicht, die meisten sind durchaus hilfreich. Dennoch: Bestimmte Fähigkeiten müssen eingeübt werden. Die persönliche Befähigung, all die klugen Ideen praktisch sinnvoll anzuwenden, entwickelt sich schrittweise.

Mit diesem Buch stellen wir eine bewährte und auf verschiedene Projektgrößen skalierbare Projektmanagementmethodik vor, das sogenannte APM-Verfahren. Wir möchten einerseits soweit möglich im Mainstream bleiben, uns also an gegebene Standards anlehnen. Andererseits möchten wir aber auch innovative Ideen agiler Projektführung einbringen. Dazu gehört das von oose seit Mitte/Ende der 1990er-Jahre (weiter-)entwickelte Timeboxing-Verfahren mit vielen kleinen nützlichen Techniken drum herum. Dies beinhaltet auch Ideen aus anderen mittlerweile gut verbreiteten agilen Ansätzen.

Was uns aber besonders wichtig ist: Wir beschreiben einen methodischen Kern (Timeboxing + Features), der mit vielen anderen bewährten Praktiken kombinierbar ist – diese Praktiken haben wir aus ihrem jeweiligen Kontext herausgelöst und in den letzten Jahren in einem „PM-Werkzeugkasten“ gesammelt. Hier finden sich viele Ideen und Erfahrungen erfolgreicher ProjektleiterInnen wieder. Die rund hundert wichtigsten Praktiken bzw. Werkzeuge haben wir für dieses Buch aufbereitet. Viele davon sind ebenso essenziell wie unspektakulär. Erfahrene ProjektmanagerInnen kennen wahrscheinlich viele dieser oder ähnliche Techniken. Allen anderen möchten wir damit Anregungen geben, schneller und erfolgreicher ihre eigenen Erfahrungen zu sammeln.

Unser besonderer Dank gilt zuallererst den vielen Hundert Teilnehmern unserer Projektmanagementschulungen der letzten Jahre und den ungezählten Menschen, denen wir im Rahmen von Coachings in konkreten Projekten begegnet sind, die uns mit ihren Fragen immer wieder herausgefordert haben und die auch immer wieder durch eigene Ideen mit dazu beigetragen haben, das APM-Verfahren weiterzuentwickeln. Des Weiteren bedanken wir uns bei allen oose-Kollegen, vor allem bei denen im Bereich Projektmanagement tätigen, für ihre hervorragende Unterstützung. Namentlich unbedingt zu erwähnen, weil sie die Buchmanuskripte teilweise immer wieder durchgegangen sind, sind hier Uwe Vigenschow und Michael Schulze-Ruhfus. Von Uwe Vigenschow stammen einige Textpassagen zum Thema Aufwandschätzungen und zu psychosozialen Themen. Von Oliver Lehmann stammen alle PMBoK-nahen Texte vor allem aus Kapitel 9.

Frau Dr. Heidi Heilmann, Dr. Hartmut Krasemann und Jutta Eckstein haben neben weiteren anonymen Gutachtern durch sehr konkrete Rückmeldungen zum Manuskript einen großen Verdienst daran, dass wir kurz vor Fertigstellung des Buches die Gliederung noch einmal komplett überarbeitet haben, die Durchgängigkeit in der Terminologie und des Fallbeispiels verbessern konnten und viele viele kleine Ungenauigkeiten erkennen und größtenteils auch beseitigen konnten. Vor allem Jutta Eckstein hat viel Arbeit investiert und mit starkem Fokus auf Agilität viele kritische Anmerkungen geliefert. Hartmut Krasemann gebührt insofern noch besonderer Dank, als dass er mir schon Mitte der 1990er-Jahre zu meinen ersten eigenen praktischen Projektleitungserfahrungen mit iterativen und agilen Verfahren in einem (letztendlich gescheiterten) Megaprojekt verholfen hat.

Dennoch bleibt darauf hinzuweisen, dass sehr viele verschiedene konkrete Einflüsse zusammengekommen sind, die zusammenzubringen im Detail gar nicht so einfach war. Als wir mit dem Buch anfangen, waren wir bei oose der Meinung, über eine einheitliche agile Projektmanagementmethodik zu verfügen. Die Arbeit an dem Buch hat aber gezeigt, dass wir bereits im Kernteam von Christian Weiss, Bernd Oestereich, Uwe Vigenschow, Markus Wittwer und Michael Schulze-Ruhfus in den letzten 10 Jahren in verschiedenen Projekten so viele unterschiedliche Erfahrungen gesammelt haben und verschiedene Ausprägungen, aber auch Tipps und Tricks entwickelt hatten, dass wir selbst zunächst noch einmal viel dazugelernt haben, bevor wir alle diese Ideen und Strömungen halbwegs zusammenbringen konnten. Insofern ist auch dieses Buch sicherlich nur ein erstes noch weiterzuentwickelndes Release.

Für den jetzt bevorstehenden ersten Akzeptanztest interessieren uns deshalb natürlich Ihre persönlichen Erfahrungen mit dem Thema sowie auch Ihre persönlichen bewährten Techniken. Wir laden Sie daher zur Diskussion mit uns in der Mailing-Gruppe *apm-buch* ein (<http://de.groups.yahoo.com/group/apm-buch>, nähere Infos auch auf unserer Website www.oose.de/apm).

Bernd Oestereich

Inhaltsüberblick

1	Einleitung	1
	Komprimierter Überblick über das APM-Verfahren • Entstehung, Motivation und die Werte agiler Softwareentwicklung • Organisatorische Besonderheiten verschiedener Projektgrößen • Hinweise zur Einführung und Adaption des APM-Verfahrens	
2	Projektvorbereitung	63
	Vor Projektbeginn die Weichen richtig stellen • Interessen und Rahmenbedingungen von Auftragnehmer und Auftraggeber klären • Preis- und Vertragsmodelle • Was ist featurebasiertes Vorgehen und warum es wichtig ist	
3	Projektplanung	99
	Projekt in seiner Gesamtheit planen • Was zu welchen Zeitpunkten wichtig ist • Nutzen einer Phasengliederung • Meilensteine in nützlicher Weise mit Iterationen (Timeboxen) kombinieren • Releases planen	
4	Releaseplanung	133
	Wie ein Iterationsplan entsteht • Anforderungen priorisieren • Iterationsfeatures planen • Wie Features beschrieben werden • Iterationsfeatures als Iterationsziele	
5	Iterationsvorbereitung	157
	Iteration im Detail vorbereiten • Teams bereiten ihre Arbeitsaufträge vor • Qualitätssicherung und Klärung teamübergreifender Ressourcenkonflikte	
6	Iterationsdurchführung	175
	Mikroprozess einer Iteration • Wie Teammitglieder, Team- und Projektleitungen täglich, wöchentlich und iterationsweise genau das Feedback erhalten, das sie für die Steuerung der Iteration benötigen • Innerhalb einer Iteration koordinieren	
7	Iterationsnachbereitung	195
	Was am Ende einer Iteration passiert • Ergebnisse messen und auswerten • Erkenntnisse für die nächste Iteration verwenden • Entwicklungsprozess kontinuierlich verbessern	
8	Release- und Projektabschluss	207
	Vorabnahmen, Teilabnahmen und Endabnahmen und was sie mit Releases zu tun haben • Frühzeitig vertragsrelevante Abnahmen erzielen • Sonstige Aspekte zum Projektende	
9	Techniken, Muster, Modelle, Standards	225
	Grundsätzliche Wissensgebiete des Projektmanagements • Internationale methodische Projektmanagementstandards (am Beispiel PMI) • Einzeltechniken, Muster und Modelle im Kontext agiler Projekte	
10	Anhang	415

Inhalt

1	Einleitung	1
1.1	Das APM-Verfahren im Überblick	3
1.2	Agile Softwareentwicklung	12
1.2.1	Historische Missverständnisse	12
1.2.2	Entstehung und Motivation	14
1.2.3	Das agile Manifest	15
1.2.4	Prinzipien agiler Softwareentwicklung	17
1.2.5	Bezug zu anderen agilen Verfahren	18
1.3	Werte	20
1.3.1	Was bestimmt die Werte in einem Projekt?	20
1.3.2	Werte als Vorbedingung für Agilität?	21
1.3.3	Führung als Dienstleistung	21
1.3.4	Eigenverantwortliches Arbeiten	22
1.3.5	Arbeits- und Verantwortungsteilung	24
1.3.6	Macht	27
1.3.7	Vertrauen	28
1.3.8	Zwischen Freiheit und Vorschrift	28
1.4	Projektorganisation und Projektgrößen	30
1.4.1	Kleinprojekt	30
1.4.2	Mittleres Projekt	31
1.4.3	Großprojekt	33
1.4.4	Megaprojekt	34
1.4.5	Wie große Projekte organisiert sind	35
1.5	Einführung und Adaption des Verfahrens	41
1.5.1	Populäre Missverständnisse	42
1.5.2	Abgrenzung zum Wasserfallansatz	45
1.5.3	Faktoren für erfolgreiche Projekte	47
1.5.4	Mit oder ohne Features planen?	50
1.5.5	Beweglichkeit systematisch maximieren	52
1.5.6	Iterationsdauer	56
1.5.7	Synchrone Iterationen	59
1.5.8	Iteratives Vorgehen in der Linienorganisation?	61
2	Projektvorbereitung	63
2.1	Fallbeispiel	65
2.2	Was vor dem Projektstart wichtig ist	66
2.2.1	Typische Fehler vor Projektstart	66
2.2.2	Die Weichen richtig stellen	68
2.3	Ziele, Risiken und Rahmenbedingungen klären	70
2.3.1	Auftragnehmerseitige Zielklärung	70
2.3.2	Auftraggeberseitige Zielklärung (Systemidee, Auftragsklärung)	72
2.3.3	Risiken identifizieren, Risikostrategie entwickeln	73

2.4	Kosten, Termine und Vertragsbedingungen bestimmen	74
2.4.1	Aufwandschätzungen	74
2.4.2	Schätztechniken	77
2.4.3	Schätzmethoden	77
2.4.4	Preis- und Vertragsmodelle	81
2.5	Featurebasiertes Vorgehen	83
2.5.1	Warum und wann featurebasiertes Planen sinnvoll ist	83
2.5.2	Features sind kaufentscheidende Leistungsmerkmale	86
2.5.3	Wie man Features findet	88
2.5.4	Arten von Features	92
2.5.5	Features versus Anwendungsfälle	94
3	Projektplanung	99
3.1	Phaseneinteilung trotz Iterationen?	101
3.1.1	Zusammenhang von Iterationen, Phasen und Meilensteinen	102
3.1.2	Entwicklungsphasen im Überblick	105
3.1.3	Zeitliche Verteilung der Phasen	107
3.1.4	Objektive Überprüfung von Meilensteinen	108
3.1.5	Teamdynamik	111
3.1.6	Einarbeitung neuer Mitarbeiter	112
3.1.7	Startphase	113
3.1.8	Hauptphase	118
3.1.9	Abschlussphase	119
3.1.10	Betriebsphase	121
3.2	Die Projektebene planen	122
3.2.1	Releases schneiden	123
3.2.2	Das Projekt mit Releases und Meilensteinen planen	125
3.2.3	Parallelreleases planen	130
4	Releaseplanung	133
4.1	Wie man Features plant	135
4.1.1	Iterationsplan aufstellen	137
4.1.2	Iterationskapazitäten bestimmen	139
4.1.3	Anwendungsfälle priorisieren	142
4.1.4	Iterationsfeatures planen	144
4.1.5	Planung justieren	148
4.1.6	Features beschreiben	149
4.1.7	Der Iterationsplan als Zielvereinbarung	151
4.1.8	Projekthalt mit Features strukturieren	153
5	Iterationsvorbereitung	157
5.1	Grundsätzlicher Aufbau einer Iteration	159
5.2	Arbeitsvorbereitung	161
5.2.1	Aufgaben für Iteration i+2 planen (Grobplanung)	162
5.2.2	Arbeitsaufträge für Folgeiterationen auswählen	165
5.2.3	Arbeitsvorbereitung der Teams	165
5.2.4	Arbeitsaufträge für Iteration i+1 planen (Feinplanung)	166

5.2.5	Qualitätssicherung durch die Teamleitung.....	169
5.2.6	Projektleitung sichtet detaillierte Arbeitsaufträge	170
5.2.7	Klärung kritischer Arbeitsaufträge durch die Projektleitung	171
5.2.8	Engpassanalyse.....	172
5.2.9	Der Warteraum	173
5.2.10	Weiter gehende Qualitätssicherungsmaßnahmen.....	173
6	Iterationsdurchführung	175
6.1	Iterationsauftakt	177
6.2	Der Iterations-Mikroprozess	178
6.2.1	Tägliches Smoke-Build	181
6.2.2	Tägliches Teamsteuerungstreffen	182
6.2.3	Wöchentliches Integrationsbuild	184
6.2.4	Wöchentliches Mikrocontrolling	185
6.2.5	Der Läufer	189
6.3	Ergebnisse abschließen	189
6.3.1	Timeboxing	190
6.3.2	Iterationsfeaturereviews.....	192
6.3.3	Störungsfreie Iteration.....	193
7	Iterationsnachbereitung	195
7.1	Iteration auswerten	197
7.1.1	Arbeitsauftragsreviews.....	198
7.1.2	Retrospektive	201
7.2	Planungskorrektur	202
7.2.1	Auswertung der Reviewergebnisse	203
7.2.2	Interne Beauftragung der Arbeiten für die nächste Iteration	204
7.2.3	Restrukturierung und planlose Weiterentwicklung.....	204
8	Release- und Projektabschluss	207
8.1	Vorbereitung des Lenkungskreises	209
8.2	Internes Controlling.....	211
8.2.1	Mikrocontrolling auf Arbeitsauftragsebene.....	211
8.2.2	Die Unvergleichbarkeit von Makro- und Mikroschätzungen.....	216
8.2.3	Planungspuffer	219
8.3	Abnahmen	220
8.3.1	Vorabnahmen und Teilabnahmen.....	220
8.3.2	Releaseentwicklung	221
8.3.3	Optional: Stabilisierungsiteration (Endgame).....	223
9	Techniken, Muster, Modelle, Standards	225
9.1	Überblick	227
9.1.1	PMBok Guide – ein De-facto-Standard	227

9.2	Der methodische Ansatz des PMBoK Guide	229
9.3	Integrationsmanagement	234
9.4	Inhalts- und Umfangsmanagement	236
9.4.1	PMBoK-Rahmenwerk	236
9.4.2	Produktkarton	238
9.4.3	Zieldefinition mit ZAK-Karten	241
9.4.4	Zielbeschreibungsschema	243
9.4.5	Projektstrukturplan	245
9.4.6	Anforderungsworkshop	246
9.4.7	Featurebasiertes Planen	249
9.4.8	Priorisierungsworkshop	252
9.4.9	Großgruppen-Priorisierung	252
9.4.10	Meilenstein	253
9.4.11	Meilensteinvereinbarung	255
9.4.12	Änderungsantrag	257
9.4.13	Kano-Innovationsmodell	258
9.4.14	Abnahmespezifikation	260
9.4.15	Exploratives Prototyping	261
9.4.16	Projektname	263
9.5	Zeitmanagement	264
9.5.1	PMBoK-Rahmenwerk	264
9.5.2	Planungseinheiten	265
9.5.3	Planungsworkshop	267
9.5.4	Planungsspiel	271
9.5.5	Planungswand	272
9.5.6	Arbeitsauftrag	274
9.5.7	Delphi-Methode (Schätztechnik)	276
9.5.8	Dreipunktschätzung	278
9.5.9	Schätzkurve	280
9.5.10	Widget-Point-Verfahren	281
9.5.11	Use-Case-Point-Methode	283
9.5.12	Wetter-von-gestern	286
9.5.13	Softwaregleichung und ihre Faustformeln	287
9.5.14	Earned-Value-Analyse	289
9.5.15	Burn-down-Chart	293
9.5.16	Meilenstein-Trendanalyse	294
9.5.17	Critical-Chain-Projektmanagement	296
9.5.18	Aktueller Staffelholzträger	303
9.6	Kostenmanagement	306
9.6.1	PMBoK-Rahmenwerk	306
9.6.2	Konzentrationsworkshop (Weglassworkshop)	308
9.6.3	Kosten-Nutzen-Priorisierungsmatrix	309
9.6.4	Tauschobjekte sammeln	312
9.7	Qualitätsmanagement	314
9.7.1	PMBoK-Rahmenwerk	314
9.7.2	Externe Projektreviews	314
9.7.3	Autor-Kritiker-Treffen	315
9.7.4	Retrospektivenworkshop	319
9.7.5	Präparationsworkshop	321
9.7.6	Stabilisierungssiteration	323

9.8	Personalmanagement.....	324
9.8.1	PMBok-Rahmenwerk	324
9.8.2	Produktivitätsbetrachtungen	325
9.8.3	Gruppendynamik.....	326
9.8.4	Newbie-Mentor.....	331
9.8.5	Prozessverantwortliche(r)	331
9.9	Kommunikationsmanagement.....	333
9.9.1	PMBok-Rahmenwerk	333
9.9.2	5-mal-Warum-Fragetechnik	334
9.9.3	Aktives Zuhören	335
9.9.4	Freud'sches Eisbergmodell.....	337
9.9.5	Feedback	339
9.9.6	Fishbowl.....	341
9.9.7	Jung'sches Typenmodell	343
9.9.8	Stehung.....	345
9.9.9	Ampelstatus	347
9.9.10	Eisenhower-Prinzip	348
9.9.11	Der Läufer	349
9.9.12	Mit dem Kunden essen / Bier trinken gehen	350
9.9.13	Projektleitungs-Jour fixe, wöchentliches	351
9.9.14	Projektmarketing	352
9.9.15	Projektparty	354
9.9.16	Prosecco-Event.....	355
9.9.17	Projektmitarbeiter-Steckbriefe.....	356
9.9.18	Salamitaktik.....	358
9.9.19	Stakeholder-Analyse	359
9.9.20	Stakeholder-Diagramm	362
9.9.21	Stakeholder-Priorisierung	363
9.9.22	Stakeholder-Profile	365
9.9.23	Moderations- und Sitzungstechniken.....	368
9.9.24	Systemmodell	369
9.10	Risikomanagement.....	371
9.10.1	PMBok-Rahmenwerk	371
9.10.2	Projektstart-Checkliste	372
9.10.3	Aussitzen	374
9.10.4	Erfolgsfaktorworkshop	375
9.10.5	Frühe Eskalationsprobe	376
9.10.6	Projektsponsortreffen.....	378
9.10.7	Projekttagebuch.....	379
9.10.8	Realisierungswettbewerb.....	380
9.10.9	Risikoliste.....	382
9.10.10	Risikoworkshop.....	383
9.10.11	T-Stich-Prototyp (vertikales Prototyping)	387
9.11	Vertrags- und Einkaufsmanagement.....	389
9.11.1	PMBok-Rahmenwerk	389
9.11.2	Projektauftrag.....	392
9.11.3	Allgemeine Projekt-Geschäftsbedingungen (APGs)	394
9.11.4	Agiler Festpreis	400
9.11.5	Anforderungseinheitspreis	402
9.11.6	Aufwandspreis	403
9.11.7	Aufwandspreis mit Obergrenze.....	404

9.11.8	Festpreis	405
9.11.9	Mehrstufiger Festpreis	406
9.11.10	Phasenfestpreis	407
9.11.11	Kulanzrechnung	408
9.11.12	Lebenszyklus-Kostenfalle	409
9.11.13	Lenkungsreis.....	411
10	Anhang.....	415
10.1	Glossar	416
10.2	Literatur	430
10.3	Index	433

1 Einleitung

Um die Welt zu ruinieren, genügt es,
wenn jeder seine Pflicht tut.

Winston Churchill

Worum geht es in diesem Kapitel?

- Sie erhalten einen sehr komprimierten Überblick über das APM-Verfahren.
- Sie erfahren etwas über die Entstehung, Motivation und die Werte agiler Softwareentwicklung.
- Die organisatorischen Besonderheiten verschiedener Projektgrößen werden skizziert.
- Sie erhalten Hinweise zur Einführung und Adaption des APM-Verfahrens.

1.1 Das APM-Verfahren im Überblick

Zu Beginn des Projektes liegt das benötigte Ergebnis in einer Wolke. Es ist unscharf. Alle Beteiligten, Fachabteilung wie Entwickler, wissen nur grob, wie das Ergebnis aussehen soll. Es werden verschiedene Annahmen getroffen, die wahrscheinlich nur in Teilen zutreffend sein werden. Auf Basis dieser Annahmen wird das Projekt geplant, es wird ein Ziel angepeilt und dieses dann verfolgt.

Abnehmende Unschärfe

Anstatt nun die Augen zu verdrehen und darüber zu schimpfen, dass das Ziel in der Regel eher milchtrübe als glasklar ist, wird beim iterativen Vorgehen akzeptiert, dass die Klarheit über das herzustellende Produkt nicht mit einem Mal plötzlich entsteht, sondern schrittweise zustande kommt, und dass das Ziel keine konstante Größe ist, sondern sich mit der Zeit verändern kann.

Deswegen wird die Projektlaufzeit beim iterativen Vorgehen in eine Sequenz von Zeitfenstern eingeteilt, die man Iterationen nennt. Am Ende jeder Iteration wird innegehalten und zurückgeblickt:

- Was haben wir tatsächlich erreicht?
- Was wollten wir ursprünglich erreichen?
- Was lernen wir daraus?

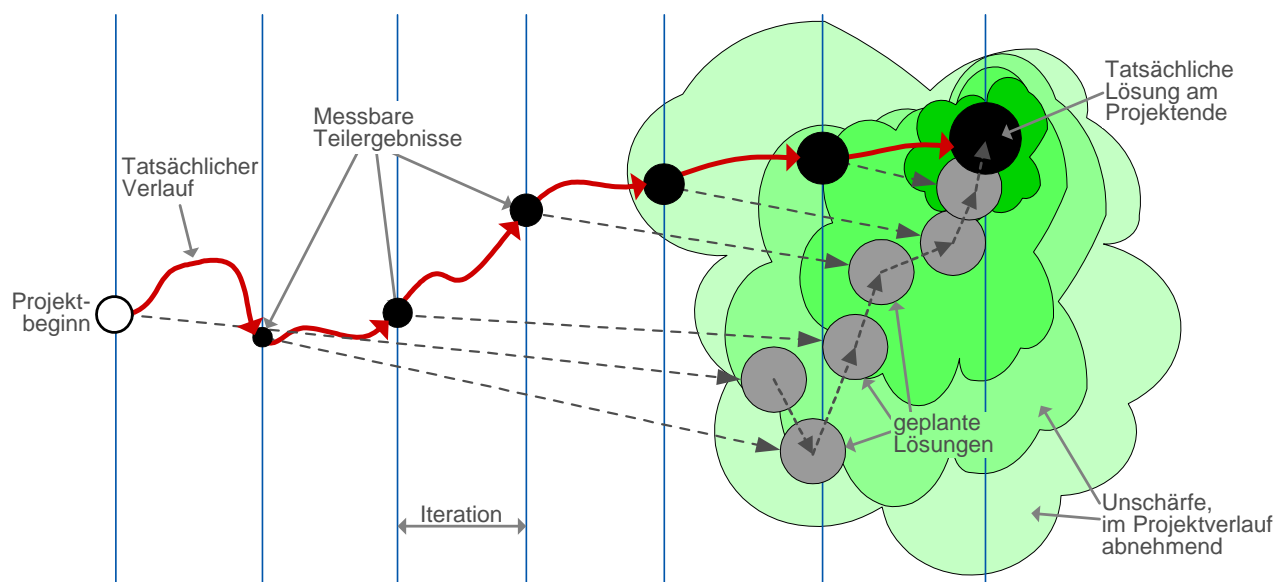


Abb. 1.1-1: Die Iterations-Wolken-Metapher: Schrittweise Zielklärung und -näherung (farbige Abbildung im PPT- und PDF-Format herunterladen unter www.oose.de/apm/download)

Dann wird der Blick wieder nach vorne gerichtet. Durch die zwischenzeitlich gewonnenen Erkenntnisse ist die Wolke etwas kleiner geworden. Das Ziel ist zwar immer noch unscharf, aber etwas weniger. Ausgehend von der in der abgeschlossenen Iteration tatsächlich erreichten Position wird nun ein neuer Plan gemacht und wieder das (nun etwas konkretere) Ziel angepeilt. Der Prozess beginnt von vorne.

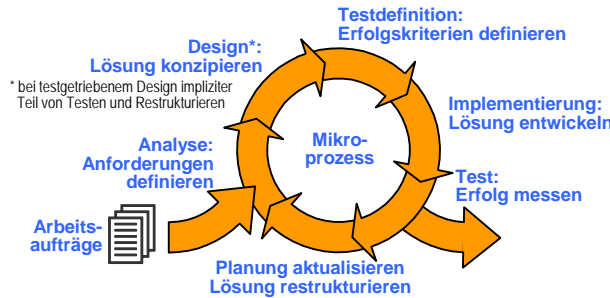


Abb. 1.1-2: Mikroprozessmodell einer Iteration

Mikroprozess
⇒178

Wichtig hierbei ist, dass in jeder Iteration prinzipiell alle elementaren Entwicklungsaktivitäten (Anforderungen definieren, Lösung konzipieren, Erfolgskriterien definieren, Lösung entwickeln, Erfolg messen und schließlich Planung aktualisieren) durchlaufen werden (siehe Abb. 1.1-2). Spätestens am Ende einer jeden Iteration steht also ein objektiv messbares Teilergebnis, ein Inkrement, d.h. eine teilfertige, vorübergehende, aber ausführbare Version der angestrebten Lösung.

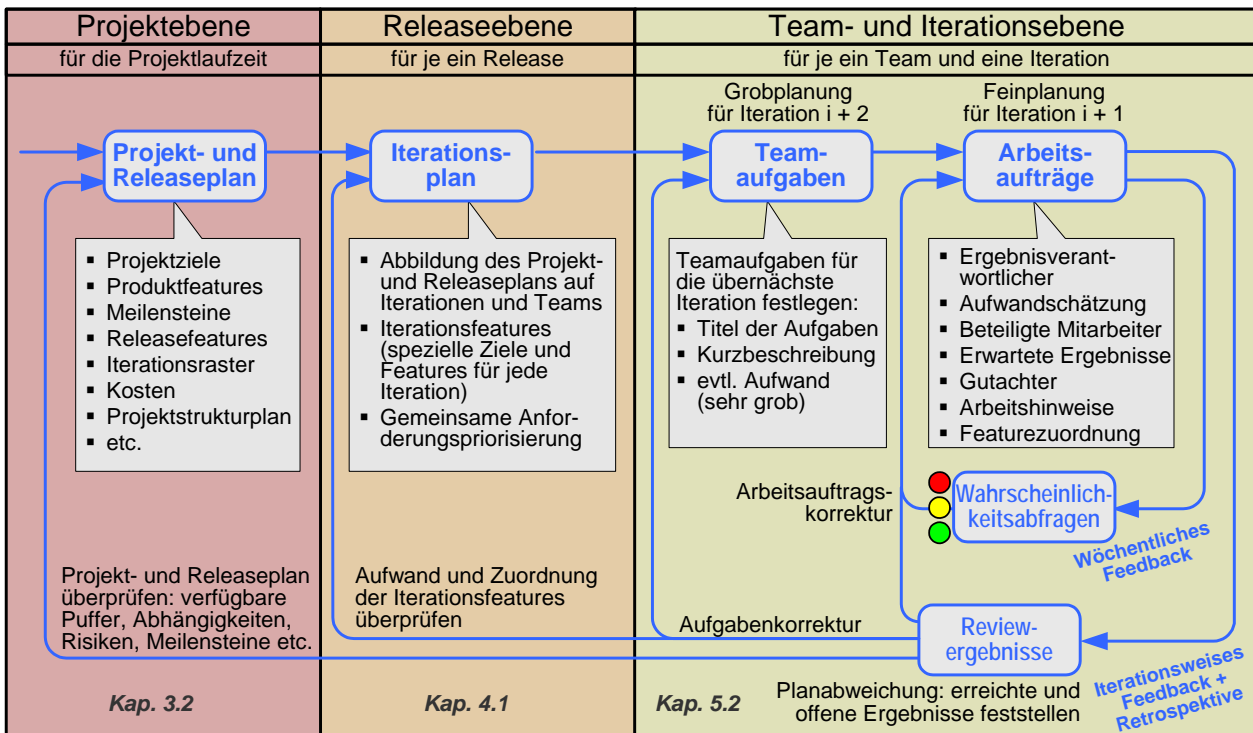


Abb. 1.1-3: Planungsebenen und Feedback-Schleifen (farbige Abbildung im PPT- und PDF-Format herunterladen unter www.oose.de/apm/download)

Wir verstehen Iterationen immer als sogenannte Timeboxen, d.h. einmal gestartet, wird der Endtermin einer Iteration nicht mehr verschoben, gerade auch dann nicht, wenn die Ergebnisse der Iteration hinter dem Plan zurückbleiben.

Iteration = Timebox
⇒190

Abb. 1.1-3 stellt diesen Regelkreis dar und zeigt die verwendeten Planungselemente und -ebenen im Überblick. Grundsätzlich unterscheiden wir dabei die Projektebene, die Releaseebene sowie die Team- und Iterationsebene.

- Die **Projektebene** enthält all jene Ergebnisse und Dokumente, die sich stets auf den gesamten Projektumfang beziehen. Sie werden während der gesamten Projektlaufzeit regelmäßig aktualisiert.

Projektebene ⇒122

Hierzu gehören die Projektziele, die Liste der herzustellenden Produktfeatures (also eine grobe Leistungsbeschreibung des oder der herzustellenden Produkte), Aussagen zu Kosten und geplanten Releases mit deren Features, Terminen und Meilensteinen sowie ein Iterationsraster (Anzahl und Dauer der Iterationen). Das wichtigste Ergebnis der Projektebene ist der **Projekt- und Releaseplan**.

- Die **Releaseebene** verfeinert die Ergebnisse der Projektebene dahingehend, dass für jedes Release die herzustellenden Releasefeatures auf die Iterationen und Teams aufgeteilt werden. Aus Releasefeatures werden **Iterationsfeatures** abgeleitet, um festzulegen, welche Ziele und Anforderungen in welcher Iteration von welchem Team zu bearbeiten sind, sodass das Release entsteht. Dabei werden die Anforderungen priorisiert, um deren Umsetzungsreihenfolge grob festzulegen. Das Ergebnis nennen wir **Iterationsplan**. Iterationsfeatures sind Zielvorgaben für die Teams.

Releaseebene ⇒135

- Die **Team- und Iterationsebene** verfeinert wiederum die Ergebnisse der Releaseebene. Für jedes Team und jede Iteration existieren grobe Zielbeschreibungen in Form von Iterationsfeatures. Nun gilt es, hieraus Aufgaben für einzelne Mitarbeiter und Teams abzuleiten. Dazu später mehr im Zusammenhang mit Abb. 1.1-6.

Team- und Iterations-
ebene⇒161

Anhand der Abb. 1.1-3 wird auch deutlich, wie Erkenntnisgewinn in die Planung zurückgekoppelt wird (Feedback).

Der prinzipielle Weg der Verfeinerung von Features zum Arbeitsauftrag ist in Abb. 1.1-4 dargestellt. Die Aufteilung von Releasefeatures in Iterationsfeatures, die dann vollständig von einem Team und in genau einer Iteration entwickelt werden, ist dabei ein Idealfall. Die Abbildung zeigt, dass Releasefeatures in Iterationsfeatures für verschiedene Teams aufgeteilt werden können und dass gegebenenfalls bereits in einer vorigen Iteration Arbeitsaufträge als Zulieferleistung für ein Iterationsfeatures existieren können. Ebenso ist zwar stets ein Team verantwortlich

für ein Iterationsfeature, dennoch können Zulieferleistungen aus anderen Teams oder gar anderen Teilprojekten erfolgen.

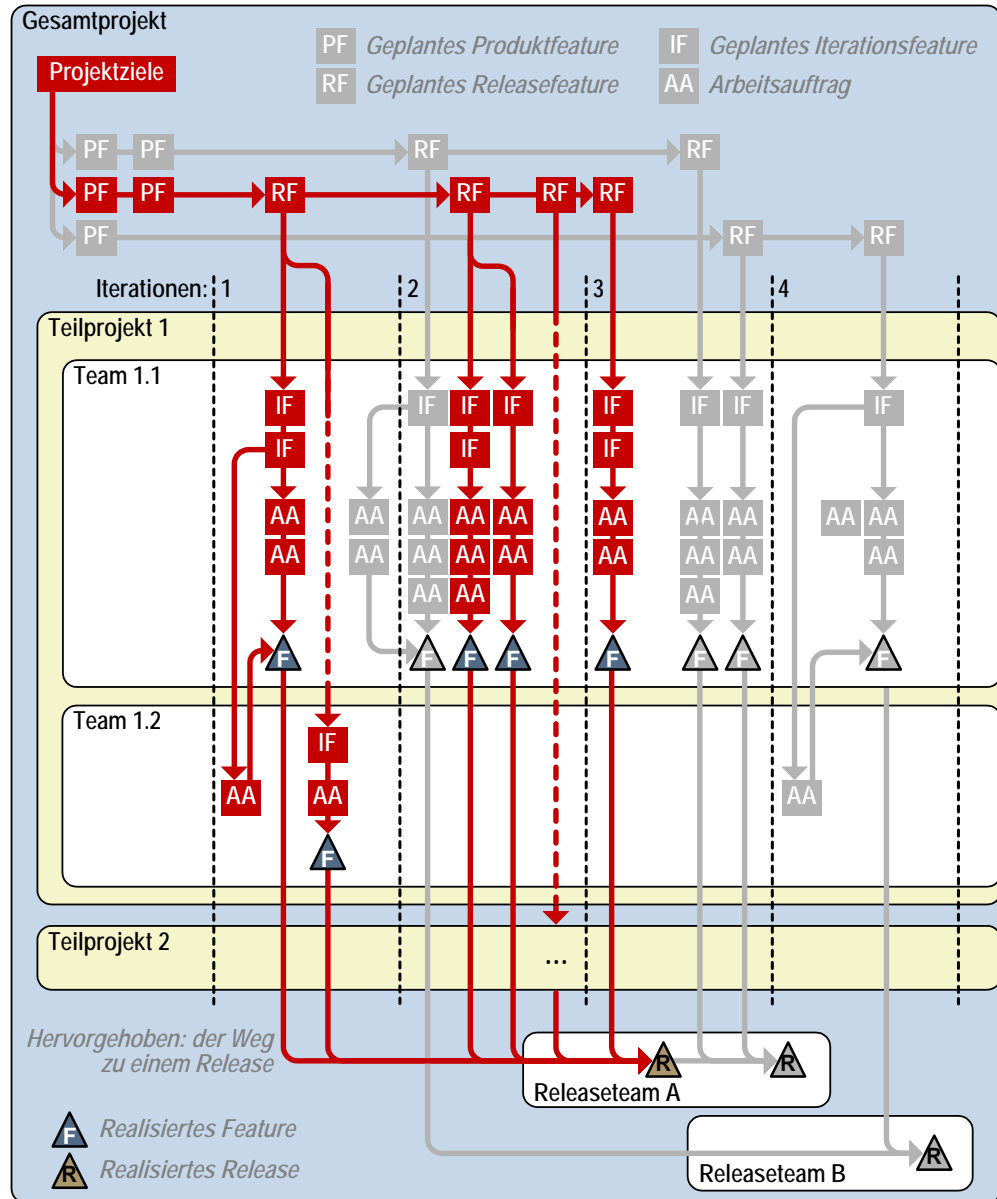


Abb. 1.1-4: Der Weg vom Projektziel über Produktfeatures, Releasefeatures und Iterationsfeatures zum Arbeitsauftrag

Produktfeatures
⇒ 138, 143, 154

Vorbereitungsphase ⇒ 63

Auf einer grobgranularen Ebene wird also das Gesamtprojekt in Form eines Projekt- und Releaseplans (vgl. Abb. 1.1-5) geplant, der im weiteren Projektverlauf kontinuierlich weiterentwickelt wird. Es werden Gesamtkosten, Produktfeatures, Endtermin, Releasetermine und Releasefeatures beschrieben.

Der Zeitraum, in dem die *erste Version* des Projekt- und Releaseplans entsteht, nennen wir **Vorbereitungsphase**, weil wir diese Informationen gewöhnlich benötigen, bevor wir ein Angebot abgeben können oder einen Auftrag bekommen.

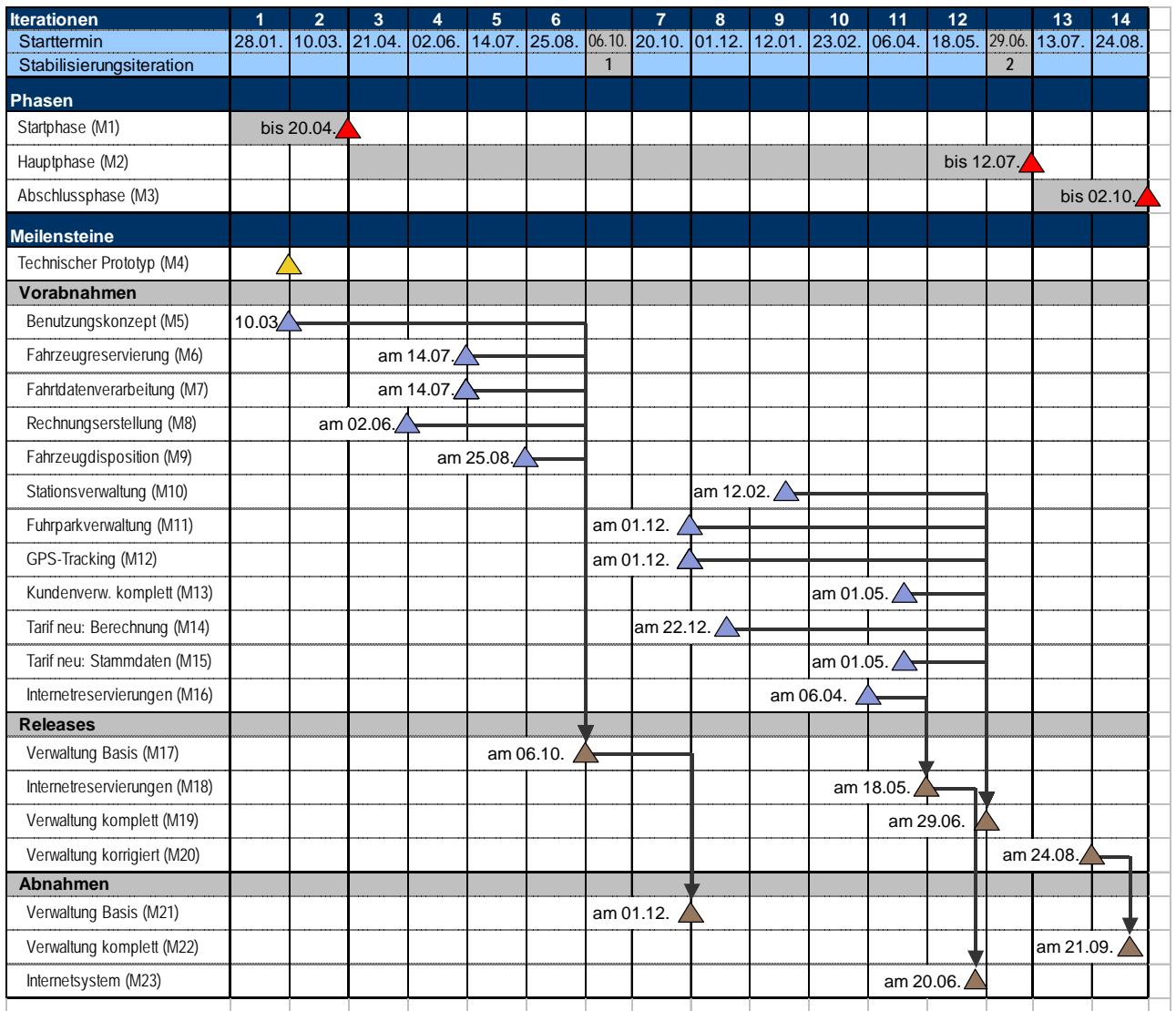


Abb. 1.1-5: Projekt- und Releaseplan mit Iterationsraster, Phasen und Meilensteinen für Vorabnahmen, Releases und Abnahmen

Die eigentliche Projektlaufzeit wiederum teilen wir nun in ein Raster etwa gleich langer **Iterationen** auf (im Sinne von **Timeboxen**). Dazu benötigen wir den Endtermin und die Entscheidung darüber, wie lange eine Iteration sein soll. Über dieses Iterationsraster werden zur Orientierung noch folgende drei grundsätzliche zeitliche Abschnitte gelegt (vgl. Abb. 1.1-5):

- Die **Startphase**, in der noch viele grundsätzliche Fragen und Entscheidungen ungeklärt oder instabil sind (besteht aus einigen wenigen Iterationen). Die Planung wird spätestens jetzt und mindestens für das erste Release bis auf Releaseebene herunter geplant.
- Die **Hauptphase**, in der auf Basis von als stabil zu betrachtenden Entscheidungen und Strukturen die eigentliche Entwicklung erfolgt (erstreckt sich über den Hauptteil der Iterationen) und bereits ver-

Startphase ⇒ 113

Hauptphase ⇒ 118

schiedene Releases entstehen. Das heißt, schrittweise werden bereits Systemteile eingeführt und gegebenenfalls auch vom Kunden abgenommen.

Abschlussphase ⇒119

- Die **Abschlussphase**, in der das Produkt (soweit noch nicht in der Hauptphase geschehen) eingeführt und dem Auftraggeber übergeben wird (erstreckt sich über eine oder wenige Iterationen).

Was in Abb. 1.1-3 als Team- und Iterationsebene aus planungstechnischer Sicht dargestellt wird, spielt sich größtenteils innerhalb einer Iteration ab, deren Ablauf wir nun näher spezifizieren. Abb. 1.1-6 gibt einen grafischen Überblick und zeigt den grundsätzlichen Aufbau einer Iteration. Dabei wird die Iteration zunächst in einen (längeren) Fortschrittsabschnitt und einen (kürzeren) Orientierungsabschnitt unterteilt.

Iterationsfeatures
⇒146, 151, 155

- Ausgehend vom **Iterationsplan**, der Iterationsziele und **Iterationsfeatures** enthält, wird nun die Planung iterationsweise verfeinert:
 - ◆ Um diese Detailplanung schon etwas vorzustrukturieren, findet für die jeweils übernächste **Iteration (i+2)**, also vorher, bereits eine Grobplanung statt. Hier werden auf Basis von Iterationsfeatures team- und iterationspezifisch die Arbeitsaufträge in Form abstrakter **Teamaufgaben** identifiziert, d.h. lediglich mit Titel, Kurzbeschreibung und gegebenenfalls mit einer ganz groben Aufwandschätzung versehen.
 - ◆ Für die jeweils nächste **Iteration (i+1)** legt jedes Team feingranular die **Arbeitsaufträge** fest, die innerhalb der Iteration erledigt werden sollen. Ein Arbeitsauftrag beschreibt, wer für das Ergebnis verantwortlich ist, welche weiteren Personen daran beteiligt sind, wie das Ergebnis aussehen und abschließend beurteilt werden soll, wer das Ergebnis abschließend begutachten soll und wie viel Aufwand insgesamt für den Arbeitsauftrag geschätzt wird.

Zwei-Bugwellen-Planung
⇒162, 166

Diese teamspezifische Grobplanung (Iteration i+2) und Feinplanung (Iteration i+1) nennen wir die **Zwei-Bugwellen-Planung**, weil sie wie Bugwellen vor der aktuellen Iteration entstehen und iterationsweise aktualisiert werden.

Tägliches Teamsteuerungstreffen ⇒182, Stehung
⇒345

- Während der Iteration koordiniert und synchronisiert jedes Team selbstverantwortlich die eigene Arbeit durch kurze **tägliche Teamsteuerungstreffen** (Daily-Scrum-Meetings, Stehungen).

Tägliches Build ⇒181

- Möglichst kontinuierlich oder zumindest täglich ist eine unvollständige, aber prinzipiell lauffähige Version der Software zu bauen und zu testen (**tägliche (Smoke-)Builds**). Je nach vertretbarem Aufwand und Machbarkeit sind alle Neuerungen und Änderungen zumindest teamweit, möglichst aber auch projektweit zu integrieren. So entstehen in sehr kurzen Zyklen **Alpha-Versionen** der Software (Always Alpha).

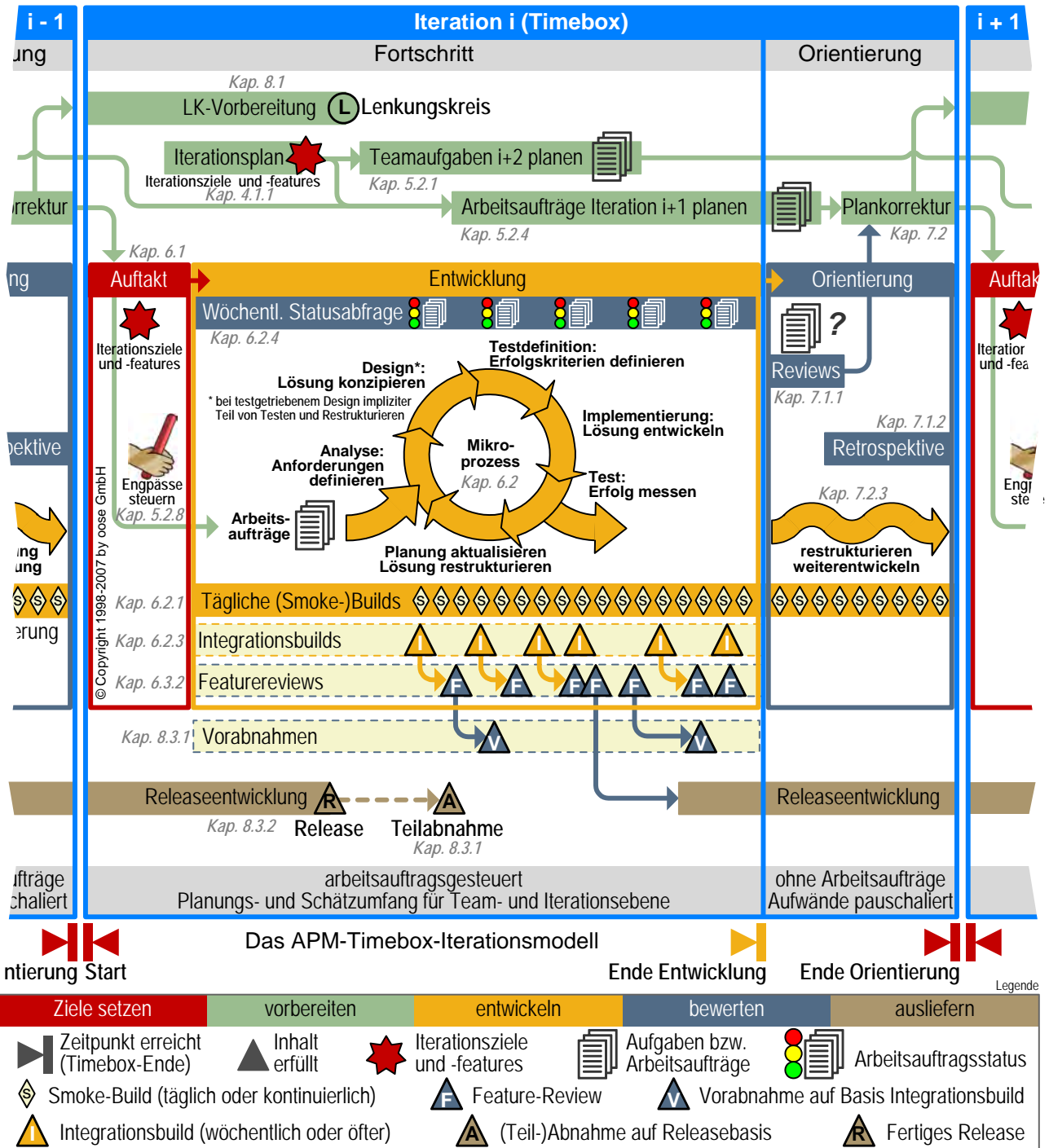


Abb. 1.1-6: Prinzipieller Aufbau einer Iteration (farbige Abbildung im PPT- und PDF-Format herunterladen unter www.oose.de/apm/download)

- Einmal wöchentlich werden systematisch teamübergreifend Problemsituationen gesucht, um innerhalb einer Iteration den Plan realistisch zu halten und auch übergeordnet intervenieren zu können (wöchentliche arbeitsauftragspezifische Statusabfragen, **Ampelstatus**, Wahrscheinlichkeitsabfragen, Läufer).

Ampelstatus ⇨ 185, 347
Läufer ⇨ 349

Arbeitsauftragsreviews
⇒198

Planungskorrektur ⇒202

- Am Ende einer jeden Iteration führen alle Teams zeitgleich eine Bestandsaufnahme (Soll-Ist-Abgleich) für alle bearbeiteten Arbeitsaufträge durch (**Arbeitsauftragsreviews**). Mit diesen Erkenntnissen wird die bereits vorliegende Feinplanung (teamspezifische Arbeitsaufträge) für die unmittelbar folgende Iteration abgeglichen und angepasst (**Planungskorrektur**).

Retrospektive ⇒201

- Ebenfalls am Ende der Iteration führen alle Teams und sonstigen Organisationseinheiten des Gesamtprojektes jeweils eigene **Retrospektiven** durch, mit denen geklärt werden soll, was gut lief und wie die zukünftige Arbeit verbessert werden kann.

Releaseentwicklung ⇒221

- Um ein **Release** herzustellen, wird die erste potenzielle Alpha-Version, die alle geforderten Releasefeatures erfolgreich und stabil umgesetzt hat, aus dem Entwicklungsprozess herausgenommen und zu einem Release weiterentwickelt. Gegebenenfalls können Releases auch von separaten eigenständigen **Releaseteams** entwickelt werden. Bei Bedarf können auch mehrere Releases gleichzeitig bzw. zeitlich überlappend entstehen (**Parallelreleases**).

Teilabnahmen ⇒220

- Releases können zu vertraglich relevanten **Teilabnahmen** führen.

Vorabnahmen ⇒220

- Unabhängig von Releases können auf der Basis von Alpha-Versionen (vorbehaltlich der Reproduzierbarkeit in einer Releaseversion) vertraglich relevante **Vorabnahmen** stattfinden.

So weit in aller Kürze der Überblick. Wie dieses Verfahren nun im Detail funktioniert, welche Varianten und andere wichtige Aspekte es gibt, das erfahren Sie in den übrigen Kapiteln dieses Buches.

Sie werden dort einen relativ festen und klaren Rahmen, Anleitungen und Hinweise zu speziellen Vorgehensweisen finden sowie viele einzelne Handlungsanweisungen, die miteinander verzahnt sind und aufeinander aufbauen: Tue dies und tue das und dann das. Manchmal erscheint Ihnen dies möglicherweise sogar bürokratisch.

Bitte beachten Sie jedoch die in Kapitel *1.3 Werte* (⇒20) beschriebenen Werte und die in den Kapiteln *1.4 Projektorganisation und Projektgrößen* (⇒30) und *1.5 Einführung und Adaption des Verfahrens* (⇒41) genannten Grundlagen und Rahmenbedingungen. Ziel dieser Rahmenbedingungen und Vorgehensweise ist es, ein Umfeld und Rahmenbedingungen zur guten Entfaltung agiler und eigenverantwortlicher Projektarbeit zu schaffen. Wir definieren in diesem Kapitel bewährte Spielregeln, grenzen Aufgaben und Verantwortlichkeiten ab, damit sich innerhalb dieses Rahmens ohne unnötige Konflikte und mit klaren Freiräumen und Verantwortlichkeiten effizient und effektiv arbeiten lässt.

Die Ausgangssituationen, Projektkulturen, Werte, vorhandenen Erfahrungsschätze etc. sind in großen Projekten sehr unterschiedlich. Was in einem Umfeld selbstverständlich ist, kann in einem anderen Umfeld revolutionär oder fragwürdig sein. Und wenn wir hier eine Vorgehensweise beschreiben, die sich vielfach bewährt hat, kann sie genau in Ihrem Projekt möglicherweise völlig unpassend sein. Also verstehen Sie den in diesem Buch beschriebenen Prozess als Anregung und Ausgangsbasis – aber es bleibt Ihre eigene Verantwortung zu entscheiden, ob oder wie dieser Prozess in Ihrer Organisation wirklich gut funktionieren kann.

Wir betrachten den hier beschriebenen Prozess als vielfach bewährt und praxistauglich, und doch unterscheiden sich alle uns bekannten Praxisbeispiele in ihrer ganz konkreten Ausgestaltung. Deswegen geben wir immer wieder Hinweise auf Alternativen und typische Hindernisse oder auf Abhängigkeiten von bestimmten Faktoren.

Die wichtigsten Faktoren hierbei sind:

■ Projektgröße

Um das Verständnis zu erleichtern, fokussieren wir in der Darstellung unseres Verfahrens vorwiegend auf ein Großprojekt mit einer Iterationsdauer von sechs Wochen, einer Gesamtdauer von 20 – 24 Monaten und ca. vier Releases bis zum Endprodukt. Unser Fallbeispiel basiert auf ca. 25 Projektmitgliedern. Darauf aufbauend beschreiben wir dann bedarfsweise, welche Änderungen oder Einschränkungen zu beachten sind, wenn andere Rahmenbedingungen vorliegen.

Vgl. Fallbeispiel ⇨65

■ Auftraggeber-Auftragnehmer-Situation

Eine unternehmensinterne Entwicklung bringt andere Herausforderungen mit sich als beispielsweise ein harter externer Auftraggeber. Je nach Situation sind andere Vorgehensweisen sinnvoll oder wichtig. In unserem Fallbeispiel beziehen wir uns auf einen externen Auftraggeber.

■ Soziale Beziehungen

Haben Auftragnehmer und Auftraggeber aus vergangenen Vorhaben ein belastbares, krisenfestes vertrauensvolles Verhältnis? Kennen sich die Beteiligten schon persönlich? Ist das Projektteam eingespielt oder ist es ein bunter Haufen Unbekannter? Werden Konflikte unter den Teppich gekehrt statt konstruktiv ausgetragen?

■ Fachliche Sicherheit

Wie gut kennen die Projektmitarbeiter die Fachlichkeit, die Ziele und das Umfeld des Kunden?

■ Werte und Nachhaltigkeit

Gerade in Großprojekten findet sich keine heile Welt, und die herrschenden Werte widersprechen oftmals jeglichen sozialromantischen Idealen. Egal, ob man solche Ideale vertritt oder im Gegenteil sogar hemmungslos mitspielt, die Werte und Spielregeln sollten von der Projektleitung zumindest erkannt und verstanden und möglichst kompetent abgefangen werden. Aus diesem Grund erwähnen wir in der Werkzeugsammlung in Kapitel 9 beispielsweise auch einige zweiseitige Managementstrategien.

Für die Einführung des APM-Verfahrens werden keine speziellen Werte vorausgesetzt, es forciert und fördert jedoch bestimmte Werte. Je nach Ausgangssituation gestaltet sich die Einführung des Verfahrens etwas anders.

1.2 Agile Softwareentwicklung

1.2.1 Historische Missverständnisse

Winston Royce

Iterative Projektmanagementverfahren sind so alt wie die Informatik. Bereits 1970 publizierte Winston Royce ein erstes Modell [Royce-1970]. Diese Publikation wurde wiederum verwendet bei der Entwicklung des US-Militär-Standards DoD-STD-2167.

David Maibor

Der 2167-Mitautor David Maibor, der sich auf die Arbeit von Winston Royce bezog, war jedoch mit iterativ-inkrementeller Entwicklung und evolutionären Anforderungen nicht vertraut. Stattdessen berief sich Maibor auf eine spezielle und stark vereinfachte Variante des Modells von Royce, die mit 1 – 2 Iterationen auskam. So legte Maibor mit dieser Vereinfachung die Grundlagen des Wasserfallmodells.

STD-2167 wurde wiederum Grundlage für andere Vorgehensmodelle, wie etwa CMMI (Capability Maturity Model Integration) oder das deutsche V-Modell.

Wasserfallmodell ist ein historischer Irrtum

Walker Royce, der Sohn von Winston Royce, berichtete später, sein Vater wäre immer ein Vertreter von iterativen, inkrementellen und evolutionären Entwicklungsmodellen gewesen. Sein Arbeitspapier hätte das Wasserfallmodell lediglich als die einfachste Möglichkeit, die aber nur für die unkompliziertesten Projekte funktioniert, beschrieben. Und auch Maibor hat später geäußert, dass er, wäre er damit damals vertrauter gewesen, das iterative Vorgehen im STD-2167 viel deutlicher emp-

fohlen hätte [Larman-2004b]. So gesehen handelt es sich beim Wasserfallmodell um einen historischen Irrtum, der bekanntermaßen problematische Effekte erzeugt, insbesondere wenn man das Wasserfallmodell unreflektiert auf große Projekte anwendet.

Interessant ist dabei, dass Winston Royce den Begriff „Wasserfall“ überhaupt nicht benutzt. Wer seine Veröffentlichung jedoch liest, dem wird unmittelbar klar, warum es sich um ein „Wasserfallmodell“ handelt. Das Papier von Winston Royce ist übersät mit Abbildungen, in denen kaskadenartige (=wasserfallartige) Phasen visualisiert werden. Mehr zu den Grundlagen des Wasserfallmodells findet sich in [Himmelreich-2006].

Warum der Erfinder des Wasserfallmodells missverstanden wurde

In den 1980er- bis 1990er-Jahren verbreitete sich das Wasserfallmodell innerhalb der Informatik sehr rasant und wurde Praxisstandard. Das iterative Modell wurde weiterentwickelt, hatte es jedoch schwer, sich durchzusetzen, obwohl es beachtliche Erfolge damit gab. So entwickelte beispielsweise die IBM von 1977 – 1980 die Software für das NASA Space-Shuttle in 17 Iterationen über 31 Monate mit durchschnittlich achtwöchigen Iterationen [Madden-1984].

Space-Shuttle-Software iterativ entwickelt

Etwas später, 1985, publizierte Barry Boehm das sogenannte Spiralmodell [Boehm-1985]. Dieses Modell, das ebenfalls einen iterativ-inkrementellen Ansatz verfolgt, wurde theoretisch viel beachtet und anerkannt, in der Praxis jedoch selten angewendet. Der irrtümliche Standard Wasserfallmodell galt als bodenständiger.

Spiralmodell

Ende der 1990er-Jahre erblühten dann verschiedene neue Varianten des iterativen Vorgehens. Am bekanntesten wurde Extreme Programming (XP, [Beck-2003]). Im Jahre 2005 schließlich wurde die erste Version des V-Modell XT veröffentlicht, das offizielle Vorgehensmodell in der Bundesrepublik Deutschland, in dem erstmalig auch iterativ-inkrementelle und agile Projektdurchführungsstrategien enthalten sind. Spätestens seit diesem Zeitpunkt sind agile Managementtechniken auch im Rahmen öffentlicher Aufträge anwendbar.

Das iterative Vorgehen wurde Anfang der 2000er-Jahre immer populärer in der Praxis. Immer mehr Studien belegten schließlich auch die Vorteile und Erfolge dieses Ansatzes, beispielsweise der regelmäßige sogenannte Chaos-Report der Standish Group [Standish-Chaos]. Dazu mehr in Kapitel 1.5.3 *Faktoren für erfolgreiche Projekte* (⇨47).

Chaos-Report

1.2.2 Entstehung und Motivation

Hinter den Schlagworten *agile Softwareentwicklung* oder *agiles Projektmanagement* verbirgt sich eine Ende der 1990er-Jahre entstandene Gegenbewegung zu den überreglementierten und starren Ansätzen der 1980er- und 1990er-Jahre.

Standardisierung

Seit Anbeginn der Informatik existieren Bestrebungen, Softwareentwicklung erfolgreicher zu machen. Ein Mittel hierfür ist die Standardisierung von Prozessen beispielsweise mithilfe von Vorgehensmodellen. Die Grundidee dabei ist, bewährte Techniken und Vorgehensweisen als Mindeststandards konkret festzuschreiben und zu standardisieren. Bekannte Fehler und Problemsituationen sollen damit vermieden werden, was vor allem für Unerfahrene nützlich ist.

Schattenseiten

Grundsätzlich funktioniert dieser Verbesserungsansatz, aber er hat auch Schattenseiten:

- Sehr erfahrene und erfolgreiche Projektmanager werden gelegentlich behindert. Es erfolgt im Zweifelsfall eine Standardisierung auf Mittelmaß.
- Projekte mit üblichen, durchschnittlichen und von den Vorgehensmodellen vorgesehenen Problemsituationen erhalten eine gute Unterstützung. Für alle anderen Projekte kann der Standard unpassend, möglicherweise sogar kontraproduktiv sein.

Projektbürokratie

- Vorgehensstandards können zum bürokratischen Selbstzweck verkommen, wenn die dafür Verantwortlichen den Praxisbezug verloren haben.
- Standards werden oftmals zu langsam weiterentwickelt und zielen somit auf Problemsituationen, die in der Vergangenheit eine Relevanz hatten, unterstützen aber die aktuellen Fragestellungen unzureichend.
- Die beteiligten Personen übernehmen weniger eigene Verantwortung für ihre Entscheidungen und ihr Handeln, da sie schließlich nur Vorschriften befolgen.

Die sogenannte agile Entwicklung ist eine Gegenbewegung hierzu, die die Vorteile und Errungenschaften einerseits erhalten will, die aber andererseits die erkannten Begrenzungen auflösen und darüber hinausgehen möchte.

Manchmal schließen sich solchen Bewegungen die falschen Personen an. Die agile Entwicklung wurde initiiert von sehr erfahrenen Personen, die an einer Weiterentwicklung und Innovation interessiert sind. Sie sollten nicht verwechselt werden mit den Protagonisten, die aus Be-

quemlichkeit die mühevollen Weiterentwicklung und das Lernen scheuen und ihre Unerfahrenheit oder Stagnation damit rechtfertigen, Agilität als Beliebigkeit zu interpretieren.

Der Wortlaut in den folgenden Abschnitten ist durchaus ernst zu nehmen. Wenn es heißt "Höchste Priorität hat die Zufriedenstellung des Auftraggebers durch frühe und kontinuierliche Lieferung brauchbarer Software", dann ist "brauchbar" nicht zu interpretieren als perfekt oder optimal, sondern einfach nur als brauchbar.

Und wenn es heißt, "Funktionierende Software ist wichtiger als umfangreiche Dokumentation", dann bedeutet dies nicht, dass Dokumentation unwichtig oder zu vermeiden ist, sondern dass die beste und umfangreichste Dokumentation keinen Wert hat, wenn die Software nicht funktioniert.

Beim Thema Standardisierung ist außerdem zu unterscheiden, ob Standards abstrakt von außen vorgegeben werden und deswegen möglicherweise für die projektspezifische Situation nicht passen oder ob es Vereinbarungen und Standards sind, die sich das Projekt selbst gegeben hat. Im letzteren Fall ist die Disziplin der beteiligten Projektmitarbeitenden zu erwarten.

1.2.3 Das agile Manifest

Das agile Manifest (siehe <http://www.agilemanifesto.org/>) wurde im Jahr 2001 von Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland und Dave Thomas im Rahmen eines Treffens in Snowbird, Utah, ins Leben gerufen und innerhalb weniger Wochen von Hunderten anderer bekannter Persönlichkeiten der Branche unterzeichnet.

Der Wortlaut des Manifestes:

„Wir entdecken bessere Wege zur Entwicklung von Software, indem wir Software entwickeln und anderen bei der Entwicklung helfen. Durch diese Tätigkeiten haben wir gelernt:

- Individuen und Interaktion sind wichtiger als Prozesse und Werkzeuge
- Funktionierende Software ist wichtiger als umfangreiche Dokumentation
- Kooperation mit Projektbetroffenen ist wichtiger als Vertragsverhandlungen
- Reaktion auf Änderungen ist wichtiger als Festhalten an einem starren Plan

Natürlich sind auch die Dinge rechts wichtig, aber im Zweifelsfall schätzen wir die linken höher ein.“

Hier ein paar Erläuterungen zum agilen Manifest:

■ **Individuen und Interaktion sind wichtiger als Prozesse und Werkzeuge**

Die Menschen stehen im Mittelpunkt agiler Methoden und gelten als wichtigster Erfolgsfaktor in Projekten. Auf die Motivation und gute Zusammenarbeit im Team wird Wert gelegt. Die Eigenverantwortung der Entwickler und der übrigen Projektbeteiligten wird betont und aktiv gefördert. Das macht die Leichtgewichtigkeit überhaupt möglich. Auf starr reglementierte Prozesse, welche die Menschen aus der Verantwortung nehmen, kann verzichtet werden. Es genügt, nur das Nötigste zu planen und vorzuschreiben.

■ **Funktionierende Software ist wichtiger als umfangreiche Dokumentation**

Laufende Software ist wichtiger als Dokumentation. Nur die wirklich notwendige Dokumentation wird erstellt. Eine gute Kommunikation zwischen den Projektbeteiligten reduziert die Notwendigkeit für umfangreiche Dokumentation. Funktionierende Software wird in kurzen Abständen von ca. 1 bis 3 Monaten an den Kunden ausgeliefert oder diesem zumindest zur Evaluierung bereitgestellt. Ein erstes Release sollte kurz nach Projektstart ausgeliefert werden. Das Projekt erhält somit direktes Feedback, Änderungen sind schnell möglich und reduzieren somit den Aufwand an Anforderungsdokumentation. Zusätzlich übt das Team den Auslieferungsprozess, dieser wird Normalität.

Sofern als Projektergebnis auch eine Dokumentation beispielsweise für die Wartung und Weiterentwicklung gewünscht ist (Dauerdokumentation), wird diese selbstverständlich auch in agilen Prozessen erstellt.

■ **Kooperation mit Projektbetroffenen ist wichtiger als Vertragsverhandlungen**

Die Zusammenarbeit mit dem Kunden steht im Mittelpunkt agiler Ansätze. Eine enge Zusammenarbeit mit dem Kunden wird angestrebt, auch um den Kunden stärker in die Verantwortung zu nehmen. Das Vertrauensverhältnis und die enge Zusammenarbeit sind wichtig. Der Kunde soll maßgeblich mitentscheiden, welche Features in welchen Releases enthalten sein sollen. Durch die frühzeitige und kontinuierliche Lieferung bereits brauchbarer Software wird das Risiko, die falsche Software zu entwickeln, erheblich reduziert. Dadurch sinkt auch die Bedeutung detaillierter Verträge. Nicht gemeint ist, vollständig auf Verträge zu verzichten, sondern den Regelungsbedarf durch eine auf Vertrauen ausgelegte Zusammenarbeit zu minimieren.

■ **Reaktion auf Änderungen ist wichtiger als Festhalten an einem starren Plan**

Während der Projektlaufzeit ergeben sich typischerweise Änderungen und neue Anforderungen. Auf diese soll flexibel reagiert werden. Agile Methoden versuchen, flexibel mit Änderungswünschen umzugehen. Die Planung und Anforderungsdefinition erfolgen evolutionär, das heißt, sie werden schrittweise verfeinert. Trotzdem existiert eine Gesamtplanung, allerdings nur so detailliert, wie es für das Sicherheitsbedürfnis der Beteiligten notwendig ist.

1.2.4 Prinzipien agiler Softwareentwicklung

Folgende Prinzipien ergeben sich direkt aus dem Manifest bzw. wurden daraus abgeleitet:

- Höchste Priorität hat die Zufriedenstellung des Kunden durch frühe und kontinuierliche Lieferung brauchbarer Software.
- Anforderungsänderungen sind auch in fortgeschrittenen Entwicklungsstadien möglich.
- Die Software wird inkrementell und in kurzen Iterationen erstellt.
- Fachexperten und Entwickler arbeiten möglichst direkt und täglich zusammen.
- Die effizienteste und effektivste Art, Informationen zu verbreiten, ist die direkte Kommunikation von Angesicht zu Angesicht.
- Funktionierende Software ist die primäre und wichtigste Kenngröße für den Projektfortschritt.
- Konzentration auf das Wesentliche heißt explizit und regelmäßig zu entscheiden, was wegzulassen ist.
- Das Entwicklungsteam reflektiert in regelmäßigen Abständen darüber, wie es die gemeinsame Arbeit verbessern kann.

Wir möchten hier nicht detailliert die Grundprinzipien wiederholen, die in anderen guten Büchern zur agilen Softwareentwicklung bereits umfassend beschrieben sind, und verweisen für weitere Details auf das Buch „Agile Software-Entwicklung“ von Alistair Cockburn [Cockburn-2003].

„Agile Software-Entwicklung“ von Alistair Cockburn

1.2.5 Bezug zu anderen agilen Verfahren

Scrum, XP, Crystal

Es gibt derzeit kein agiles Standardverfahren, und wahrscheinlich wäre dies auch nicht zielführend. Bekannte Verfahren sind Extreme Programming (XP) [Beck-2003, Wolf-2005], Scrum [Schwaber-2004, Beedle-2002, Schwaber-2007], die Crystal-Methodiken [Cockburn-2003] und der Eclipse-Way [Gamma-2007]. Diese Verfahren sind (mit Ausnahme des Eclipse-Way) ebenso wie unser APM-Verfahren Mitte/Ende der 1990er-Jahre erstmalig publiziert worden. In der Praxis des deutschen Sprachraums gehören Scrum und XP zu den beliebtesten Verfahren.

Viele Elemente, Techniken und Werte dieser Verfahren sind ähnlich. Während zu Beginn der agilen Bewegung die Unterschiede beispielsweise bezüglich typischer Anwendungsgebiete, adressierter Projektgrößen oder Beschränkungen auf spezielle Teildisziplinen oder Projektrollen noch deutlich waren, haben sich in den letzten 5 – 7 Jahren alle Verfahren weiterentwickelt und sind umfassender, allgemeiner sowie praktisch und theoretisch fundierter geworden.

Das APM-Verfahren hat die größten Ähnlichkeiten wahrscheinlich mit Scrum, vor allem bei den Techniken auf der Mikroebene. Scrum macht in diesem Bereich klare Vorgaben beispielsweise bezüglich der Rollen, Dauer von Sprints und spezieller Meetings.

Es existieren viele Ähnlichkeiten

APM beinhaltet Elemente aus Scrum, die aufgrund der eigenständigen Entwicklung teilweise jedoch unter anderem Namen oder in anderer Ausprägung existieren. Scrum beinhaltet eine eigene Terminologie (bspw. *Product Backlog*, *Sprints*, *Sprint Backlog*), während wir im APM-Verfahren uns eher an traditioneller Terminologie orientieren (bspw. *Arbeitsauftrag*, *Featureliste*, *Releaseplan*).

Das APM-Verfahren lehnt sich also bezüglich Methodik, Konzepten und Terminologie so weit wie möglich an klassischen Projektmanagementverfahren an. Ausgangspunkt ist daher nicht die Unterstützung von Entwicklerteams durch Mikromanagementtechniken, sondern ein umfassendes Projektmanagement.

APM ist die agile Erweiterung bewährter Verfahren

APM verstehen wir als eine Erweiterung und Ergänzung traditioneller Projektmanagementansätze und bauen auf diesen auf.¹ Der international bekannteste Ansatz stammt vom Project Management Institute (PMI). Das PMI bietet eine PMP (Project Management Professional) genannte Zertifizierung an, die an eine relativ hohe Eintrittsschwelle

PMI/PMP

¹ Die 5-tägige APM-Ausbildung von oose ist vom PMI in der Weise anerkannt, dass zertifizierte PMPs durch die Teilnahme am APM-Training Punkte zum Erhalt ihres PMP-Status erwerben können.

geknüpft ist. Neben dem PMI hat auch die GPM (Gesellschaft für Projektmanagement) eine Relevanz und hohe Wertigkeit.

Agilität heißt Beweglichkeit. Beweglichkeit setzt Freiräume und Freiheiten voraus, das heißt Abweichung von Standards oder vorgegebenen vielleicht starren, aber möglicherweise auch bewährten Regelwerken. Wo immer Freiheit entsteht, gibt es Verantwortung als Kehrseite der Medaille. Freiheit ohne Verantwortung ist egoistisch und einseitige Interessenverfolgung. Verantwortung wiederum setzt Kompetenz voraus.

Verantwortung als Kehrseite von Freiheit und Agilität

Bezogen auf agiles Projektmanagement heißt dies für uns, dass wir von agilen Projektteams und Führungskräften in agilen Projekten eine höhere Projektmanagementkompetenz erwarten als von solchen aus herkömmlichen Projekten. APM sehen wir, zumindest für mittlere und größere Projekte, nicht als Einstieg in das Thema Projektmanagement, sondern als Zusatzqualifikation.

APM baut auf herkömmlichem PM auf

Erst auf dieser Grundlage, so unsere Erfahrung, gelingt es, auch große Projekte mit 50, 100 oder 200 Projektmitgliedern so aufzusetzen, zu planen, zu steuern und zu führen, dass sie durch die Nutzung agiler Verfahren erfolgreicher werden als ohne agile Techniken.

In der Praxis lässt sich das APM-Verfahren mit Vorgehensmodellen wie dem OEP (oose Engineering Process, [OEP-2007]) oder dem V-Modell XT kombinieren.

Die im APM-Verfahren optional verwendeten Features mit ihren speziellen Ausprägungen Releasefeature, Iterationsfeature etc. haben keinen direkten Bezug zu dem von Jeff DeLuca entwickelten *Feature Driven Development* (FDD). FDD ist eine spezielle Entwicklungsmethodik, die auf feingranularen Features basiert. Zwar gibt es begriffliche und methodische Überschneidungen zu APM, aber die Unterschiede sind wahrscheinlich größer als die Gemeinsamkeiten, sodass wir eine detaillierte Abgrenzung zum Ansatz von DeLuca für nicht relevant halten.

FDD

Auch zu dem Buch „Agile Project Management“ von Jim Highsmith [Highsmith-2004], das gelegentlich auch „APM“ abgekürzt wird, hat unser Buch im Übrigen keinen besonderen Bezug, außer dass das Thema identisch ist.

APM

Die Verantwortung für die Wartung des Produktes kann in dieser Phase von den ursprünglichen Entwicklern in andere Hände übergehen.

Wartung vs. Weiterentwicklung

In den meisten Fällen ist mit „Wartung“ die Weiterentwicklung der Software gemeint. Genauer wäre es, Wartung (Fehlerbehebung, Anpassungen der technischen Konfiguration, Betrieb und Überwachung von Datensicherung, Performance, Datensicherheit etc.) und Weiterentwicklung (Hinzufügen neuer Features, Erweiterung um neue fachliche Anforderungen etc.) zu unterscheiden.

3.2 Die Projektebene planen

Die alleroberster Planungsebene ist der Projekt- und Releaseplan, der die Projektziele, Produktfeatures, wichtige Meilensteine, Releasefeatures und ein grundsätzliches Iterationsraster zusammenfasst.

Vorbereitungsphase ⇒ 63

Je nach Sicherheits- und Vorab-Detaillierungsbedürfnis des Kunden werden diese Aspekte bereits mehr oder weniger während der Vorbereitungsphase des Projektes ausgearbeitet und festgelegt. Diese Ergebnisse werden nun auf den aktuellen Erkenntnis- und Planungsstand gebracht und soweit notwendig um noch fehlende Informationen ergänzt.

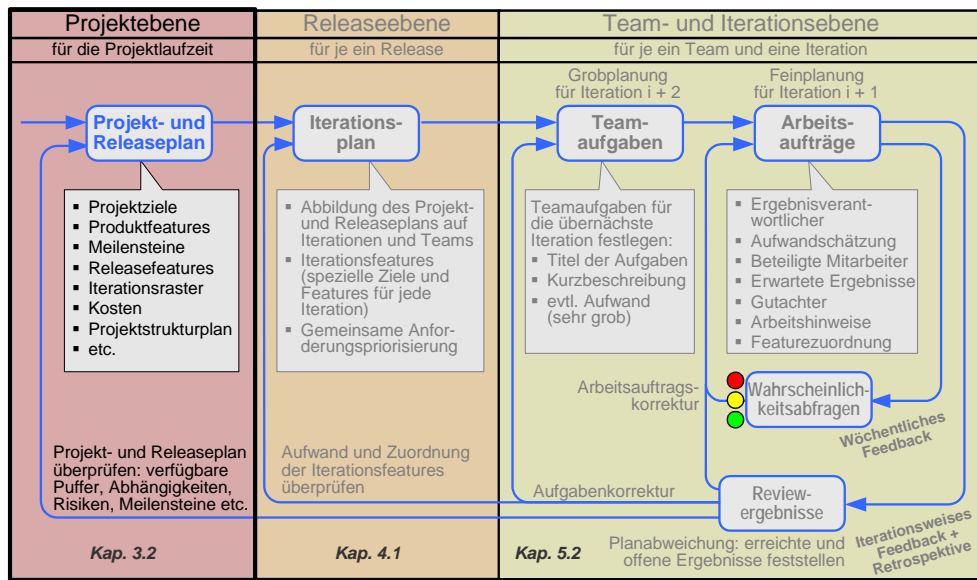


Abb. 3.2-1: Projektebene (Ausschnitt aus Abb. 1.1-3, ⇒4)

Da wir mit einem agilen Vorgehen das Endprodukt üblicherweise als Sequenz inkrementell wachsender Releases herstellen, stellt sich die spannende Frage: Welche Features kommen in welches Release? Noch spannender ist es, wenn das Projekt eine Größenordnung hat, bei der mehrere Produkte mit mehreren Releases parallel gebaut werden müs-

sen. Wie bewältigt man so etwas? Welche Kriterien sind entscheidend? Und wie bringe ich das mit iterativem Vorgehen und featurebasiertem Planen zusammen? Im Folgenden gehen wir auf diese Fragen näher ein.

3.2.1 Releases schneiden

Die weitaus meisten Projekte, die uns in der Praxis begegnet sind, hatten vielleicht noch keine sehr detaillierte Vorstellung von ihrem Inhalt, aber der Preis wie auch der Endtermin waren bereits fest vorgegeben. Sei es, dass der Budgettopf für das kommende Jahr schlichtweg nur X Euro enthält, oder sei es, dass aus Marketinggründen ein bestimmtes Produkt zu einem festgelegten Termin am Markt sein soll.

In einem uns bekannten Fall musste ein Projekt zu einem bestimmten Stichtag fertig werden, weil kurz darauf ein Bereichsleiter das Unternehmen verließ. Selbstverständlich musste „sein“ Projekt vorher noch erfolgreich beendet werden, quasi als sein Meisterstück. Dummerweise wurde es aber nicht fertig, sodass man zum Stichtag ein Release ausrollte, das Projekt kurz anhielt und denselben Inhalt unter neuem Projektnamen (und neuem Bereichsleiter) weiterentwickelte [...].

Es gibt natürlich auch solche Projekte, bei denen zu Beginn zusätzlich auch Inhalt und Güte fest vorgegeben sind („mindestens genauso gut wie das abzulösende Produkt ...“). Okay, wir wissen natürlich auch, dass ein Projekterfolg ziemlich unwahrscheinlich ist, wenn Zeit, Kosten und auch noch Funktionalität und Qualität vorgegeben sind. Aber einigen wir uns einmal darauf, dass wir Inhalt und Qualität in vielen Fällen leichter beeinflussen können als Termin und Budget. Was spricht dagegen, um diese eingerammten Pflöcke herumzuplanen? Der Auftraggeber bekommt zum Stichtag in jedem Fall etwas Brauchbares geliefert, und zwar ungefähr zu dem angedachten Preis.

Umfang und Qualität sind oft leichter beeinflussbar als Termin und Budget

Nicht, dass wir uns missverstehen: Wir verlangen von Ihnen nicht, wider besseren Wissens Betrug zu begehen, sondern wir gehen davon aus, dass Sie als verantwortlicher Projektleiter selbstverständlich die Reißleine ziehen, wenn Sie bemerken: Es geht nicht. Wie aber können wir einen Kompromiss zwischen dem Machbaren und dem politisch Vertretbaren herausarbeiten?

Nun, zunächst einmal akzeptieren wir den Endtermin. Das bedeutet, wir rollen nicht mit den Augen und denken: „Diesen Inhalt kriegen wir bis dahin doch nie fertig!“, sondern wir fragen uns: „Welches sind die für ein erfolgreiches Projekt wichtigsten Features und was davon kriegen wir bis zum Endtermin fertig?“. Diese Denkweise ist ein erster

Was kriegen wir bis zum Endtermin fertig?

kleiner, aber sehr wichtiger Schritt. Die richtige Priorisierung ist der Schlüssel dafür.

Angenommen das Projekt für die Realisierung des Abrechnungs- und Reservierungssystems für unsere Kfz-Vermietung (vgl. Systemkontext in Abb. 2.5-2) soll nach 7 Monaten eingeführt werden (sagen wir aus politischen Gründen). Sie jedoch schätzen nach allen Regeln der Kunst für die bisher gefundenen Anwendungsfälle bzw. Produktfeatures eine Dauer von 19 Monaten. Was tun Sie dann? Richtig! Sie machen das, was jeder erfahrene Projektleiter tun würde: Sie denken über einen Stufenplan nach, also über eine Zerlegung des Umfangs in zwei aufeinander folgende Releases. Sie überlegen für das erste Release: „Auf welche Funktionalität können wir in den ersten 4 Betriebsmonaten verzichten?“ Und das ist, was Sie tun:

In 3 Schritten zum
Stufenplan

- 1. Kompletzt verzichtbare Produktfeatures identifizieren.** Überlegen Sie kurz, welche Produktfeatures unbedingt im ersten Release laufen müssen, zumindest teilweise. Oder umgekehrt: Welche Produktfeatures können möglicherweise komplett im zweiten Release realisiert werden? Vielleicht kommen Sie schon beim ersten Hingucken auf die Idee, dass ein *GPS-Tracking*, eine *Online-Schufa-Auskunft* sowie die *Internetreservierungen* für ein paar Monate komplett verzichtbar wären und erst im zweiten Release produktiv gehen könnten. Selbst eine *Stationsverwaltung* brauchen Sie eventuell nicht sofort, wenn Sie deren Daten im Rahmen der Altdatenübernahme migrieren und somit Änderungen, Neueröffnungen oder Schließungen von Stationen notfalls übergangsweise auch per Hand in der Datenbank nachtragen können.
- 2. Anwendungsfälle priorisieren und zuordnen.** Als Nächstes nehmen Sie sich die Liste der Anwendungsfälle vor und priorisieren jeden einzelnen (*Priorisierungsworkshop* ⇒252). Für die niedrig priorisierten fragen Sie: „Welche von denen können komplett im zweiten Release realisiert werden?“ Möglicherweise gelangen Sie so zu der Erkenntnis, dass die Anwendungsfälle *Kunde zeitweise sperren*, *Kunde kündigen* sowie *Fahrzeug aus dem Bestand nehmen* genauso gut auch erst in Release 2 umgesetzt werden können. Diese sind Teile der Produktfeatures *Zentrale Kundenverwaltung* und *Fuhrparkverwaltung*, sodass beide Produktfeatures in jedem der zwei Releases nur teilweise umgesetzt werden.
- 3. Anwendungsfallinhalte zerschneiden.** Reicht das immer noch nicht, können Sie jeden einzelnen Anwendungsfall gedanklich durchgehen. Wenn diese bereits essenziell beschrieben vorlie-

gen, ist das sehr hilfreich, es geht aber auch mit Nachdenken. Überlegen Sie, welche Schritte eines Anwendungsfalles besonders aufwendig sind und wie sie zunächst einfach gehalten werden können (diese gehören dann ins erste Release). Unter Umständen können Sie auf Tooltips verzichten, aufwendige Grafikgimmicks streichen oder komfortablen Schnickschnack später realisieren. Meistens sind dies Dinge, die der Bequemlichkeit dienen. Vielleicht wollen Sie auch eine bestimmte Nachbarsystemanbindung zunächst nicht über eine automatisierte Synchronisierung lösen, sondern über einen einfachen Datelexport, der zunächst nur per Mail verschickt wird und erst im Release 2 vollständig automatisiert wird.

Das Schneiden von Releases lösen Sie am besten nicht im stillen Kämmerlein, sondern im Team mit fachlichen Domänenexperten und Systemarchitekten. In der Gruppe kommen Sie erfahrungsgemäß auf bessere Lösungen, auf die Sie alleine so nicht gekommen wären. Und Sie haben gleich die richtigen Personen am Entstehungsprozess des Releaseschnittes beteiligt. Das ist viel einfacher, als die eigene Idee hinterher gegen Vorbehalte verteidigen zu müssen.

Releases schneidet man am besten im Team

Erst wenn Sie merken, dass die Terminvorstellungen all Ihrer Bemühungen zum Trotz völlig absurd sind, sollten Sie das Rückgrat durchstrecken und die Projektverantwortung ablehnen. Aber seien Sie gewarnt: In den meisten Fällen gelingt es erstaunlich gut, Funktionalität passend abzuspecken. Der Rest ist dann eine Frage des Verhandlungsgeschickes, d.h., nun können Sie dem Fachbereich bzw. Kunden einen konkreten Vorschlag für einen realistischen Releaseplan unterbreiten und sehr genau darlegen, was wann geliefert wird und warum das so sein muss.

Absurde Terminvorstellungen

3.2.2 Das Projekt mit Releases und Meilensteinen planen

Es gibt viele gute Gründe, ein Produkt inkrementell in mehreren aufeinander aufbauenden Releases zu entwickeln, und nur wenige dagegen. Sie können die Projekte kleiner halten, Risiken reduzieren und damit die Erfolgswahrscheinlichkeit erhöhen. Dafür müssen Sie den Übergang von einem Release auf das nächste handhaben und für diese Zeitstrecke unter Umständen sogar einen Parallelbetrieb in Erwägung ziehen. Daher ist es vielleicht auch sinnvoll, die Gesamtlaufzeit lieber etwas länger einzuplanen als ursprünglich gerechnet.

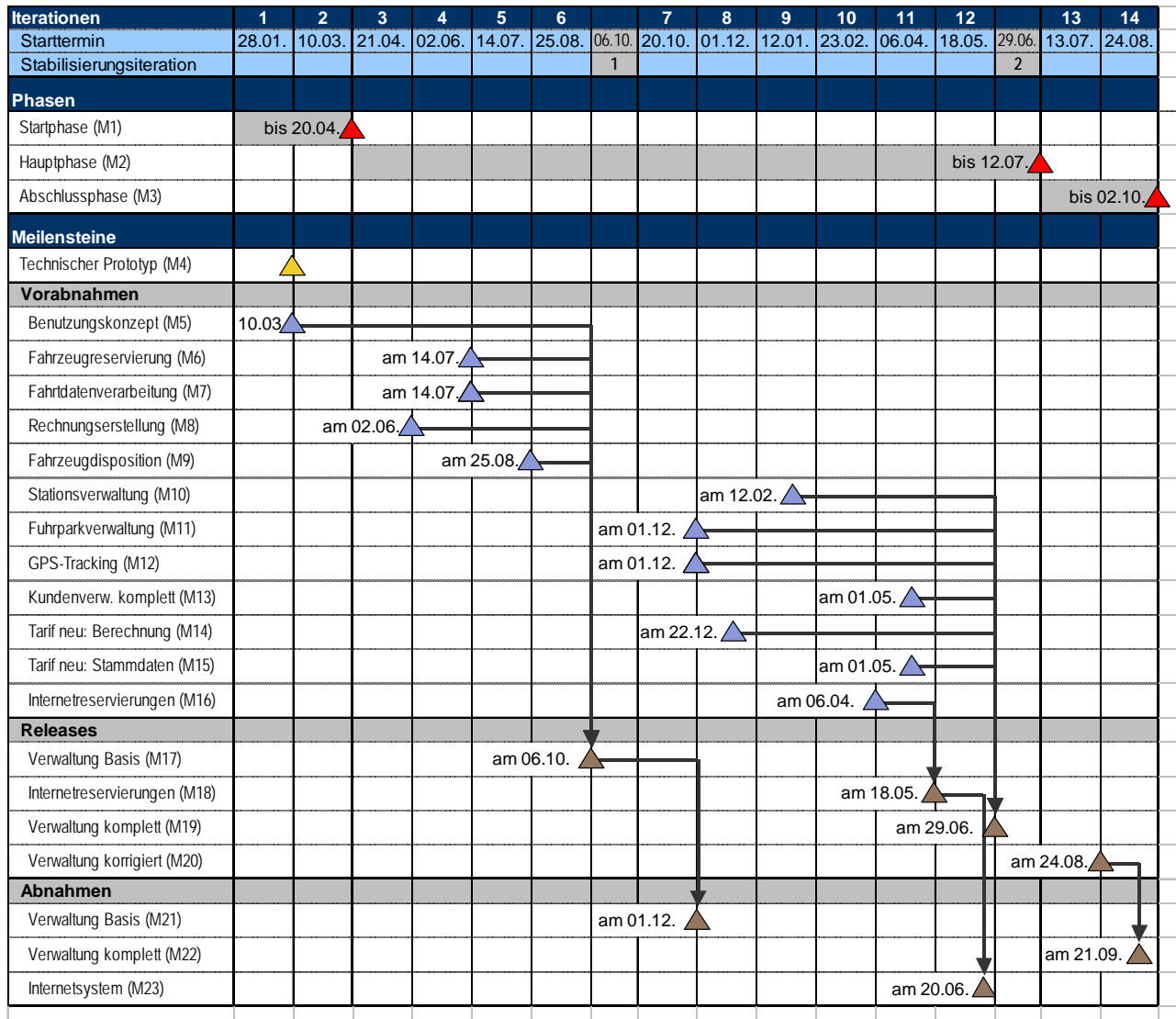


Abb. 3.2-2: Projekt- und Releaseplan mit Meilensteinen für Vorabnahmen, Releases und Abnahmen (Abb. ist identisch mit Abb. 1.1-5 ⇨7)

Das Beispiel in Abb. 3.2-2 ist ein mit einem normalen Tabellenkalkulationsprogramm erstellter Projekt- und Releaseplan, der auch alle geplanten Vorabnahmen enthält. Alle Meilensteine tragen sprechende Namen („Vorabnahme Benutzungskonzept“, „Abnahme Verwaltung komplett“ usw.) und Meilensteinnummern („M5“, „M22“ usw.). Die Meilensteinnummern sind manchmal einfacher zum Zitieren und Verweisen. In diesem Beispiel sind die Nummern ordentlich von oben nach unten aufsteigend angegeben. Spätestens nach einigen Aktualisierungen und Weiterentwicklungen des Plans gerät diese Ordnung durcheinander, wenn die Nummern nicht jedes Mal neu vergeben werden. Von einer Neunummerierung ist jedoch abzuraten, da dann auch jedes Mal alle abhängigen Dokumente, die auf Meilensteine verweisen, aktualisiert werden müssten. Die Meilensteinnummern sind also als unveränderliche Identifikatoren zu betrachten.

Die Releases sind im Beispiel mit fachlichen Bezeichnungen versehen, beispielsweise *Internetreservierungen (M18)*. Alternativ können auch Releasenummern verwendet werden (z.B. „Release 1.2.1“ u.Ä.). Hierbei wäre aber wahrscheinlich auch zu akzeptieren, dass mit den Releasenummern inhaltliche Aspekte beschrieben werden, d.h., wenn sich die inhaltliche Featurereihenfolge ändert, würden auch die Releasenummern nicht mehr sauber aufsteigend folgen.

Wir kennen aus einem Megaprojekt das Phänomen, dass das „Release 8.0“, das schon lange in aller Munde war, vorgezogen wurde und dadurch „Release 5.0“ erst nach Release 8.0 kam.

Ebenfalls alternativ können auch künstliche (frei erfundene) Bezeichnungen ohne jeden Bezug zu Inhalten verwendet werden. Dieses Vorgehen ist von großen Softwarehäusern wie Microsoft bekannt, wo in Pressemitteilungen dann beispielsweise vom Release „Chicago“ die Rede ist. Statt M17 – M20 könnten wir unsere Releases also auch *Arthur, Marvin, Trillian* und *Zaphod* nennen.

Arthur, Marvin, Trillian,
Zaphod

Im Folgenden erläutern wir nun den Projekt- und Releaseplan inhaltlich. Beginnen wir mit der Startphase, die die ersten zwei Iterationen überdeckt. Welche wichtigen und risikobehafteten Ergebnisse müssen innerhalb der Startphase abgesichert werden? Zum einen haben wir hier angenommen, dass ein technischer Prototyp (M4) erstellt wird, mit dem die grundsätzliche Funktionalität der Architektur nachgewiesen wird. Dieser hat im Wesentlichen keine Außenwirkung zum Kunden, sondern ist ein intern bedeutendes Ergebnis.

Technischer Prototyp (M4)

Zum anderen möchten wir das Benutzungskonzept vom Kunden abnehmen lassen. Das Benutzungskonzept beschreibt, nach welchen Prinzipien die Software benutzt werden kann. Dies betrifft zum einen die äußere Gestaltung (GUI-Styleguide), vor allem aber die grundsätzlichen Funktionalitäten, über die die Benutzungsoberfläche verfügen soll. Beispielsweise wie zwischen verschiedenen Eingabedialogen navigiert werden kann: Kann der Benutzer frei aus einem Menü auswählen oder wird er durch einen Assistenten streng geführt? Gibt es grafische und textliche Ein- und Ausgaben? Welche Metaphern werden eingesetzt (z.B. „Postkorb“)? Existiert eine explizite Workflow-Steuerung? Kann die Oberfläche personalisiert werden?

Benutzungskonzept (M5)

Das Benutzungskonzept sollte vom Kunden abgenommen worden sein, bevor in größerer Zahl Dialoge entsprechend diesem Konzept entwickelt werden.

Wichtige Features aus den Bereichen Fahrzeugreservierung, Fahrdatenverarbeitung, Rechnungserstellung und Fahrzeugdisposition werden bereits beginnend mit der Startphase realisiert und dann innerhalb der

ersten zwei bis drei Iterationen fertiggestellt. Anschließend können Sie vom Kunden vorabgenommen werden.

Vorabnahmen und Teilabnahmen ⇒ 220

Als Vorabnahme bezeichnen wir eine Abnahme auf Basis eines normalen Builds, also einer Version, die nicht im Zielsystem, sondern nur in der Entwicklungsumgebung oder einer entwicklungsnahen Testumgebung läuft. Mit einer Vorabnahme wird nur eine begrenzte Menge von Anforderungen oder Features abgenommen und dies auch nur vorbehaltlich der späteren Reproduzierbarkeit im Rahmen einer Abnahme oder Teilabnahme in der Produktionsumgebung.

Priorisierungsstrategie

Die von uns gewählte Priorisierungsstrategie hinter dem Projekt- und Releaseplan zielt darauf ab, anspruchsvolle Features zuerst und triviale und risikoarme Features später umzusetzen. Das führt beispielsweise dazu, dass die relativ einfache Kundenverwaltung spätmöglichst entwickelt wird. Eine Grundfunktionalität wird zwar schon wegen Abhängigkeiten anderer Features frühzeitig notwendig, der größere Teil wird aber erst mit Meilenstein M13 abgeschlossen. Ebenso beispielsweise die *Tarifstammdaten (M15)*.

Release Verwaltung Basis (M17)

Die Meilensteine M6 – M9 stellen bereits ein erstes rudimentäres, aber produktiv benutzbares System dar, weswegen diese zu einem Release *Verwaltung Basis (M17)* zusammengefasst werden. Mit M17 wird dann auch das erste Mal das Freigabeverfahren als solches und das Verhalten der Anwendung in der Produktionsumgebung erprobt.

Releaseteam
Tägliches Build ⇒ 181

Das erste Build aus dem normalen (täglichen) Buildprozess, das alle für dieses Release geforderten Features in stabiler Weise enthält, greifen wir hierzu aus dem normalen Entwicklungsprozess ab und übergeben es einem eigenen Releaseteam zur Überführung in die Produktionsumgebung. Dafür wurde die Dauer einer Iteration eingeplant (Iteration 6 zwischen M9 und M17). Durch die Releaseentwicklung werden wahrscheinlich einige Schwächen deutlich, die in den Entwicklungsprozess zurückgekoppelt werden müssen.

1. Stabilisierungsiteration
⇒ 223, 323

Deswegen haben wir im Anschluss an Iteration 6 eine sogenannte Stabilisierungsiteration vorgesehen, die deutlich kürzer ist und in der keine neue Funktionalität hinzugefügt wird, sondern hauptsächlich an der Robustheit und Stabilität des bisherigen Systems gearbeitet wird.

Abnahme (M21)

Der Kunde hat dann bis zum Ende der 7. Iteration Zeit, die mit dem Release umgesetzten Features abzunehmen (M21).

Verwaltung komplett M19
korrigiert M20

Währenddessen arbeiten die Entwickler bereits an den nächsten Features für die Vorabnahmen M10 – M15, die dann wiederum zu einem Release (*Verwaltung komplett M19*) zusammengefasst werden. Das Release M19 umfasst eigentlich alle vertraglich geforderten Features (außer die Internetreservierungen, aber dazu gleich). Wir gehen aber

davon aus, dass der Kunde noch verschiedene Mängel entdecken wird, wahrscheinlich auch abnahmeverhindernde, sodass wir mit Release M20 nochmals eine korrigierte Version hinterherschicken.

Spätestens mit diesem Release erwarten wir dann die endgültige Abnahme (M22). Unmittelbar nach Release M19 ist nochmals eine kurze Stabilisierungsiteration vorgesehen, um die besonderen Erkenntnisse aus dieser zweiten Version innerhalb der Produktionsumgebung verdauen zu können.

Abnahme Verwaltung
komplett (M22)
2. Stabilisierungsiteration

Die Internetreservierungen sind in der Planung abgetrennt, da hier zumindest das Frontend-System speziellen Anforderungen genügen muss (beispielsweise sind alle gängigen Internetbrowser in allen gängigen Versionen zu unterstützen) und auch im Backend bezüglich der Benutzerverwaltung und Sicherheit besondere, vor allem auch nicht funktionale Anforderungen existieren.

Internetreservierungen M16,
M19 und M23

Weil es sich hierbei letztendlich auch um eine andere Produktionsumgebung (andere Server) handelt, ist die Entwicklung über den Pfad M16, M18 und M23 separat beschrieben.

Zu beachten ist abschließend noch, dass die in Abb. 3.2-2 aufgeführten Vorabnahmen nur einen Teil der jeweiligen Iterationsfeatures widerspiegeln, nämlich den, der für Vorabnahmen wirklich interessant ist. Das heißt vor allem nicht, dass nicht darüber hinaus dem Kunden weitere Versionen präsentiert und mit ihm abgestimmt werden. Vor allem für das Internet-Reservierungssystem (mit dem relativ späten Meilenstein M16) scheint dies mehr als angebracht.

Als Vorabnahmen sind hier vor allem solche Features eingeplant, die das Projekt, nachdem sie entwickelt wurden, möglichst nicht mehr anpassen möchte. Durch die Vorabnahmen kann der Kunde in die Situation gebracht werden, nachträgliche Änderungen an bereits vorabgenommener Funktionalität gegebenenfalls extra bezahlen und Terminverzögerungen des Gesamtprojektes hinnehmen zu müssen (siehe beispielsweise 9.11.3 *Allgemeine Projekt-Geschäftsbedingungen (APGs)*, ⇒394).

Vertragsrelevante
Vorabnahmen

Ebenfalls beachtenswert ist, dass es sich nur um eine erste (oder frühe) Version des Projekt- und Releaseplans handelt. Wie wir im folgenden Kapitel 4 *Releaseplanung* (⇒133) sehen werden, verschieben sich durch Erkenntnisse und Entscheidungen zur Detailplanung noch einige Meilensteine. Möglicherweise wird auch die Startphase nicht mit zwei Iterationen auskommen.

Wenn Sie den Releaseplan in Abb. 3.2-2 im Hinterkopf behalten und sich noch einmal die Abb. 1.4-5 (⇒36) als exemplarische Obermenge möglicher Projektorganisationsformen ansehen, dann könnten Sie nun

Projektorganisation
vgl. ⇒36

auch erste konkretere Annahmen bezüglich der notwendigen Projektorganisation für unser Fallbeispiel treffen.

3.2.3 Parallelreleases planen

Parallele Releaseteams

Der Releaseplan in Abb. 3.2-2 lässt erkennen, dass wir es mindestens an einer Stelle auch mit parallel arbeitenden Releaseteams zu tun bekommen werden. Das Team für *Internet-Release M18* wird zeitlich überlappend zu dem Team für *Verwaltung komplett M19* arbeiten. Aufgrund der starken inhaltlichen und zeitlichen Zusammenhänge können Release M19 und M20 von demselben Releaseteam entwickelt werden. Das Releaseteam für M17 wird mit der 6. Iteration starten und wahrscheinlich bis zur 7. Iteration wieder aufgelöst werden.

In größeren Projekten trifft man oft auf die Konstellation paralleler Releases bzw. Teilprojekte, die regelmäßig Reibungsverluste zur Folge hat.

Wie viele Teilprojekte braucht das Land?

Ob man nun die Releaseentwicklungen desselben Produktes zu einem Teilprojekt zusammenfasst oder jede Releaseentwicklung einzeln als eigenständiges Teilprojekt definiert, ist letztlich Geschmackssache. Wichtig ist eigentlich nur die Unterscheidung zwischen den unterschiedlichen herzustellenden *Produkten* und den *Releases*. Bei den Releases M17, M19 und M20 handelt es sich um das gleiche Produkt in verschiedenen Ausbaustufen. Das Internetsystem ist hingegen als eigenständiges Produkt anzusehen, da es in einer von den anderen unabhängigen Produktionsumgebung läuft.

Wenn die Produkte disjunkt sind, d.h. inhaltlich wie technisch keinerlei Berührungspunkte haben, und Sie über genügend Mitarbeiter und technische Ressourcen verfügen, sind parallele Releaseteams unproblematisch. Alles ist gut.

Herausforderungen einer Parallelentwicklung

Die Schwierigkeit mit parallelen Entwicklungen entsteht genau dann, wenn irgendeine dieser Bedingungen nicht erfüllt ist (oder womöglich mehrere). Stellen Sie sich vor, in den Releases M18 und M19 müsste an derselben Komponente gearbeitet werden. Dann entstehen zwangsläufig Konflikte zwischen den beteiligten Teams. Auch indirekt gibt es Engpässe, nämlich wenn derselbe Fachbereich betroffen ist (der kann sich ja auch nicht beliebig zerreißen) oder wenn mehrere Teilprojekte ihre Anforderungen alle auf einmal den wenigen Datenbankadministratoren vor die Füße werfen.

Wir wollen nichts schönreden: Parallele Entwicklung ist *immer* teuer. Da ändern auch iteratives Vorgehen und Agilität zunächst nichts dran. Aber zwei Dinge können Sie dennoch tun:

Parallelentwicklung beherrschen

■ **Engpässe managen:** Denken Sie das Vorhaben zu Ende und suchen Sie vor allem in der Vorbereitungsphase des Gesamtprojektes genau nach diesen potenziellen Engpässen. Führen Sie sich als Reichsbedenkenträger auf und definieren Sie jeden potenziellen Engpass als Risiko und managen Sie dieses Risiko (vgl. Kapitel 9.10 *Risikomanagement* ⇒371). Spielen Sie außerdem auch Verzögerungen jedes Releases durch und spüren Sie auf, an welcher Stelle sich dann Engpässe zeigen würden. Wenden Sie nach Möglichkeit Critical-Chain-Projektmanagement (⇒296) an.

Critical-Chain-Projektmanagement ⇒296

■ **Gleichtakt herstellen:** Legen Sie nur ein einziges Iterationsraster über alle beteiligten Teams bzw. Teilprojekte. Für jeden im Gesamtprojekt gelten dieselben Iterationstermine. Das erleichtert die Zulieferleistungen zwischen den Teams ganz erheblich! Führen Sie also keine elendig langen Terminlisten, sondern nehmen Sie die Synchronisierung von Zuarbeiten nur an den Abgabeterminen der Iterationen vor. Dadurch erzielen Sie einen mächtigen Gleichtakt im Projekt, der weit über die Grenzen des Projektes zu spüren ist.

In einem uns bekannten Megaprojekt mit 200 Projektmitarbeitern hat die Einführung eines für alle identischen Iterationsrasters etwa 3 – 4 Monate gedauert. Danach wusste wirklich jeder im Schlaf, angefangen beim Sekretariat bis zum Vorstand, in welcher Iteration sich das Projekt gerade befand. Nach etwa 6 Monaten passten sich die Sitzungen des Lenkungskreises den Iterationsterminen an, was vorher als „nicht machbar“ galt. Selbst Organisationseinheiten in der Linie (also außerhalb des Projektes) beugten sich der normativen Kraft des Faktischen. Wer möchte schon gerne Schuld sein, wenn 200 Menschen wegen fehlender Zulieferungen nicht arbeiten können? So geriet der Fachbereich unter Druck, weil die Abgabe von Konzepten zum Beginn der Iteration fertig sein musste. Und wo früher ein Liefertermin eines Projektes von der Verfügbarkeit bestimmter Personen abhing, richtete nun die Betriebsabteilung ihre Urlaubsplanung an bestimmten Iterationsterminen aus.

Die Macht des Gleichtaktes

Natürlich kann man teure Großprojekte verdammen oder sagen: „Vermeide parallele Releaseentwicklung!“ Statistisch gesehen erhöht man mit der Vermeidung von Großprojekten und paralleler Releaseentwicklung die Erfolgswahrscheinlichkeit eines Projektes spürbar. Die einzige Alternative zur parallelen Releaseentwicklung ist die sequenzielle Entwicklung. Diese ist sehr viel billiger. Nur dauert sie eben auch länger. Und für manche Unternehmen ist Time-to-Market nun mal

Ein Wort zu Großprojekten

überlebenswichtig, koste es, was es wolle. Das wiederum ist eine Frage von Prioritäten, die wir zwar gerne hinterfragen dürfen, aber über die wir nur bedingt befinden können.

Projektumfang klein halten

Unsere Aufgabe als Projektleitung besteht in so einem Fall darin, den Projektumfang so klein *wie möglich* zu halten, d.h. unter Berücksichtigung eben dieser Prioritäten. Wir unterstützen jede Projektleitung enthusiastisch, die als Erstes (und immer wieder) fragt: „Wie kann ich Großprojekte und parallele Entwicklung vermeiden?“ Jede Faser des ProjektleiterInnenhirns sollte durchsetzt sein von dem unbändigen Willen, die Dinge einfach zu halten. Erfahrungsgemäß erreichen Sie mit dieser Geisteshaltung mehr, als Sie selbst für möglich halten, und können damit die Risiken halbwegs kontrollieren, die großen Projekten naturgemäß innewohnen.

Und dennoch gibt es manchmal *zu* viel in *zu* kurzer Zeit zu tun. Dann ist unser Job, nach Lösungen zu suchen, die das Vorhaben trotzdem möglich machen, und dem Auftraggeber die Folgen aufzuzeigen, sei es, dass er dafür mehr Geld ausgeben muss oder dass er dafür auf andere Dinge verzichten muss.

Erfolgreiche Projekte
erfordern sorgfältige
Vorbereitung

Großprojekte hin oder her, zwei oder viele Teilprojekte, parallele Releases oder auch nicht, klar ist: Ein erfolgreiches Projekt beginnt mit einer sorgfältigen Vorbereitung. Und der Projekt- und Releaseplan ist ein wichtiger Baustein dafür.

9.5.8 Dreipunktschätzung

Kurzbeschreibung

Einfache Schätztechnik unter Berücksichtigung der Wahrscheinlichkeiten von Schätzwerten.

Stichworte und verwandte Themen

Schätzen, Delphi-Methode, Schätzkurve, Wetter-von-gestern

Indikationen

- Die Wahrscheinlichkeiten für Schätzwerte stellen sich vom Mittelwert in eine Richtung verschoben dar. Dies kann durch die Dreipunktschätzung näherungsweise berücksichtigt werden.
- Der Schätzwert soll bewusst eine andere Wahrscheinlichkeit als 50 % haben, um z.B. Risiken definiert zu berücksichtigen.

Beschreibung

- Ein Moderator wird ernannt.
- Das Gesamtschätzobjekt wird in möglichst kleine, einzeln zu schätzende Teile z.B. Arbeitspakete zerlegt.
- Es folgt die Diskussion der Aufgabenstellung:
 - ◆ Was soll geschätzt werden?
 - ◆ Was soll nicht geschätzt werden?

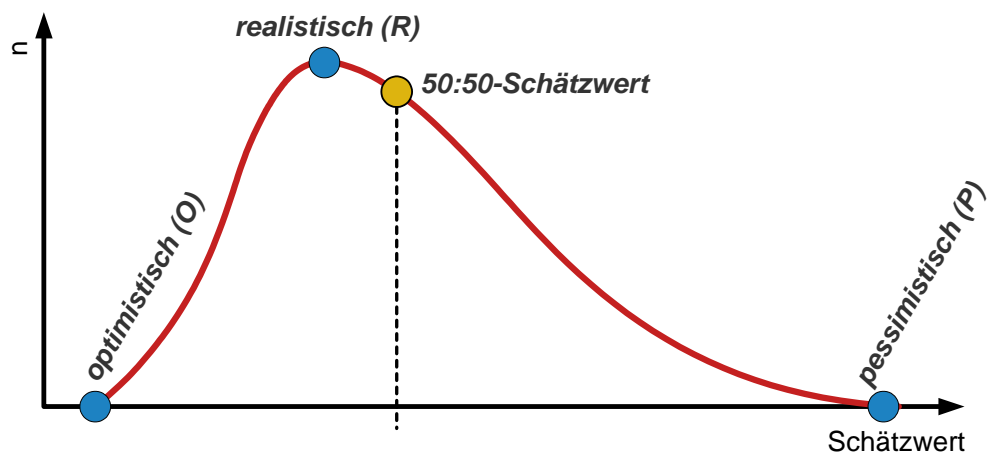


Abb. 9.5-6: Schema der Dreipunktschätzung

- Es werden für jeden zu ermittelnden Schätzwert nicht nur einer, sondern drei Werte geschätzt (Abb. 9.5-6):
 - ◆ Optimistischer Wert: der Idealfall, der minimal notwendige Aufwand bzw. kürzeste Dauer.

- ♦ Realistischer Wert (modal): der Wert mit der höchsten Wahrscheinlichkeit, häufigster Aufwand bzw. häufigste Dauer.
- ♦ Pessimistischer Wert (Worst Case): höchster Aufwand bzw. längste Dauer.

Der Schätzwert (Mittel in Abb. 9.5-6) ist der Wert mit einer Wahrscheinlichkeit von 50 Prozent. Er halbiert die unter der Kurve liegende Fläche in zwei gleich große Teile. Der Schätzwert, seine Standardabweichung und die Unsicherheit lassen sich über die folgenden drei Formeln aus den drei Ursprungsschätzwerten näherungsweise bestimmen.

Der 50:50-Schätzwert:

$$\text{Schätzwert} = \frac{\text{optimistisch} + (4 \times \text{realistisch}) + \text{pessimistisch}}{6}$$

Die Standardabweichung für jeden Schätzwert:

$$\text{StdAbw} = \frac{\text{pessimistisch} - \text{optimistisch}}{6}$$

Die Unsicherheit für alle Schätzwerte aus der Summe der Quadrate ihrer Standardabweichungen:

$$\text{Unsicherheit} = \sqrt{\sum (\text{StdAbw})^2}$$

Die Dreipunktschätzung ist eine Geradennäherung einer β -Verteilung (Abb. 9.5-7). Durch eine unterschiedliche Aufteilung der Fläche unterhalb der Kurve können Schätzwerte mit geringerer oder höherer Wahrscheinlichkeit abgeleitet werden.

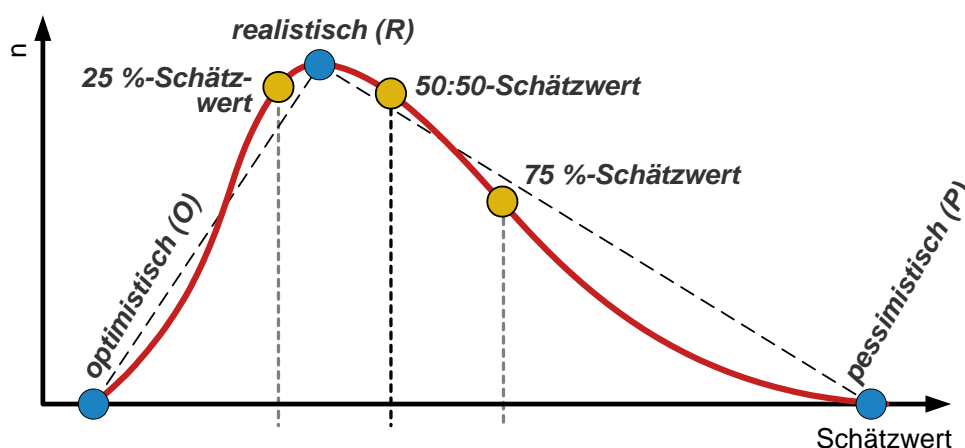


Abb. 9.5-7: Unterschiedliche Wahrscheinlichkeiten

Weiterführendes

PERT: Programming Evaluation and Review Technique.

9.7 Qualitätsmanagement

9.7.1 PMBoK-Rahmenwerk

Projektqualitätsmanagement erstreckt sich über zwei Ebenen: die Qualität des Projektes selbst und die des Produktes, der Befähigung zur Dienstleistung oder eines sonstigen Ergebnisses. Sie setzt die qualitätsrelevanten Richtlinien, Zielsetzungen und Verantwortungen der ausführenden Organisation um und beinhaltet dazu die Prozesse Qualitätsplanung, Qualitätssicherung und Qualitätssteuerung.

Prozesse des Qualitätsmanagements sind:

- **Qualitätsplanung** – Festlegung, welche Qualitätsstandards für das Projekt gültig sind und wie diese zu erfüllen sind. Der Prozess sollte parallel zu den anderen Planungsprozessen durchgeführt werden.
- **Durchführung der Qualitätssicherung** – Qualitätssicherung (*Quality Assurance*) ist die geplante und proaktive Anwendung von Maßnahmen, die sicherstellen sollen, dass alle qualitätsrelevanten Prozesse durchgeführt werden. Eines der wichtigsten Qualitätssicherungswerkzeuge ist das Qualitätsaudit.
- **Durchführung der Qualitätssteuerung** – Ziel der Qualitätssteuerung (*Quality Control*) ist, durch Überprüfung von Arbeitsergebnissen deren Übereinstimmung mit den Qualitätsnormen festzustellen und bei Abweichungen Wege zu finden, die Ursachen zu beseitigen. Im Mittelpunkt der Qualitätssteuerung stehen Überprüfungen von Arbeitsergebnissen (*Inspections*) und deren Auswertung.

Qualitätssicherung

9.7.2 Externe Projektreviews

Kurzbeschreibung

Entweder regelmäßig oder gelegentlich zu ausgewählten Themen werden externe Experten zur Begutachtung von Teilergebnissen herangezogen, um frühzeitig Sicherheit zur Tragfähigkeit von Lösungen zu erlangen und mögliche Schwachstellen zu entdecken und um gegebenenfalls weitere oder alternative Ideen und Lösungsansätze kennenzulernen.

Stichworte und verwandte Themen

Risikomanagement

Indikationen

- Sicherheit zur Tragfähigkeit von Lösungen erlangen
- Schwachstellen entdecken
- Neue Idee und Lösungsansätze kennenlernen

Beschreibung

Sofern regelmäßig, beispielsweise iterationsweise, der aktuelle Projektstand von externen Experten begutachtet werden soll, dann ist es eher sinnvoll, jedes Mal andere Experten zu engagieren, um immer wieder neue Ideen und Sichtweisen und neue mögliche Schwachstellen zu entdecken.

Wenn der Fokus auf einem speziellen Thema liegt, kann eine spätere erneute Begutachtung durch Externe den zwischenzeitlichen Umgang mit dem Thema kritischer würdigen.

Die externen Reviews oder Audits müssen nicht unbedingt sehr lang oder ausufernd sein. Im einfachsten Fall stellen Projektmitarbeiter den Gutachtern eine Lösung bzw. ein Arbeitsergebnis vor, die Gutachter erhalten noch ein paar Stunden oder ein bis zwei Tage Zeit, die Lösung und gegebenenfalls weiterführende Dokumente zu studieren und ihre Thesen zu hinterfragen, und präsentieren dann im unmittelbaren Anschluss ihre Erkenntnisse.

Wir kennen große Projekte, in denen regelmäßig solche externen Reviews stattfanden, für die von vornherein das Budget eingeplant war. Manchmal wurden sehr namhafte Einzelexperten geholt und manchmal haben auch namhafte Firmen ihre Experten geschickt. Selbst bei gleichen Themenbereichen wurden sehr unterschiedliche, aber meistens hilfreiche Erkenntnisse geliefert. Manchmal erkannte das Projekt auch nur, dass es selbst viel weiter ist als der sogenannte Experte, aber eine solche Bestätigung hat ja auch einen Wert.

9.7.3 Autor-Kritiker-Treffen

Kurzbeschreibung

Ein Autor-Kritiker-Treffen ist ein dreiteiliges Arbeitstreffen bei dem

- ein oder mehrere Autoren eine Lösung, ein Arbeitsergebnis o.Ä. vorstellen und dabei von den übrigen Teilnehmern (den Kritikern) nicht unterbrochen werden;

- die übrigen Teilnehmer (Kritiker) den vorgestellten Gegenstand kritisieren, d.h. ihre Eindrücke, Gedanken, Bemerkungen, Bewertungen etc. hierzu darlegen, und dabei nicht durch die Autoren unterbrochen werden;
- der oder die Autoren sich zurückziehen und das erhaltene Feedback für sich aus- und bewerten.

Stichworte und verwandte Themen

Feedback, Autor-Kritiker-Zyklus, konstruktive Qualitätssicherung

Indikationen

- Eine Person oder ein Team möchte von sachverständigen Kollegen Feedback zu einem Arbeitsergebnis erhalten.
- Aufspüren von Schwachstellen, Stärken und Schwächen eines Arbeitsergebnisses oder Produktes
- Sammlung von Ideen für die Qualitätsverbesserung eines Arbeitsergebnisses

Beschreibung

Feedback zu geben und zu nehmen ist gar nicht so einfach. Wenn man nicht aufpasst, kommt es zu Missverständnissen, emotionalen Verletzungen, Vorwürfen, Demütigungen, Demotivation und Ähnlichem.

Das Autor-Kritiker-Treffen bietet eine konstruktive Struktur, sachlich, mit Wertschätzung und frei von emotionalen Verletzungen Kritik zu erfahren und daraus zu lernen.

Vorgehensweise

Vorbereitung:

- Die Initiative zu dem Treffen geht normalerweise von den Autoren aus, die gerne Kritik erfahren möchten. Andere Personen, bspw. die Projektleitung, Vorgesetzte oder Kollegen, sollten das Treffen nicht initiieren, sondern können den Autoren lediglich einen unverbindlichen Vorschlag hierzu machen. Die Autoren legen den zu bearbeitenden Gegenstand fest.
- Ein neutraler Moderator wird festgelegt, der das Treffen leiten wird.
- Die Teilnehmer (Kritiker) werden ausgewählt und unter Angabe des Themas/Gegenstandes, Ort, Zeitpunkt und Dauer des Treffens eingeladen. Sofern es hilfreich ist, werden den eingeladenen Kritikern rechtzeitig alle relevanten Unterlagen zugestellt, damit die Kritiker sich vorab einarbeiten und vorbereiten können.

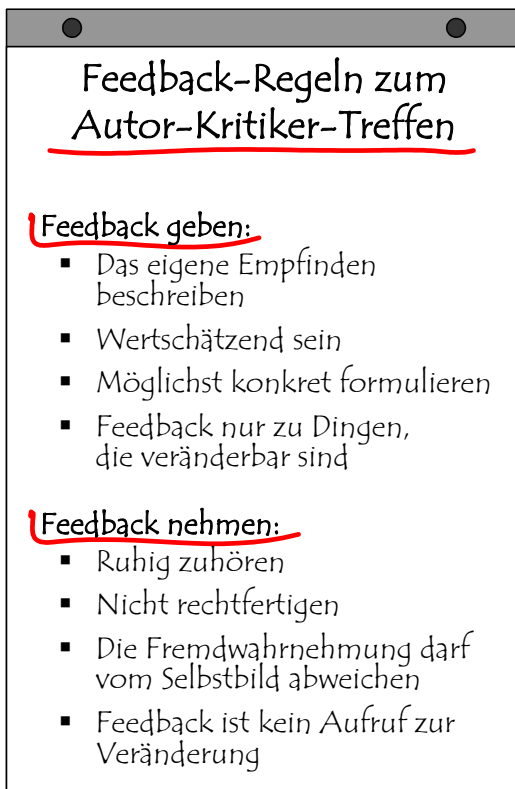


Abb. 9.7-1: Feedback-Regeln zum Autor-Kritiker-Treffen

Ablauf des Treffens:

- Der Moderator stellt die Autoren und ihr Thema vor und erläutert die Regeln sowie den geplanten Ablauf des Treffens. Vor allem hat der Moderator darauf hinzuweisen, dass die Kritiker sich zunächst nicht zur Präsentation äußern dürfen.
- Der oder die Autoren stellen ihren zu kritisierenden Gegenstand vor, bspw. ein Arbeitsergebnis. Die Kritiker hören zu, notieren sich Kritik, Anmerkungen, Fragen etc. hierzu, dürfen diese jedoch nicht äußern. Lediglich Verständnisfragen sind zulässig. Auch nonverbale Äußerungen, bspw. bewertende Gesten oder Körperhaltungen, sind zu vermeiden. Der Moderator achtet auf die Einhaltung dieser Regel und interveniert gegebenenfalls sofort.
- Der Moderator bedankt sich bei den Autoren für die Präsentation.
- Der Moderator erläutert die Feedback-Regeln. Er wiederholt diese auch dann, wenn alle Beteiligten diese bereits kennen sollten. Hilfreich ist eine kurze Zusammenfassung auf einem Flipchart, das während der Kritikphase für alle sichtbar bleibt.
- Der Moderator bittet die Kritiker ihre Bemerkungen, Kritik, Ideen und Fragen vorzutragen. Die Autoren können und sollten sich die Kritik notieren. Sie dürfen jedoch keine Stellung hierzu nehmen:

keine Rechtfertigungen, Erläuterungen, Richtigstellungen oder Widersprüche, auch dann nicht, wenn die Kritik offensichtlich falsch oder ungerecht ist. Lediglich Verständnisfragen sind zulässig. Auch nonverbale Äußerungen, bspw. bewertende Gesten oder Körperhaltungen, sind zu vermeiden. Der Moderator achtet auf die Einhaltung dieser Regel und interveniert gegebenenfalls sofort.

Um eine Fokussierung auf einzelne Kritiker zu vermeiden, ist es hilfreich, wenn die Kritiker reihum jeweils nur einen Punkt vortragen, was so lange fortgesetzt wird, bis alle Kritiker alle Kritikpunkte vorgetragen haben. Kritiker, die keine weitere Kritik mehr haben, werden in der Reihe übersprungen. Ein Kritikpunkt, der schon vorgetragen wurde, sollte von einem anderen Kritiker nur wiederholt werden, wenn dieser etwas Wichtiges hinzuzufügen oder eine andere Bewertung hierzu hat.

Je nach dem Gegenstand des Treffens kann es sinnvoll sein, von den Kritikern zu jedem Punkt explizit eine Einordnung in folgende Kategorien zu verlangen: 1. Offene Frage, 2. Verbesserungsidee, 3. Hinweis auf eine negative Eigenschaft bzw. einen Mangel, 4. Hinweis auf besonders positive Eigenschaft, 5. Hinweis auf einen Show-Stopper bzw. K.O.-Kriterium, der jede weitere Diskussion obsolet macht und den Moderator zum Abbruch des Autor-Kritiker-Treffens auffordert.

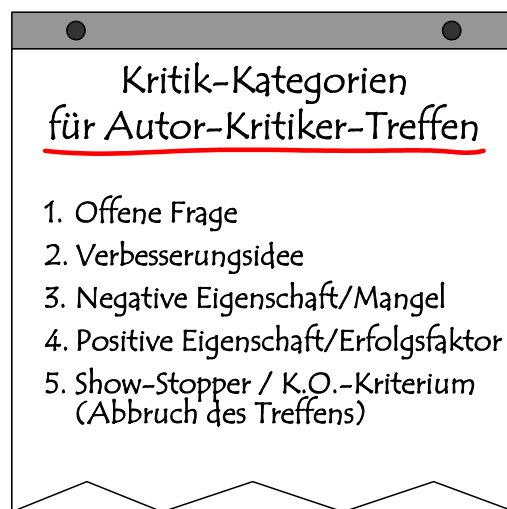


Abb. 9.7-2: Kritik-Kategorien für das Autor-Kritiker-Treffen

- Der Moderator bedankt sich bei den Kritikern und verabschiedet diese und bittet die Autoren, sobald sie wieder alleine sind, sich ausreichend Zeit zu nehmen, die erhaltene Kritik auszuwerten und zu versuchen, möglichst viel daraus zu lernen. Er bittet die Autoren abschließend, auch zu späteren Zeitpunkten gegenüber den Kritikern möglichst keine Stellung zu beziehen, sondern es einfach dabei zu belassen.
- Die Autoren werten die Kritik aus und lernen daraus.

Weiterführendes

Das Autor-Kritiker-Treffen ist eine Gruppenvariante des normalen (paarweisen) Feedbacks. Siehe *Feedback* ⇨339. Diese Technik ist auch unter dem Begriff Autor-Kritiker-Zyklus bekannt. Beispielsweise in der WAM-Methode [Züllighoven-1998].

9.7.4 Retrospektivenworkshop

Kurzbeschreibung

Eine Retrospektive ist, wie der Name sagt (lat.: *retrospectare* = rückblicken), ein Rückblick. Im hier verwendeten Kontext verbinden wir eine Retrospektive jedoch immer mit dem Ziel, etwas für die Zukunft zu lernen und zu verbessern.

Stichworte und verwandte Themen

Moderation, Verbesserungsprozess, Präparationsworkshop

Indikationen

- Ende einer Iteration
- Abschluss eines Meilensteins, eines Releases, einer Abnahme etc.
- Bedürfnis nach Verbesserung der Strukturen und Prozesse der Arbeit

Beschreibung

Die Ergebnisse und die aktuelle Istsituation werden dahingehend kritisch gewürdigt und geprüft, um herauszufinden, was zukünftig besser laufen könnte, was sich bewährt hat, was nicht gut funktioniert hat usw.

Entsprechend lauten die Kernfragen der Retrospektive:

- Was lief gut? Was hat sich bewährt?
- Was hat nicht richtig funktioniert? Was hat sich nicht bewährt?
- Was und wie wollen wir zukünftig verbessern?

Kernfragen

Retrospektiven können auf verschiedenen Organisationsebenen stattfinden: innerhalb der Entwicklungsteams, in der Teamleitungsrunde, in der Gesamtprojektleitung, innerhalb von Releaseteams, Architektorteams usw.

Zielgruppe, Teilnehmer

Retrospektiven sind Workshops von 20 Minuten bis ca. 2 Stunden Dauer, die moderiert werden sollten und zu denen die Teilnehmer explizit eingeladen werden.

Teilnehmerauswahl
Offenheit
Managementbeteiligung

Die Auswahl der Teilnehmer sollte stets eine bewusste Entscheidung sein. Es sollten immer ein komplettes Teams und nach Möglichkeit auch nur dieses eingeladen werden. Teamfremde Personen, außer gegebenenfalls der Moderator, gehören nicht dazu. Einzelne Personen eines Teams auszuschließen, ist in der Regel sozial destruktiv. Lediglich die Frage, ob Führungskräfte (Teamleitung, Projektleitung etc.) dabei sind, ist variabel zu beantworten. Dies betrifft die Offenheit der Teilnehmer – wenn Ängste, wenig Vertrauen oder Unbehagen existieren ist es im Zweifelsfall besser, das Team bleibt unter sich.

Ergebnissicherung
und -weitergabe

Das Gleiche gilt für die Ergebnisse. Auch hier ist vorher zu entscheiden und allen mitzuteilen, ob oder welche Ergebnisse des Workshops an Dritte weitergegeben werden.

Unter Umständen ist es zweckmäßig, den Betrachtungsgegenstand einzugrenzen oder überhaupt klarer festzulegen. Die Retrospektive könnte beispielsweise eingeschränkt werden auf teaminterne Kommunikation (Metaebene, Metakommunikation), Werkzeuge, Strukturen, Verantwortungsteilung oder anderes.

Systemmodell ⇒369

Für den Fall, dass nicht die oben genannten Standardfragen verwendet werden sollen, empfehlen wir, das *Systemmodell* (⇒369) als Ausgangspunkt zu nehmen.

Eine externe Moderation ist vor allem dann wichtig, wenn Konflikte, soziale Spannungen, Ängste oder gar Aggressionen zu erwarten sind.

Dass externe Moderation nicht nur in (agilen) Projekten, sondern auch für Unternehmen insgesamt hilfreich sein kann, haben wir (oose) einmal in einem für uns ungewöhnlichen Kundenauftrag erlebt. Für ein mittelständisches IT-Unternehmen mit weit über 100 Mitarbeitern haben wir an einem Tag zeitgleich für 10 verschiedene Abteilungen jeweils eigene Workshops unter Ausschluss der Führungskräfte durchgeführt, um der Unternehmensleitung anschließend anonym eine Rückmeldung über die Situationseinschätzung und Veränderungswünsche der Mitarbeiterschaft zu geben. Ausgerüstet mit der gleichen Agenda haben dann 10 oose-Moderatoren strukturell identische Workshops durchgeführt und anschließend dem Management die Augen geöffnet.

Checkliste

- Wer soll teilnehmen?
- Was soll betrachtet werden (welcher Zeitraum, welche Arbeitsbereiche etc.)?
- Wer moderiert?
- Wer sichert und dokumentiert die Ergebnisse?
- Wann und wo soll der Workshop stattfinden?

- ☑ Was ist inhaltlich vorzubereiten? Gegebenenfalls vorher Standardfragen auf Flipchart schreiben.
- ☑ Welche Hilfsmittel und Materialien werden benötigt? In der Regel: Pinnwand, Karten, Stifte, Pinnadeln.

Weiterführendes

Bücher speziell zu Retrospektiven: [Kerth-2001] und [Derby-2006].

In folgenden Büchern finden sich kurze lesenswerte Kapitel über Retrospektiven: [Eckstein-2004, S. 92 ff.], [Cockburn-2003, S. 245 ff.] .

Mehr zu Fragetechniken, Kommunikation, sozialen und psychologischen Aspekten finden Sie in [Vigenschow-2007].

9.7.5 Präparationsworkshop

Kurzbeschreibung

Workshop zur Vorbereitung einer unmittelbar bevorstehenden oder begonnenen Phase oder Abschnittes

Stichworte und verwandte Themen

Retrospektive, Projektplanung, Risikomanagement

Indikationen

- Ein neuer grundlegender Arbeitsabschnitt beginnt, beispielsweise
 - ◆ eine neue Phase oder
 - ◆ die Arbeit an einem neuen Release.

Beschreibung

Bevor die Arbeit in einer neuen Entwicklungsphase so richtig startet, wird mithilfe eines kurzen Workshops versucht,

- die besonderen Herausforderungen der Phase zu erkennen,
- Klarheit über die eigenen Ziele, Absichten und Möglichkeiten zu erlangen
- und alle Betroffenen in diese Erkenntnisse einzubeziehen.

Der Workshop sollte moderiert werden, ein schriftlich formuliertes Thema haben („Worauf kommt es <Name des Arbeitsabschnitt> an?“), und es sollten geeignete Teilnehmer eingeladen werden.

Für die Auswahl der Teilnehmer sind folgende Kriterien relevant:

- Wer kann inhaltlich etwas beitragen?

- Wer soll die Erkenntnisse und Ergebnisse erhalten/erfahren? Direkt am Workshop teilzunehmen, auch in einer passiven Rolle, ermöglicht eine nachhaltigere Einbeziehung und ein tiefer gehendes Verständnis als eine nachträgliche Zusammenfassung oder Ergebnisprotokoll.
- Stakeholder, die inhaltliches Diskussionsobjekt sein könnten, also über die diskutiert wird, sollten möglicherweise nicht eingeladen werden.

Normalerweise ist der Präparationsworkshop eine rein projektinterne Veranstaltung.

In der einfachsten Form läuft der Workshop wie folgt ab:

- Einführung und Erläuterung des aktuellen Kontextes: Was steht jetzt an? Was ist der Betrachtungsgegenstand dieses Workshops? Welche Ziele sind von außen (bspw. durch den Auftraggeber oder Vertrag) vorgegeben?
- Es wird auf die (unten stehenden) Standardfragen hingewiesen.
- Die Teilnehmer schreiben ihre Stichworte und Antworten auf Karten. Je eine Karte pro Stichpunkt.
- Die Karten werden auf der Pinnwand thematisch gegliedert (Cluster). Gegebenenfalls werden Verständnisfragen geklärt.
- Die Gliederungsbereiche (Cluster) werden kurz priorisiert.
- In der Reihenfolge ihrer Priorität werden die Gliederungsbereiche diskutiert und mit einem Stichwort oder kurzen Satz gemeinsam neu formuliert.
- Die gering priorisierten Bereiche werden gegebenenfalls ausgelassen. Normalerweise reichen die 5 – 10 wichtigsten Punkte.

Siehe hierzu auch das Beispiel in Kapitel 2.2.2 *Die Weichen richtig stellen* (⇒68).

Standardfragen

- ⊙ Was sind jetzt die typischen Herausforderungen?
- ⊙ Wer sind die wichtigsten Stakeholder in dieser Phase?
- ⊙ Was darf jetzt nicht passieren?
- ⊙ Was sollte jetzt neu eingeführt werden?
- ⊙ Was ist jetzt (viel) einfacher als in späteren Phasen?
- ⊙ Was sind *unsere* Ziele für diese Phase?
- ⊙ Wann ist die Phase beendet?
- ⊙ Welche Ergebnisse und Meilensteine sind zu erzielen?

10 Anhang

10.1 Glossar

Das Glossar wurde freundlicherweise von der Firma oose Innovative Informatik GmbH zur Verfügung gestellt und befindet sich in aktueller Version unter <http://www.oose.de/glossar>. Unter Angabe der Quelle *oose Innovative Informatik GmbH* und der URL <http://www.oose.de/glossar> darf dieses Glossar vollständig oder ausschnittsweise jederzeit kostenlos weitergegeben werden, solange die einzelnen Definitionen unverändert bleiben.

Abhängigkeitsmanagement

Teildisziplin der Softwarearchitektur, die das Ziel verfolgt, störende Abhängigkeiten zwischen den Einzelteilen einer Software zu optimieren. Hierzu werden Abhängigkeiten mit Metriken bzgl. ihres Störungspotenzials bewertet und mithilfe verschiedener Entwurfsmuster aufgelöst oder reduziert.

Ablauforganisation

Ablauforganisation ist die zeitliche und räumliche Ordnung betrieblicher Prozesse innerhalb des durch die Aufbauorganisation geschaffenen Rahmens.

Abnahme ⇒ Vorabnahme, ⇒ Teilabnahme, ⇒ Abnahmetest

Verbindliche und vertragsrelevante Bestätigung des Auftraggebers über die Erfüllung von Anforderungen.

Abnahmetest

Test gegen die expliziten Anforderungen des Auftraggebers/Anwenders, wie sie in einem Anforderungsdokument (z.B. Pflichtenheft oder Fachkonzept) für beide Seiten verbindlich festgelegt sind, sowie gegen die implizierten Erwartungen des Auftraggebers, die dem allgemeinen Stand der Technik entsprechen. (Auch Systemtest aus Sicht des Auftraggebers.)

Abstraktion

Abstraktion ist eine Methode, bei der unter einem bestimmten Gesichtspunkt die wesentlichen Merkmale eines Gegenstandes oder Begriffes hervorgehoben werden.

Akzeptanztest

Im Allgemeinen ein Abnahmetest aus Sicht des Benutzers. Unter Umständen auch die Teilmenge aller vorhandenen Testfälle, um eine bestimmte Teststufe zu bestehen.

Alpha-Test ⇒ Beta-Test

Test oder Erprobung einer vorläufigen Softwareversion durch repräsentative Kunden oder Anwender in einer Testumgebung des Herstellers.

Analyse

Die systematische Untersuchung eines Sachverhaltes oder Gegenstandes hinsichtlich bestimmter Aspekte, Faktoren oder Eigenschaften, die ihn bestimmen.

Anforderung

Eine Anforderung beschreibt eine oder mehrere Eigenschaften oder Verhaltensweisen, die stets erfüllt sein sollen.

Anwendungsfall ⇒ Geschäftsanwendungsfall, ⇒ Systemanwendungsfall

Ein Anwendungsfall beschreibt einen zusammenhängenden Arbeitsablauf aus der Sicht seiner Akteure. Ein Anwendungsfall wird stets durch einen Akteur bzw. Ereignis initiiert und führt zu einem für die Akteure wahrnehmbaren Ergebnis. Ein Anwendungsfall beschreibt das gewünschte externe Systemverhalten aus der Sicht und in der Sprache der Anwender (d.h. in natürlicher Sprache) und somit Anforderungen, die das System zu erfüllen hat. Beschrieben wird, was es leisten muss, aber nicht, wie es dies leisten soll. Es werden verschiedene Arten von Anwendungsfällen unterschieden.

den: Geschäftsanwendungsfall, Systemanwendungsfall.

Anwendungsfall-Feature-Matrix ⇒ Anwendungsfall, ⇒ Feature

Die Anwendungsfall-Feature-Matrix ist eine Tabelle, in der Anwendungsfälle (Zeilen) und Iterationsfeatures (Spalten) einander zugeordnet werden, sodass daraus deutlich wird, in welche Iterationsfeatures ein Anwendungsfall zerlegt wird bzw. welche Anwendungsfälle ein Iterationsfeature umfasst.

Arbeitsauftrag

Ein Arbeitsauftrag ist eine Vereinbarung von und für ein oder mehrere Projektmitglieder, ein Ergebnis zu erzielen. Er beschreibt Voraussetzungen, Rahmenbedingungen und erwartete Ergebnisse.

Architekturabhängigkeitsmanagement

⇒ Abhängigkeitsmanagement

Audit

Eine systematische und unabhängige Untersuchung, um festzustellen, ob bestimmte (meistens qualitätsbezogene) Tätigkeiten und die damit zusammenhängenden Ergebnisse den geplanten Vorgaben und Zielen entsprechen.

Aufbauorganisation

Die Aufbauorganisation beschreibt die Zuständigkeiten für die arbeitsteilige Aufgabenerfüllung im Unternehmen. Sie legt Organisationseinheiten und deren hierarchische Beziehungen untereinander fest.

Aufgabe ⇒ Arbeitsauftrag

Eine Aufgabe ist ein grober Arbeitsauftrag oder eine grobe Beschreibung einer Menge inhaltlich zusammenhängender Arbeitsaufträge.

Aufwandsfortschrittsindikator ⇒ Fertigstellungsgrad

Kennzahl: $\text{Verbraucher Aufwand} * 100 / \text{Geplanter Aufwand}$.

AW

Abkürzung für Aufwand. AW_i = Istaufwand, AW_p = Planaufwand.

Balanced Scorecard ⇒ Strategiekarte

Die Balanced Scorecard ist ein Instrument der Unternehmensführung und ein

systemischer Ansatz, der hilft, zielgerichtetes strategisches Denken und Handeln auf allen Ebenen im Unternehmen zu fördern. Die Grundidee nach [Kaplan-2001] besteht darin, strategische Führungsziele durch die Messung von Kennzahlen in den vier Bereichen Finanzperspektive, Kundenperspektive, Interne Prozessperspektive sowie Lern- und Entwicklungsperspektive in operative Maßnahmen umzusetzen.

Baseline

Eine Baseline ist eine Ausgangsbasis für Entscheidungen. Man könnte sie als Kopie der zum Zeitpunkt der Entscheidungen gültigen Planung interpretieren, die sicher verwahrt wird. Später kann sie zu einem Soll-Ist-Vergleich der dann aktualisierten Planung und der tatsächlichen Projektleistung herangezogen werden. Die wichtigste der Baselines ist die Kostenbaseline, auf deren Basis die ⇒ Earned-Value-Analyse durchgeführt wird, mit der Fortschrittmessung für Arbeit und Kosten möglich wird.

Bearbeitungsgrad ⇒ Fertigstellungsgrad

Benutzungskonzept

Das Benutzungskonzept beschreibt, nach welchen Prinzipien die Software benutzt werden kann. Dies betrifft zum einen die äußere Gestaltung (GUI-Styleguide), vor allem aber, über welche grundsätzlichen Funktionalitäten die Benutzungsoberfläche verfügen soll. Beispielsweise wie zwischen verschiedenen Eingabedialogen navigiert werden kann: Kann der Benutzer frei aus einem Menü auswählen? Oder wird er durch einen Assistenten streng geführt? Gibt es grafische und textliche Ein- und Ausgaben? Welche Metaphern werden eingesetzt (z.B. „Postkorb“)? Existiert eine explizite Workflow-Steuerung? Kann die Oberfläche personalisiert werden?

Beta-Test ⇒ Alpha-Test

Test oder testweiser Betrieb einer vorläufigen Softwareversion durch repräsentative Kunden oder Anwender in der Einsatzumgebung des Kunden bzw. Anwenders.

Big Bang

Bezeichnet die komplette Einführung eines neuen Softwaresystems auf einen Schlag (im Gegensatz zu einer schrittweisen Einführung).

Broken-Window-Theorie

Bezeichnet die These, dass Verwahrlosung und Kriminalität erfolgreich präventiv reduziert werden können, wenn bereits sehr geringfügiges (sozial-)schädliches Verhalten bekämpft wird; ein 1982 erstmals publizierter Ansatz aus der Kriminalprävention [Wilson-1996], vgl. auch Nulltoleranzstrategie („Zero Tolerance“). Übertragene Bedeutung: Wenn erst einmal irgendwo etwas geringfügig im Argen liegt (eine Scheibe eingeschlagen wurde), gesellen sich schnell weitere Probleme dazu (weitere Scheiben werden eingeschlagen).

BT, Bearbeitertag ⇒PT

Alternative Bezeichnung für Personentag (PT).

Build

Ein Build ist eine gewöhnlich unvollständige und vorübergehende, aber ausführbare Version eines in Entwicklung befindlichen Systems, die sich in der Entwicklungsumgebung oder gegebenenfalls in einer entwicklungsnahen Testumgebung befindet.

CAPM ⇒Certified Associate in Project Management**Cashflow**

Diese Kennzahl dient der Beurteilung des Innenfinanzierungspotenzials und der Ertragskraft des Unternehmens. Sie errechnet sich wie folgt: Cashflow = Jahresüberschuss bzw. -fehlbetrag - Erträge aus Verlustübernahme + abgeführte Gewinne + Abschreibungen und Wertberichtigungen auf Sachanlagen und immaterielle Anlagenwerte + Abschreibungen und Wertberichtigungen auf Finanzanlagen mit Ausnahme des Betrages, der in die Pauschalwertberichtigung zu Forderungen eingestellt ist, + Zuführung zu Pensionsrückstellungen.

Certified Associate in Project Management (CAPM)

Projektmanagementzertifikat des ⇒PMI.

Chance ⇒Risiko

Die Chance ist die Aussicht auf Erfolg. Eine Chance kann ausgedrückt werden durch die Wahrscheinlichkeit ihres Auftretens und eines Gewinns im Eintrittsfall.

CI ⇒Corporate Identity**CMMI**

Das Capability Maturity Model Integration (CMMI) ist ein Prozessmodell zur Beurteilung und Verbesserung der Reife von Produktentwicklungsprozessen in Organisationen. Es wurde vom Software Engineering Institute (SEI) der Carnegie Mellon University des US-Verteidigungsministeriums entwickelt.

Continuous Integration ⇒Fortlaufende Integration**Controlboard** ⇒Lenkungskreis**Conway's Law**

Bezeichnet das Prinzip, dass die Softwarearchitektur die verwendete Teamstruktur widerspiegelt.

Corporate Identity

Die Corporate Identity stellt die anzustrebende Einmaligkeit bzw. Persönlichkeit eines Unternehmens dar und hat den Zweck, das Unternehmen für seine relevanten Bezugsgruppen unverwechselbar zu machen. Sie wird explizit definiert und niedergeschrieben und trägt dadurch zu einem einheitlichen Auftreten, einheitlicher Haltung und gemeinsamen Werteverständnis bei. Mit den Mitteln der Corporate Communications, des Corporate Designs, der Corporate Culture sowie des Corporate Behaviours wird versucht, ein ganz bestimmtes Image betriebsintern und -extern aufzubauen.

Critical-Chain-Projektmanagement (CCPM)

⇒Kritische Kette

Managementkonzept der ⇒Theory of Constraints für Projektmanagement.

Crystal

Crystal ist eine Familie agiler Entwicklungsmethoden [Cockburn-2003], die in Abhängigkeit von der Projektgröße (Anzahl der Beteiligten) und Kritikalität (Ge-

fahr für Leben, Unternehmen, Geld, Komfort) variieren.

Domänenexperte ⇨ Fachexperte

Earned Value (EV)

Fortschrittswert, Ertragswert. Kennzahl: Fertigstellungsgrad * Projektbudget.

Earned-Value-Analyse (EVA)

Bei der EVA werden die geplanten Kosten, die tatsächlichen Kosten und die tatsächlich erbrachte Leistung gegenübergestellt. Vgl. ⇨289.

EBIT

Abkürzung für Earnings before Interests and Taxes. Die Ertragskennzahl EBIT wird berechnet aus dem Jahresüberschuss vor Zinsergebnis, Steuern und außerordentlichem Ergebnis. Sie macht eine von der individuellen Kapitalstruktur unabhängige Aussage zur operativen Ertragskraft eines Unternehmens. EBIT wird auch als betriebliches Ergebnis bezeichnet.

Eclipse-Way

Name des Vorgehensmodells im Eclipse-Projekt.

effektiv ⇨effizient

Die Effektivität bezeichnet die Wirksamkeit im Hinblick darauf, ob man das Richtige getan hat. Etwas ist ineffektiv, wenn das Ergebnis wenig wirksam ist.

effizient ⇨effektiv

Die Effizienz bezeichnet die Wirksamkeit im Hinblick darauf, wie man etwas macht. Etwas ist ineffizient, wenn man das Richtige beispielsweise umständlich ausführt.

Endgame ⇨Stabilisierungsiteration

Begriff aus dem Eclipse-Way. Bezeichnet einen Zeitabschnitt vor einem Release, der vorwiegend oder ausschließlich der Robustheit, Fehlerbehebung und Restrukturierung dient und keine neuen Funktionalitäten hinzufügt.

Engpass

Person oder Ressource, die eine Arbeit ausführt, die auf der ⇨kritischen Kette liegt.

Entwurfsmuster

Entwurfsmuster sind generalisierte und bewährte Lösungsansätze zu immer wiederkehrenden Entwurfsproblemen. Sie sind keine fertig codierten Lösungen, sie beschreiben lediglich den Lösungsweg bzw. die Lösungsidee.

Ergebnis, betriebswirtschaftliches ⇨EBIT

Dieser Begriff wird meist synonym für eine Vielzahl von Ergebniskennzahlen verwendet. Je nach Erfolgsrechnung fällt das Ergebnis jedoch unterschiedlich aus und heißt auch anders. So wird das Ergebnis der Gewinn- und Verlustrechnung der Handelsbilanz Jahresüberschuss bzw. -fehlbetrag oder Bilanzgewinn bzw. -verlust genannt, bei der Steuerbilanz spricht man vom steuerlichen Gewinn bzw. Verlust, und bei einer Einnahmen-Ausgabenrechnung heißt das Ergebnis Einzahlungs- bzw. Auszahlungsüberschuss.

Erledigungsgrad ⇨Fertigstellungsgrad

Extreme Programming (XP)

Ist eine agile Softwareentwicklungsmethode die von Kent Beck initiiert wurde [Beck-1999, Wolf-2005].

Fachabteilung

Eine Fachabteilung ist eine aus Sicht der Softwareentwicklung, der Betriebsorganisation oder eines Projektes andere Organisationseinheit. Eine Fachabteilung kann ein möglicher Auftraggeber für Software- oder Organisationsprojekte sein.

Fachexperte ⇨Domänenexperte

Ein Experte in einem bestimmten Fachgebiet (einer Domäne).

FDD ⇨ Feature Driven Development

Feature ⇨Planungsrelevantes Feature,

⇨Iterationsfeature, ⇨Releasefeature

Ein Feature ist eine Sammlung von (häufig noch zu konkretisierenden) Anforderungen. Vgl. Abgrenzung ⇨Feature Driven Development.

Feature Driven Development (FDD)

Im allgemeinen Sinne eine Entwicklungsmethode, die allgemein ⇨Features in den Mittelpunkt der Entwicklung stellt. Im spezielleren Sinne eine von Jeff DeLuca entwickelte phasen- bzw. prozess-

orientierte Softwareentwicklungsmethode, die feingranulare Features als Mittel zur Anforderungsanalyse und Planung verwendet. Wir verwenden den Begriff Feature in diesem Buch *nicht* in diesem speziellen Sinne.

Fehlerklasse

Einteilung von Fehlern nach Schwere der Fehlerwirkung aus Sicht des Auftraggebers, wobei die Einteilung abnahme- und vertragsrelevant sein kann.

Feldtest

Erprobung einer vorläufigen Softwareversion durch einen ausgewählten Anwenderkreis mit dem Ziel, Einflüsse auch aus unzureichend bekannten oder spezifizierten Einsatzumgebungen zu erkennen oder um die Marktakzeptanz zu prüfen.

Fertigstellungsgrad ⇒ Aufwandsfortschrittsindikator

Kennzahl: $\frac{\text{Verbrauchter Aufwand} \cdot 100}{\text{Verbrauchter Aufwand} + \text{Restaufwand}}$

Fertigstellungswertmethode ⇒ Earned-Value-Analyse

Fortlaufende Integration

Entwicklungsprinzip, bei dem die Entwickler ihre Änderungen so schnell und so oft wie möglich zu einer gemeinsamen Lösung zusammenführen.

GAF ⇒ Geschäftsanwendungsfall, ⇒ Kern-Geschäftsanwendungsfall, ⇒ Unterstützender Geschäftsanwendungsfall

Geschäftsanwendungsfall ⇒ Systemanwendungsfall

Ein Geschäftsanwendungsfall beschreibt einen geschäftlichen Ablauf, wird von einem geschäftlichen Ereignis ausgelöst und endet mit einem Ergebnis, das für den Unternehmenszweck und die Gewinnerzielungsabsicht direkt oder indirekt einen geschäftlichen Wert darstellt.

Geschäftsprozess

Ein Geschäftsprozess ist eine Zusammenfassung einer Menge fachlich verwandter Geschäftsanwendungsfälle. Dadurch bildet ein Geschäftsprozess gewöhnlich eine Zusammenfassung von organisatorisch evtl. verteilten, fachlich jedoch zusammenhängenden Aktivitäten,

die notwendig sind, um einen Geschäftsvorfall (z.B. einen konkreten Antrag) ergebnisorientiert zu bearbeiten. Die Aktivitäten eines Geschäftsprozesses stehen gewöhnlich in zeitlichen und logischen Abhängigkeiten zueinander.

Geschäftsprozessmodellierung

Bei der Geschäftsprozessmodellierung werden Geschäftsprozesse analysiert und in abstrakter Form zur Veranschaulichung bildhaft oder textuell dargestellt.

Gesellschaft für Projektmanagement

Die GPM Deutsche Gesellschaft für Projektmanagement e.V. ist der deutsche Fachverband für Projektmanagement. 1979 als gemeinnütziger Verein gegründet, besteht die satzungsgemäße Aufgabe in der Förderung des Projektmanagements, insbesondere der Aus- und Weiterbildung sowie der Forschung und Information auf diesem Gebiet. Die GPM ist der deutsche Vertreter in der IPMA.

Gewinnerzielungsabsicht

Womit beabsichtigt das Unternehmen bzw. die im Modellierungsfokus stehende Organisation Geld zu verdienen? Für gemeinnützige Organisationen ist die Gewinnerzielungsabsicht durch etwas Entsprechendes zu ersetzen, beispielsweise „Beitrag zum festgelegten Gemeinnutz“ o.Ä.

GP ⇒ Geschäftsprozess

GPM ⇒ Geschäftsprozessmodellierung

GPM ⇒ Gesellschaft für Projektmanagement

Heuristik

Eine Heuristik ist eine wahrscheinlichkeitsbehaftete Regel, also eine sog. Daumenregeln, die nicht immer, aber in vielen oder den meisten Fällen zutrifft.

Idealer Tag

Ein idealer (Entwickler-)Tag ist eine Einheit zur Aufwands- und Termenschätzung, die auf der Annahme basiert, dass eine Person sich einen Arbeitstag lang (8 Stunden) ohne Unterbrechung der Aufgabe widmen kann.

identifizieren

Identifizieren bedeutet, dass etwas eindeutig und unverwechselbar benennbar

und von allen anderen gleichartigen Dingen unterschieden werden kann.

IF ⇒ Iterationsfeature

Inkrement ⇒ Iteration

Ein Inkrement bezeichnet die Funktionalität einer Software, die am Ende einer bestimmten Iteration fertiggestellt wurde. Beim featurebasierten Planen wird das Inkrement der Iteration i durch die Menge der Iterationsfeatures definiert, die am Iterationsende fertig sein soll oder ist. Für die Herstellung des Iterationsfeatures sollte stets genau ein Team bzw. eine Person verantwortlich sein.

Innenfinanzierungspotenzial ⇒ Cashflow

Integration, Integrationsbuild

Verbinden von einzelnen Komponenten (Hardware oder Software) zu einem größeren Teilsystem oder zum Gesamtsystem.

Integrationstest

Test mit dem Ziel, Fehlerwirkung in Schnittstellen und im Zusammenspiel zwischen integrierten Komponenten zu finden.

International Project Management Association (IPMA)

Internationale Dachorganisation nationaler Projektmanagementorganisationen.

IPMA ⇒ International Project Management Association

ISO

Die International Organization for Standardization (ISO) ist eine 1947 gegründete Vereinigung der einzelnen nationalen Standardisierungsinstanzen aus über 140 Ländern.

Iteration

Eine Iteration ist ein in ähnlicher Weise wiederholter Zeitabschnitt innerhalb eines Entwicklungsprozesses. Der Gesamtprozess wird in eine Vielzahl kürzerer Zeitabschnitte untergliedert, in denen prinzipiell die gleichen elementaren Entwicklungsaktivitäten stattfinden.

Iterationsfeature ⇒ Feature

Ein Iterationsfeature repräsentiert eine geplante Ausbaustufe eines Produktfeatures. Die Zuordnung eines Iterationsfea-

tures zu genau einer Iteration i bedeutet, dass diese Ausbaustufe am Ende dieser Iteration abgeschlossen, d.h. dann vorzeigbar, sein soll.

Iterationskapazität

Als Iterationskapazität bezeichnet man die Anzahl an Personentagen (oder anderen Aufwandseinheiten), die innerhalb einer bestimmten Iteration vom Projekt netto geleistet werden kann. Hierzu rechnet man Nettokapazität eines Arbeitstages je Mitarbeiter * verfügbare Arbeitstage der betreffenden Iteration * Anzahl verfügbarer Mitarbeiter, z.B. $0,875$ PT je Arbeitstag und Mitarbeiter * 20 Arbeitstage * $3,5$ Mitarbeiter = $61,25$ PT.

Iterationsplan

Der Iterationsplan ist eine Matrix mit Produkt- bzw. Releasefeatures (Zeilen) und Iterationen (Spalten), in deren Kreuzungspunkten Iterationsfeatures dargestellt sind. Dieser Plan dient als Gesamtübersicht über den geplanten Funktionszuwachs eines Produktes (oder auch Teilen davon) und zeigt auf, welche Iterationsfeatures am Ende welcher Iteration abgeschlossen, d.h. vorzeigbar, sein sollen.

Kaizen

Kaizen (das; japanisch „Veränderung zum Besseren“) ist ein von Taiichi Ohno entwickeltes Managementkonzept [Ohno-2005].

Kern-Geschäftsanwendungsfall

⇒ Geschäftsanwendungsfall,

⇒ Unterstützender Geschäftsanwendungsfall

Ein Kern-Geschäftsanwendungsfall beschreibt einen geschäftlichen Ablauf, der von einem aktiven Geschäftspartner ausgelöst wird, ein Ergebnis von geschäftlichem Wert für mindestens einen aktiven Geschäftspartner erzeugt und direkt mit dem Unternehmenszweck und einer Gewinnerzielungsabsicht (bei gemeinnützigen Organisationen mit dem beabsichtigten Gemeinnutz) zusammenhängt. Eine zeitliche Kohärenz des Ablaufes wie bei einem Systemanwendungsfall ist nicht gefordert. In Kontrast zu den Kern-Geschäftsanwendungsfällen sind die unterstützenden Geschäftsan-

wendungsfälle zu sehen. In einem Handelsunternehmen ist der Verkauf von Waren gewöhnlich ein Kern-Geschäftsanwendungsfall, während beispielsweise die Finanzbuchhaltung oder der Wareneinkauf einen unterstützenden Geschäftsanwendungsfall darstellt.

Konsolidierung (auch Konsolidation)

Festigung, Sicherung. Sichert bzw. festigt etwas Bestehendes. Schließt häufig auch eine abschließende Überarbeitung und Bereinigung von offenen Punkten ein.

Kommunikationsplan

Ein Kommunikationsplan (auch Kommunikationsmanagementplan genannt) beschreibt die Informationsbedürfnisse der Stakeholder, die Informationsabsichten gegenüber den Stakeholdern und Regeln und Standards zum Kommunikations- und Dokumentationswesen.

Kostenfortschrittsindikator (KFI)

⇒ Aufwandsfortschrittsindikator
Kennzahl: $\text{Aktueller Kostenverbrauch} \cdot 100 / \text{Geplante Kosten}$.

Kostenperformance (KP)

Kennzahl: $\text{Realisierungsgrad} \cdot 100 / \text{Kostenfortschrittsindikator}$.

Kritische Kette

Die längste Abfolge inhaltlich oder personell (oder durch eine andere Ressource) voneinander abhängiger Arbeiten in einem Projekt. Die kritische Kette plus der Projektpuffer ergeben die Projektlaufzeit.

Kritischer Pfad ⇒ ressourcennivellierter kritischer Pfad

Die längste Abfolge inhaltlich voneinander abhängiger Arbeiten in einem Projekt.

Läufer

Der Läufer ist ein Mediator zwischen Entwicklungsteam und Projektmanagement, der in regelmäßigen Abständen den Statusbericht von den Entwicklern erfragt und diesen samt berichteten Problemen der Projektleitung vorträgt. Mehr ⇒ 349.

Leitbild eines Unternehmens ⇒ Leitziel

Mit dem Leitbild werden die Werte und Haltungen ausgedrückt, die das Unternehmen von anderen unterscheiden.

Leitziel eines Unternehmens ⇒ Leitbild

Das Leitziel des Unternehmens beschreibt mit einem Satz die Vision des Unternehmens. Es ist das an erster Stelle stehende und markanteste Ziel, das das Unternehmen hat. Gewöhnlich ist es mit einer Kennzahl, d.h. einer messbaren Größe, verknüpft.

Lenkungskreis, Lenkungsausschuss

Der Lenkungsausschuss ist ein regelmäßiges tagendes Managementgremium, dem vom Projekt berichtet wird und das die oberste Entscheidungsinstanz für das Projekt darstellt. Neben der Projektleitung gehören normalerweise Vertreter des Auftraggebers, des Auftragnehmers und der projektrelevanten Fachabteilungen dazu.

Load-Faktor

Bezeichnet im Extreme Programming den Faktor, mit dem unproduktive Arbeitszeit zur produktiven Zeit hinzuge-rechnet wird, wobei produktiv sich im engeren Sinne nur auf die reine Programmier-tätigkeit beschränkt.

LOC

Lines of Code

Makroschätzung ⇒ Mikroschätzung

Schätzverfahren, bei dem ein Gegenstand als Ganzes in Relation zu einem anderen bekannten Gegenstand geschätzt wird („B ist 1,5-mal so teuer wie A“).

Meilenstein

Ein Meilenstein definiert einen Termin, zu dem eine Menge von Ergebnissen in einer bestimmten Detaillierung und Vollständigkeit nachprüfbar und formal dokumentiert vorliegen muss. Liegen die Ergebnisse zum geplanten Termin nicht vor, wird der Meilensteintermin verschoben. Ein Meilenstein ist ein Hilfsmittel zur Planung und Überwachung eines Entwicklungsprozesses.

Methodik, Methode

Eine Methode ist eine Handlungsvorschrift, die beschreibt, wie ein Ziel bzw. Ergebnis unter gegebenen Bedingungen erreicht werden kann. Methode und Methodik sind synonym zu verwenden. Methodik klingt bedeutsamer.

Mikroschätzung ⇒ Makroschätzung

Schätzverfahren, bei dem ein Gegenstand so lange weiter in Einzelteile zerlegt wird, bis diese unmittelbar zu schätzen sind. Erfordert Wissen über die Struktur des zu schätzenden Gegenstandes.

Mission ⇒ Leitbild**MT, Manntag** ⇒ PT, ⇒ BT

Unübliche Alternativbezeichnung für Personentag (PT).

Object Management Group (OMG)

Die Object Management Group ist ein Industriekonsortium u.a. zur Entwicklung, Vereinheitlichung und Standardisierung von Informationstechnologien. Siehe <http://www.omg.org>.

OEP ⇒ oose Engineering Process**OMG** ⇒ Object Management Group**oose Engineering Process (OEP)**

Der OEP ist ein Vorgehensmodell und Leitfaden zur agilen objektorientierten Softwareentwicklung, basierend auf UML und Unified Process und einer Abbildungskonvention auf das V-Modell XT. Siehe <http://www.oose.de/oep>.

Operativ, Operatives Ziel

Zur Erreichung operativer Ziele werden vorhandene Potenziale und Möglichkeiten ausgenutzt, um operative Kennzahlen wie Umsatz, Gewinn u.Ä. zu steigern. Operative Maßnahmen finanzieren sich aus vorhandenen Ressourcen und werden durchgeführt unter der Annahme, dass sich die Einnahme-Kosten-Relation verbessert.

Organigramm ⇒ Organisationsplan**Organisation** ⇒ Organisationseinheit

Eine Organisation besteht aus Organisationseinheiten und strukturiert diese zielorientiert. Sie kann aufgabenbezogen als Ablauf- oder strukturbezogen als Auf-

bauorganisation in Form eines bildhaften Organisationsplanes dargestellt werden.

Organisationseinheit ⇒ Organisation

Eine Organisationseinheit ist Teil einer Organisation oder einer anderen Organisationseinheit und erfüllt einen bestimmten Zweck. Sie definiert sich durch einen abgegrenzten Aufgabenbereich, der dem Zweck der Organisation dient. In der Regel wird ein Mitarbeiter einer Organisationseinheit zugeordnet.

Organisationsplan ⇒ Organisation, ⇒ Organigramm

Ein Organisationsplan ist die grafische Darstellung struktureller Beziehungen zwischen Organisationseinheiten eines betrachteten Unternehmens.

Pareto-Prinzip

Vilfredo Pareto (1848–1923) entdeckte bei einer Untersuchung des italienischen Volksvermögens, dass 20 Prozent der Familien 80 Prozent des Vermögens besitzen. Dahinter steht eine stetige Wahrscheinlichkeitsverteilung, die auch auf viele andere Sachverhalte übertragbar ist. Beispielsweise dass mit 20 Prozent des Aufwandes 80 Prozent der Ergebnisse entwickelt werden können. Auch bekannt als 80-20-Regel.

Parkinson's Law

Bezeichnet die Beobachtung, dass jede Arbeit stets die dafür vorgesehene Zeit vollständig ausfüllt.

Personentag ⇒ PT**PF** ⇒ Produktfeature**PgMP** ⇒ Program Management Professional**Planungsrelevantes Feature** ⇒ Feature

Ein planungsrelevantes Feature ist eine Eigenschaft, die a) beschreibt, was das zu planende Produkt (normalerweise die zu erstellende Software) leistet, b) üblicherweise eine Funktionalität beschreibt, die idealerweise automatisiert testbar ist (wenn nicht, beschreibt es zumindest eine nicht funktionale Anforderung, die nachweisbar ist), c) meist einen kaufentscheidenden Wert für den Auftraggeber hat (der im Zweifel darüber befinden darf), d) zusammengehörige Anforderungen zu einem sinnvollen Ganzen ag-

gregiert (z.B. eine Menge mehrerer Anwendungsfälle, die gemeinsam alle Lebenszyklen eines fachlichen Objektes abdecken) und e) planbar ist, d.h., es sind konkrete Aufgaben ableitbar.

PMI ⇒ Project Management Institute

PMP ⇒ Project Management Professional

PRINCE2

PRINCE (PROjects IN CONTROLLED ENVIRONMENTs) ist eine Projektmanagementmethode für Organisation, Management und Steuerung von Projekten. Sie wurde ursprünglich 1989 von der britischen Central Computer and Telecommunications Agency (CCTA) als Regierungsstandard für Projektmanagement im Bereich der Informationstechnik (IT) entwickelt, wurde jedoch bald regelmäßig auch außerhalb von reinen IT-Umgebungen angewendet. PRINCE2 wurde 1996 als allgemeine Projektmanagementmethode veröffentlicht. PRINCE2 ist zunehmend populärer geworden und ist nun der De-facto-Standard für Projektmanagement in Großbritannien. Seine Anwendung ist mittlerweile in mehr als 50 anderen Ländern verbreitet. Die aktuelle Version wurde 2005 vom Office of Government Commerce (OGC) veröffentlicht, das mittlerweile die CCTA abgelöst hat (aus Wikipedia).

Prinzip

Ein *axiomatisches Prinzip* ist eine Gesetzmäßigkeit, Regel oder Richtlinie, die anderen Gesetzen übergeordnet ist. Ein *systematisches Prinzip* ist die Beschreibung einer Konstellation spezifischer Faktoren, die einen bestimmte Effekt hervorrufen.

Privates Build ⇒ Build

Build in der lokalen Entwicklungsumgebung eines Entwicklers, das noch nicht mit dem Code aller anderen Entwickler synchronisiert ist.

Produktfeature ⇒ Feature

Ein Produktfeature repräsentiert ein Leistungsmerkmal eines zu erstellenden Produktes (z.B. einer Software) und beschreibt im Normalfall eine Funktionalität, die idealerweise automatisiert testbar ist. Dabei handelt es sich in der Regel um

eine Gruppe mehrerer zusammengehöriger Anwendungsfälle bzw. anderer Anforderungen, die gemeinsam alle Lebenszyklen eines fachlichen Objektes abdeckt und für den Auftraggeber einen kaufentscheidenden Wert hat.

Produktivumgebung

Beim Anwender oder Betreiber im Einsatz befindliche Hard- und Softwareumgebung, auf der das entwickelte System betrieben wird.

Produktteam ⇒ Releaseorientierte Projektorganisation

Organisationseinheit innerhalb eines Projektes, die speziell für die Entwicklung eines Releases eingerichtet wird.

Program Management Professional (PgMP)

Projektmanagementzertifikat des ⇒ PMI.

Project Management Institute (PMI)

Größte internationale Projektmanagementorganisation.

Project Management Professional (PMP)

International am weitesten verbreiteter und anerkannter Befähigungsnachweis im Projektmanagement. Zertifizierung erfolgt durch PMI.

Projekt

Ein Projekt ist ein einmaliges Vorhaben, es ist zeitlich begrenzt, es verfügt über begrenzte finanzielle, personelle und sonstige Ressourcen, es hat klare Ziele, es ist ein komplexes Vorhaben, es macht die Zusammenarbeiten von Menschen verschiedener Fachgebiete notwendig, es hat neuartige und unbekannte Probleme zu lösen, es steht unter einem besonderen Risiko und erzeugt einen besonderen Druck auf die Beteiligten.

Projektakte

Aggregation verschiedener Planungs-, Steuerungs- und Controlling-Dokumente eines Projektes.

Projektpuffer

Ein Zeitpuffer am Ende des Projektes, der dafür sorgt, dass der versprochene Liefertermin des Projektes trotz Störungen und ungeplanter Ereignisse eingehalten werden kann.

Projektstrukturplan

Ein Projektstrukturplan strukturiert die Elemente eines Projektes durch hierarchische Gliederung. Er dient dazu, den Inhalt eines Projektes vom Groben ins Feine zu zerlegen, um für verschiedene Zielgruppen unterschiedlich granulare Überblicke über den Projekteinhalt und die damit verbundenen Arbeitspakete zu schaffen.

Prozesskette ⇒ Geschäftsprozess

PSP ⇒ Projektstrukturplan

PT, Personentag ⇒ BT, MT

Aufwandseinheit, die den Verbrauch von einem Arbeitstag (üblicherweise 8 Arbeitsstunden) angibt.

Qualität

Gesamtheit der Merkmale und Merkmalswerte eines Produktes oder einer Dienstleistung, die sich auf deren Eigenschaft beziehen, festgelegte oder vorausgesetzte Erfordernisse zu erfüllen.

Qualitätsmerkmal

Satz von Eigenschaften eines Produktes, anhand dessen seine Qualität beschrieben und beurteilt wird.

Realisierungsgrad

Kennzahl: $(\text{Geplanter Aufwand} - \text{Restaufwand}) / \text{Geplanter Aufwand}$.

Redundanz

Lat. überreichlich, überflüssig. Bezeichnet gewöhnlich Dinge, die ohne Verlust entfallen können. Als redundanzfrei wird etwas bezeichnet, wenn jeder Sachverhalt (z.B. eine Anwendungsfallbeschreibung) nur genau einmal (im gesamten Anwendungsfallmodell) vorhanden ist.

Refactoring (Restrukturierung)

Bezeichnet das (verbessernde) Umstrukturieren und Ändern des Programmcodes unter Beibehaltung seiner Funktionalität, ohne dass in diesem Arbeitsgang neue Funktionalität hinzugefügt wird.

Regressionstest

Erneuter Test einer bereits getesteten, aber danach modifizierten Funktionalität mit dem Ziel nachzuweisen, dass durch die vorgenommenen Änderungen keine neuen Fehler auftreten.

Reichsbedenkenträger

Eine Person, die in allem Probleme sieht („das geht nicht“) bzw. Bedenken hat („ich weiß nicht so recht“) und dies auch ständig äußert. Der Reichsbedenkenträger hat erstaunliche Fähigkeiten und kann sehr gut neue Ideen torpedieren, indem er blitzschnell haufenweise phantasievoller Was-ist-wenn-Szenarien aufstellt und alle Beteiligten damit in tiefe Verunsicherung stürzt. Auf der anderen Seite ist er aber auch extrem gründlich und kann daher sehr gut eingesetzt werden, um z.B. Spezifikationslücken aufzuspüren, Qualität zu sichern, Tests durchzuführen, aber auch um Projektrisiken zu finden.

rekursiv

Selbstbezüglich, zurückgehend bis zu einem bekannten Punkt.

Release ⇒ Build

Ein Release ist ein Build, das dahingehend weiterentwickelt wurde, dass es an den Kunden ausgeliefert werden bzw. in seiner Zielumgebung laufen *kann*, d.h., es sind die notwendigen Freigabeverfahren zu durchlaufen. Ein Release ist immer auch ein ⇒ Meilenstein.

Releasefeature

Ein Releasefeature ist eine Ausbaustufe eines Produktfeatures für genau ein Release und umfasst alle Iterationsfeatures des Produktfeatures, die bis zum Releasetermin erstellt werden sollen.

Releaseorientierte Projektorganisation

Konsequente Zergliederung eines Projektes in einzelne organisatorisch voneinander vollständig unabhängige Teilprojekte (Produktteams), die fachlich und inhaltlich aufeinander aufbauend eine gemeinsame Software entwickeln, wobei jedes Produktteam genau ein Release entwickelt und nur für die Entwicklung dieses einen Release existiert.

Releaseplan

Der Releaseplan stellt die Dauer der Phasen, die Iterationen sowie die Meilensteine eines Projektes auf einer Zeitachse dar, sodass für das Gesamtprojekt und/oder für die einzelnen Releases eines Projektes erkennbar wird, welche

Phasen sich über welche Iterationen erstrecken und wann welche Meilensteine zu erreichen sind.

Releaseteam ⇒Produktteam

Temporäre Suborganisation innerhalb des Projektes, das eine Alpha- oder Beta-Version zu einem Release weiterentwickelt, das heißt die Version zur Inbetriebnahme in der Zielumgebung vorbereitet und bereitstellt.

Ressourcennivellierter kritischer Pfad

⇒Kritischer Pfad

Kritischer Pfad, der die verfügbaren Ressourcen (Personal) berücksichtigt, ohne den kritischen Pfad dadurch selbst zu verändern.

Restrukturierung ⇒Refactoring

Review

Manuelle Prüfmethode, mit der ein Prüfobjekt qualitativ oder quantitativ bewertet wird.

Risiko ⇒Chance

Ein Risiko ist ein möglicher Verlust. Ein Risiko kann ausgedrückt werden durch die Wahrscheinlichkeit seines Auftretens und einer Schadenhöhe im Eintrittsfall. Ein Risiko ist eine negative Chance.

Robustheit

Verhalten gegenüber Fehlbenutzung, Fehlprogrammierung, Hardwareausfall u.Ä. inkl. Fehlerbehandlung und Weiter- oder Wiederanlaufverhalten.

ROCE ⇒ROI

Gesamtkapitalrentabilität (Return on Capital Employed)

ROI ⇒ROCE

Abkürzung für Return on Investment. Der Return on Investment ist das, was aus dem Investment „zurückkehren“ soll. Er drückt somit das Gewinnziel aus. Der Gewinn wird auf das investierte, betriebsnotwendige Vermögen bezogen.

Rolle

Wird hier als Allgemeinbegriff verwendet. Eine Rolle drückt eine Aufgabe oder Funktion aus, die von jemandem oder etwas übernommen wird.

SAF ⇒Systemanwendungsfall

Schätzmethode

Unter Schätzmethode verstehen wir mathematisch bzw. statistisch abgesicherte Verfahren zur Abschätzung umfangreicher Projekte. Wir klassifizieren Schätzmethode in Analogiemethoden, Relationsmethoden, Gewichtungsmethoden, Multiplikatormethoden, Prozentsatzmethoden und parametrische Schätzmethode.

Scrum

Scrum ist eine agile Softwareentwicklungsmethode, die maßgeblich von Ken Schwaber initiiert wurde [Schwaber-2004].

Smoke-Build ⇒Smoke-Test

Build, das mit dem Ziel erstellt wird, einen Smoke-Test durchzuführen.

Smoke-Test

Normalerweise automatisierter Test, der möglichst alle Hauptfunktionen eines Testobjektes auslöst, ohne die Ausgaben des Testobjektes mit vorgegebenen Sollergebnissen zu vergleichen, mit dem Ziel, die grundsätzliche Testbarkeit und Robustheit (z.B. Absturzfreiheit) zu prüfen. Begriff kommt aus der Elektrotechnik und bezeichnet dort den ersten Test, bei dem eine elektrische Schaltung unter Strom gesetzt wird und dabei geguckt wird, ob irgendetwas anfängt zu rauchen.

SPICE (Software Process Improvement and Capability Determination)

SPICE oder ISO/IEC 15504 ist ein internationaler Standard zur Durchführung von Bewertungen (Assessments) von Unternehmensprozessen mit Schwerpunkt auf der Softwareentwicklung. Er wurde 1998 als Technischer Report (TR) in einer Vorversion verabschiedet und im März 2006 durch den aktuellen internationalen Standard (IS) ersetzt. Dieser besteht zurzeit (2007) aus fünf Teilen, von denen jedoch nur der 2. Teil normativen Charakter hat; die anderen Teile dienen lediglich als Beispiele, Erläuterung und Information (aus Wikipedia).

Spike

Ein Spike ist eine Art explorativer (Wegwerf-)Prototyp.

Sprint ⇒ Iteration

Iteration in Scrum

Stab, Stabsfunktion

Element einer Projektorganisation, hat im Gegensatz zu anderen Funktionen keine oder allerhöchstens fachliche Weisungsbefugnis gegenüber anderen Personen oder Bereichen.

Stabilisierungsiteration ⇒ Endgame

Bezeichnet einen Zeitabschnitt, der vorwiegend oder ausschließlich der Robustheit, Fehlerbehebung und Restrukturierung dient und keine neuen Funktionalitäten hinzufügt. Im Gegensatz zum Endgame wird die Stabilisierungsiteration als Pseudo-Iteration aufgefasst, die möglichst geplant (⇒ Releaseplan) und selten eingesetzt werden soll, prinzipiell aber jederzeit zwischen den Iterationen stattfinden kann.

Staffelläuferprinzip

Arbeitsprinzip, bei dem die Arbeit so schnell wie möglich erledigt wird, sobald sie möglich ist, und bei dem der Durchführende sich vorbereitet und keine andere Arbeit macht, wenn die Arbeit noch nicht begonnen werden kann. Führt zu wartenden Leerlaufzeiten und steht somit im Gegensatz zum Prinzip lokaler Effizienz, bei dem jeder stets beschäftigt sein soll.

Stakeholder

Person, die in irgendeiner Form „teil hat“ bzw. betroffen ist (engl. für Teilhaber). Mögliche deutsche Bezeichnungen: Interessenhalter, Anforderungsbeitragender oder Projektbetroffener. Mit Stakeholder werden also die verschiedenen an einem Unternehmen, einer Organisation oder einem Projekt interessierten Gruppen wie Mitarbeiter, Lieferanten, Kapitalgeber, Gesellschaft, Gesetzgeber bezeichnet.

Steering Committee ⇒ Lenkungskreis**Steuerungsausschuss, Steuerungskreis**

⇒ Lenkungskreis

Strategiekarte ⇒ Balanced Scorecard**Strategisch, Strategisches Ziel**

Eine Strategie ist ein Vorgehensplan zur Erreichung eines Ziels, bei dem von

vornherein versucht wird, diejenigen Faktoren einzukalkulieren, die in die geplanten Handlungen hineinspielen könnten. Ein strategisches Ziel besteht gewöhnlich im Erreichen einer zukünftig besseren Ausgangsposition. Es dient normalerweise nicht zur unmittelbaren Verbesserung bestimmter Kennzahlen, sondern mit ihm sollen die Möglichkeiten, Potenziale und allgemeinen Rahmenbedingungen verbessert werden. Aufgrund dieser verbesserten Möglichkeiten können dann später wiederum (noch besser) operative Ziele verfolgt werden. Strategische Ziele sind daher meistens mittel- oder langfristig ausgerichtet.

Stresstest

Prüfung des Systemverhaltens bei Überlastung.

Studentensyndrom

Arbeitsverhalten, bei dem der Beginn einer Arbeit so lange hinaus gezögert wird, bis alle zuvor eingeplanten Puffer verbraucht sind.

Synergie

Synergie ist der Effekt, dass bei optimaler Kombination von Einzelelementen die sich daraus ergebende Gesamtheit mehr ist als die Summe der Einzelteile. Insbesondere in der strategischen Planung ist die Diskussion von Synergien durch die optimale Kombination von Geschäftsfeldern von Bedeutung.

Systemakteur

Ein Systemakteur ist eine Person oder ein externes System, das mit einem technischen System (Hard- oder Software) interagiert. Dies können also die Benutzer oder andere externe Systeme, z.B. SAP, sein. Außerdem können auch Ereignisse, wie Zeitereignisse (z.B. Monatswechsel), als Systemakteure aufgefasst werden.

Systemanwendungsfall

Ein Systemanwendungsfall ist ein Anwendungsfall, der speziell das für die außen stehenden Akteure (Benutzer) wahrnehmbare Verhalten eines Systems beschreibt.

Teilabnahme ⇒ Vorabnahme

Abnahme auf Basis eines Releases, das nur eine begrenzte Menge von Anforderungen oder Features umfasst. Beispielsweise wenn ein weiteres umfassenderes Release später geliefert wird.

Terminperformance (TP)

Kennzahl: Realisierungsgrad * 100 / Aufwandsfortschrittsindikator

Testautomatisierung

Einsatz von Softwarewerkzeugen und Programmierung von Testfällen mit dem Ziel, Testfälle rechnergestützt wiederholt ausführen und automatisch prüfen zu können.

Testgetriebenes Design

Testgetriebenes Design bedeutet, für eine Anforderung noch vor deren Realisierung die notwendigen Tests herzustellen. Da Tests stets gegen eine Schnittstelle gerichtet sind, sind also beim Herstellen der Tests bereits Annahmen über diese Schnittstelle zu treffen. Da eine Schnittstelle andererseits Teil des Designs ist, ist das Design sozusagen Abfallprodukt testgetriebenen Vorgehens und der daran anschließenden Restrukturierungen.

Timebox ⇒ Meilenstein

Eine Timebox definiert einen unverrückbaren Zeitrahmen, an dessen Ende eine Menge von Ergebnissen in einer bestimmten Detaillierung und Vollständigkeit nachprüfbar und formal dokumentiert vorliegen soll. Liegen die Ergebnisse nicht wie geplant vor, werden die offenen Teile in eine nachfolgende Timebox verschoben. Hierzu werden zum geplanten Endtermin die tatsächlich erreichten Ergebnisse bestimmt. Eine Timebox ist ein Hilfsmittel zur Planung und Überwachung eines Entwicklungsprozesses.

Theory of Constraints

Von Eliyahu M. Goldratt entwickelte Unternehmensphilosophie, die Logistik, Leistungskennzahlen und strenger Logik betreffend.

ToC ⇒ Theory of Constraints**Transaktion**

Eine Transaktion ist ein Vorgang, der entweder vollständig oder gar nicht voll-

zogen werden kann. Wenn der Vorgang mittendrin aufhört oder abgebrochen wird, ist dies im Ergebnis so, als hätte der Vorgang nie stattgefunden, der Zustand ist derselbe wie vorher. Eventuelle Zwischenergebnisse gehen im Falle eines vorzeitigen Abbruchs verloren. Unter Umständen sind hierfür auch inverse Operationen notwendig, um den ursprünglichen Zustand wieder herzustellen.

Transition

Eine Transition ist innerhalb eines Zustandsdiagramms ein Zustandsübergang, häufig ausgelöst durch ein Ereignis. Der Begriff Transition wird auch als Übergang von einer Aktivität zur nächsten innerhalb von Aktivitätsdiagrammen verwendet.

Truck-Faktor

Der Truck-Faktor ist ein Synonym für das Risiko, dass ein Projektmitglied von einem Moment zum nächsten komplett ausfällt (so als wäre er vom Truck überfahren worden).

UC ⇒ Anwendungsfall, ⇒ Use Case**UML**

UML ist die Abkürzung für Unified Modeling Language. Die UML ist eine von der Object Management Group (OMG) standardisierte Notation und Semantik zur Visualisierung, Konstruktion und Dokumentation von Modellen für die objektorientierte Softwareentwicklung.

Unified Modeling Language ⇒ UML**Unified Process**

Der Unified Process ist ein Vorgehensmodell zur objektorientierten Softwareentwicklung auf Basis der UML und wird in dem Buch *The Unified Software Development Process* beschrieben [Jacobson-1999].

Unternehmensidentität ⇒ Corporate Identity**Unterstützender Geschäftsanwendungsfall**

⇒ Geschäftsanwendungsfall, ⇒ Kern-Geschäftsanwendungsfall

Ein unterstützender Geschäftsanwendungsfall beschreibt einen geschäftlichen Ablauf, der für den Unternehmenszweck und die Gewinnerzielungsabsicht keinen

oder nur indirekt einen Wert darstellt. Im Kontrast hierzu sind die Kern-Geschäftsanwendungsfälle zu sehen. In einem Handelsunternehmen ist die Finanzbuchhaltung oder der Wareneinkauf gewöhnlich ein unterstützender Geschäftsanwendungsfall, während beispielsweise der Verkauf von Waren hier ein Kern-Geschäftsanwendungsfall wäre.

Use Case ⇨ Anwendungsfall

Verantwortungsdiffusion

Bezeichnet das Prinzip, dass Menschen sich umso weniger verantwortlich fühlen, je größer die Gruppe ist, in der sie sich befinden.

Vorabnahme ⇨ Teilabnahme

Abnahme auf Basis eines Builds (oder Releases), die nur eine begrenzte Menge von Anforderungen oder Features umfasst und die vorbehaltlich der späteren Reproduzierbarkeit im Rahmen einer Abnahme oder Teilabnahme gilt.

Vorbedingung ⇨ Nachbedingung

Eine Vorbedingung beschreibt einen Zustand, der vor dem Ablauf einer Tätigkeit, Aktivität, Operation o.Ä. gegeben sein muss.

WBS ⇨ Work Breakdown Structure, ⇨ Projektstrukturplan

Wert

Mit dem Begriff Wert (wenn er nicht offensichtlich kaufmännisch gemeint ist) bezeichnen wir Werte im sozialwissenschaftlichen Sinne, d.h. Vorstellungen einer sozialen Gruppe oder einzelner Menschen über wichtige oder wünschenswerte Eigenschaften (oftmals Zu-

stände) von Dingen, Ideen und Beziehungen. Einen Konsens über den Begriff des Wertes gibt es allerdings weder in der Philosophie, Psychologie noch in anderen Sozialwissenschaften.

Wiki

Ein Wiki ist eine von Ward Cunningham initiierte Open-Source-Technologie zum kooperativen Erstellen von Internet- und Intranetseiten, d.h., jeder Benutzer darf jederzeit alles lesen, ändern und neue Seiten hinzufügen. Die Benutzung ist extrem einfach. *Wiki-Wiki* kommt aus dem Hawaiianischen und heißt *schnell*. Auch die Taxis heißen dort so.

Work Breakdown Structure ⇨ Projektstrukturplan

Workflow

Ein Workflow ist die computergestützte Automatisierung und Unterstützung eines Geschäftsprozesses oder eines Teils davon.

XP ⇨ Extreme Programming

Zulieferkette, Zulieferleistung ⇨ Kritische Kette

Aufgabe oder Folge von Aufgaben, die nicht Bestandteil der kritischen Kette sind, jedoch in diese einmünden.

Zulieferpuffer ⇨ Zulieferleistung

Puffer, der eingeplant wird, damit er die Verzögerung einer Zulieferleistung gegenüber den nachfolgenden Aufgaben auf der kritischen Kette abfängt. Im Critical-Chain-Projektmanagement wird dafür 1/3 der Bearbeitungszeit der Zulieferleistung angesetzt.

10.2 Literatur

Die **fett** hervorgehobenen Titel empfehlen wir zur Vertiefung oder Ergänzung.

[Beck-1999]

K. Beck: *Extreme Programming explained*, Addison Wesley Longman, 1999.

[Beck-2001]

K. Beck, M. Fowler: *Extreme Programming planen*, Addison Wesley, 2001.

[Beck-2003]

K. Beck: *Extreme Programming, Das Manifest*, 2. Auflage, Addison Wesley Longman, 2003.

[Beedle-2002]

M. Beedle: *Agile Software Development with Scrum*, Prentice Hall, 2002.

[Berne-1970]

E. Berne: *Spiele der Erwachsenen – Psychologie der menschlichen Beziehungen*, Rowohlt Taschenbuch, 1970.

[Boehm-1981]

B. Boehm: *Software Engineering Economics*, Prentice Hall, 1981.

[Boehm-1985]

B. Boehm: *A Spiral Model of Software Development and Enhancement*. Proceedings International Workshop on Software Process and Software Environments. 3/1985.

[Boehm-2000]

B. W. Boehm, C. Abts, A. W. Brown: *Software Cost Estimation with Cocomo II*. Prentice Hall, 2000.

[Brooks-1995]

F. Brooks: *The Mythical Man-Month*, Addison-Wesley, 1995.

[Bundschuh-2004]

M. Bundschuh, A. Fabry: *Aufwandschätzung von IT-Projekten*, MITP-Verlag, 2004.

[Caroll-2005]

E. R. Caroll: *Estimating Software based on Use Case Points*, OOPSLA 2005

[Cockburn-2003]

A. Cockburn: *Agile Software-Entwicklung*, MITP-Verlag, 2003.

[Cockburn-2004]

A. Cockburn: *Crystal Clear*, Addison-Wesley, 2004.

[Cohn-2005]

M. Cohn: *Agile Estimating and Planning*, Prentice Hall, 2005.

[Darley-1968]

J. M. Darley, B. Latané: *Bystander intervention in emergencies: Diffusion of responsibility*. In: Journal of Personality and Social Psychology, 8, 377-383, 1968.

[DeMarco-1997]

T. DeMarco: *Warum ist Software so teuer?*, Hanser, 1997.

[DeMarco-1998]

T. DeMarco: *Der Termin, ein Roman über Projektmanagement*, Hanser, 1998.

[DeMarco-1999]

T. DeMarco: *Wien wartet auf Dich! Der Faktor Mensch im DV-Management*, 2. Auflage, Hanser, 1999.

[DeMarco-2001]

T. DeMarco: *Spielräume, Projektmanagement jenseits von Burn-out, Stress und Effizienz-wahn*, Hanser, 2003.

[DeMarco-2003]

T. DeMarco: *Bärentango – mit Risikomanagement Projekte zum Erfolg führen*, Hanser, 2003.

[Derby-2006]

E. Derby, D. Larsen: *Agile Retrospektives, Making Good Teams Great*, Pragmatic Bookshelf, 2006.

[Eckstein-2004]

J. Eckstein: *Agile Softwareentwicklung im Großen*, dpunkt.verlag, 2006.

[Fisher-2004]

R. Fisher, W. Ury, B. Patton: *Das Harvard-Konzept – der Klassiker der Verhandlungstechnik*, 22. Auflage, Campus, 2004.

[Francis-1996]

D. Francis, D. Young: *Mehr Erfolg im Team*, Windmühle, 1996.

[Friedag-2002]

H. R. Friedag, W. Schmidt: *Balanced Scorecard*, Haufe, 2002.

[Frohnhoff-2006]

S. Frohnhoff et al.: *Use Case Points in der industriellen Praxis*, sd&m, 2006.

[Gamma-2007]

E. Gamma: *The Eclipse-Way*, Vortrag auf der Konferenz SE 2007, Hamburg, 2007.

- [Goldratt-2001]
E. Goldratt, J. Cox: *Das Ziel – Ein Roman über Prozessoptimierung*. Campus, 2001.
- [Goldratt-2002]
E. Goldratt: *Critical Chain, A Business Novel*; deutsch: *Die kritische Kette*, Campus, 2002.
- [Harris-1975]
T. Harris: *Ich bin O.K., Du bist O.K. – Eine Einführung in die Transaktionsanalyse*, Rowohlt Taschenbuch, 1975.
- [Highsmith-2004]
J. Highsmith: *Agile Project Management*, Addison-Wesley, 2004.
- [Himmelreich-2006]
J. Himmelreich: *Agile Softwareentwicklung nach Winston Royce*, in: B. Oestereich (Hrsg.): *Agiles Projektmanagement, Beiträge zur Konferenz interPM*, dpunkt.verlag, 2006.
- [Jacobson-1999]
I. Jacobson et al.: *The Unified Software Development Process*, Addison-Wesley, 1999.
- [John-2004]
F. John, G. Peters-Kühlinger: *Mit Druck richtig umgehen*, Haufe, 2004.
- [Jones-1996]
C. Jones: *Assuring Productivity and Quality Applied Software Measurement*, McGraw-Hill, 1996.
- [Jung-1990]
C. G. Jung: *Typologie*, dtv, 1990.
- [Kaplan-2001]
R. Kaplan, D. Norton: *Die strategiefokussierte Organisation. Führen mit der Balanced Scorecard*. Schäffer-Poeschel Verlag, 2001.
- [Kellner-1995]
H. Kellner: *Konferenzen, Sitzungen, Workshops effizient gestalten – Nicht nur zusammensitzen*, Hanser, 1995.
- [Kellner-1999]
H. Kellner: *Konflikte verstehen, verhindern, lösen – Konfliktmanagement für Führungskräfte*, Hanser, 1999.
- [Kerth-2001]
N. L. Kerth, *Project Retrospectives, A Handbook for Team Reviews*, Dorset House, 2001.
- [Kohl-2000]
H. Kohl: *Mein Tagebuch 1998 – 2000*, Droemer Knauer, 2000.
- [König-2002]
E. König, G. Vollmer: *Systemisches Coaching, Handbuch für Führungskräfte, Berater und Trainer*, Beltz, 2002.
- [Krasemann-2001]
H. Krasemann: *Messen und Schätzen*. In: B. Oestereich (Hrsg.): *Erfolgreich mit Objektorientierung*, Oldenbourg, 2001.
- [Krasemann-2005]
H. Krasemann: *Makro-Schätzen von Projekten*, Objekt-Spektrum 4/2005.
- [Larman-2004a]
C. Larman: *Agile and iterative Development*, Addison-Wesley, 2004.
- [Larman-2004b]
C. Larman: *The historical accident of waterfall validity*, in: *Agile and iterative development*, 2004.
- [Lipp-1996]
U. Lipp, H. Will: *Das große Workshop-Buch – Konzeption, Inszenierung und Moderation von Klausuren, Besprechungen und Seminaren*, Beltz, 1996.
- [Lippitt-1999]
G. Lippitt, R. Lippitt: *Beratung als Prozess*, Rosenberger Fachverlag, 1999.
- [Lörz-2007]
H. Lörz, U. Techt: *Critical Chain, Beschleunigen Sie Ihr Projektmanagement*, Haufe, 2007.
- [Madden-1984]
W. A. Madden, K. Y. Rone: *Design, Development, Integration: Space Shuttle Primary Flight Software System*, Communications of the ACM 27 (9), 914-925, 11/1984.
- [Manns-2005]
M. L. Manns, L. Rising: *Fearless Change*, Addison-Wesley, 2005.
- [Maro-2002]
F. Maro: *Mitreibende Meetings und gelungene Events*, Metropolitan-Verlag, 2002.
- [Mayrshofer-2006]
D. Mayrshofer, H. Kröger: *Prozesskompetenz in der Projektarbeit*, 3. Auflage, Windmühle, 2006.
- [Mohaghegi-2005]
A. Mohaghegi et al.: *Effort Estimation of Use Cases for Incremental Large-Scale Software Development*, ICSE 2005.
- [OEP-2007]**
B. Oestereich, C. Schröder, M. Klink, G. Zockoll: OEP – oose Engineering Process, Vorgehensleitfaden für agile Softwareprojekte, dpunkt.verlag, 2006.
- [Oestereich-2001]
B. Oestereich et al.: *Erfolgreich mit Objektorientierung – Vorgehensmodelle und Managementpraktiken für die objektorientierte Softwareentwicklung*. 2. Aufl., Oldenbourg Wissenschaftsverlag, 2001.
- [Oestereich-2004]
B. Oestereich et al.: *Objektorientierte Geschäftsprozessmodellierung mit der UML*, 1. korr. Nachdruck, dpunkt.verlag, 2004.

- [Oestereich-2005]
B. Oestereich: *Beweglich bleiben: Möglichkeiten und Grenzen des iterativen Vorgehens*, Objekt-Spektrum 1/2005.
- [Oestereich-2006]
B. Oestereich: *Analyse und Design mit der UML 2*, 8. Auflage, Oldenbourg Wissenschaftsverlag, 2006.
- [Ohno-2005]
T. Ohno: *Das Toyota-Produktionssystem*, Campus, 2005.
- [PMBok-2004]
Project Management Institute (PMI): *A Guide to the Project Management Body of Knowledge*, 3. Auflage, 2004.
- [Putnam]
Putnam, D.: *How many software can be developed in a year?*, www.qsm.com
- [Rieman-2003]
F. Riemann: *Grundformen der Angst*, Band 36, Reinhardt, 2003.
- [Royce-1970]
W. Royce: *Managing the development of large software systems*, Proceedings of IEEE Westcon, 1970.
- [Schulz von Thun-2002]
F. Schulz von Thun: *Miteinander Reden 3 – Das innere Team und situationsgerechte Kommunikation*, 9. Auflage, Rowohlt, 2002.
- [Schulz von Thun-2003]
F. Schulz von Thun: *Miteinander Reden 1 – Störungen und Klärungen*, 38. Auflage, Rowohlt, 2003.
- [Schulz von Thun-2004]
F. Schulz von Thun, W. Stegemann (Hrsg.): *Das innere Team in Aktion – Praktische Arbeit mit dem Modell*, Rowohlt, 2004.
- [Schulz von Thun-2005]
F. Schulz von Thun, J. Ruppel, R. Stratmann: *Miteinander Reden – Kommunikationspsychologie für Führungskräfte*. Rowohlt, 2005.
- [Schwaber-2004]
K. Schwaber: *Agile Project Management with Scrum*, Microsoft Press, 2004.
- [Schwaber-2007]
K. Schwaber: *The Enterprise and Scrum*, Microsoft Press, 2007.
- [Seifert-2001]
J. Seifert: *Visualisieren, Präsentieren, Moderieren*, 21. Auflage, Gabal, 2001.
- [Seiwert-2003]
L. Seiwert: *Das neue 1x1 des Zeitmanagements*, Gräfe und Unzer, 2003.
- [Sneed-2005]
H. Sneed: *Software-Projektkalkulation*, Hanser, 2005.
- [Standish-Chaos]
The Standish Group International, Inc.: *The CHAOS Report, 1994 – 2006*, <http://www.standishgroup.com>.
- [Varga-2000]
M. Varga von Kibéd, I. Sparrer: *Ganz im Gegenteil, Tetralemmaarbeit und andere Grundformen systemischer Strukturaufstellungen für Querdenker und solche, die es werden wollen*, Carl-Auer-Systeme Verlag, 2000.
- [Vigenschow-2003]
U. Vigenschow, C. Weiss: *Das Essenzschritt-Verfahren: Aufwandschätzungen auf der Basis von Use Cases*, Objekt-Spektrum 2/2003.
- [Vigenschow-2007]
U. Vigenschow, B. Schneider: *Soft Skills für Softwareentwickler – Fragetechniken, Konfliktmanagement, Kommunikationstypen und -modelle*, dpunkt.verlag, 2007.
- [Will-2002]
F. Will: *Was bremst mein Team?*, Beltz, 2002.
- [Wilson-1996]
J. Wilson, G. Kelling: *Broken Windows*, Kriminologisches Journal 1996, S. 121 ff.
- [Wolf-2005]
H. Wolf, S. Roock, M. Lippert: *Extreme Programming*, 2. Auflage, dpunkt.verlag, 2005.
- [Zahrnt-2005]
C. Zahrnt: *Richtiges Vorgehen bei Verträgen über IT-Leistungen*, 2. Auflage, dpunkt.verlag, 2005.
- [Zockoll-2005]
G. Zockoll, L. Gummels, H. Krasemann: *Iterativ-inkrementelle Entwicklung mit UML 2 im Projekt HALIS - Seehafen Kiel*, Vortrag OOP-Konferenz München, 2005.
- [Züllighoven-1998]
H. Züllighoven et al.: *Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz*, dpunkt.verlag, 1998.

10.3 Index

- Literaturhinweise
- [Beck-1999] 419, 430
 [Beck-2001] 287, 346, 430
 [Beck-2003] 13, 18, 430
 [Beedle-2002] 18
 [Berne-1970] 430
 [Boehm-1981] 81, 430
 [Boehm-1985] 13, 430
 [Boehm-2000] 74, 81, 430
 [Brooks-1995] 112, 267, 430
 [Bundschuh-2004] 81, 430
 [Caroll-2005] 286, 430
 [Cockburn-2003] 17, 18, 321, 418, 430
 [Cockburn-2004] 294, 430
 [Cohn-2005] 172, 430
 [Darley-1968] 25, 430
 [DeMarco-1997] 326, 430
 [DeMarco-1998] 61, 70, 430
 [DeMarco-1999] 430
 [DeMarco-2001] 61, 430
 [DeMarco-2003] 430
 [Derby-2006] 321, 368, 430
 [Eckstein-2004] 321, 430
 [Fisher-2004] 368, 430
 [Francis-1996] 328, 330, 430
 [Friedag-2002] 243, 430
 [Frohnhoff-2006] 286, 430
 [Gamma-2007] 18, 184, 324, 430
 [Goldratt-2001] 302, 431
 [Goldratt-2002] 296, 302, 306, 431
 [Harris-1975] 431
 [Highsmith-2004] 19, 431
 [Himmelreich-2006] 13, 431
 [Jacobson-1999] 428, 431
 [John-2004] 431
 [Jones-1996] 289, 326, 431
 [Jung-1990] 431
 [Kano-1984] 260
 [Kaplan-2001] 431
 [Kellner-1995] 368, 431
 [Kellner-1999] 368, 431
 [Kerth-2001] 321, 368, 431
 [Kohl-2000] 375, 431
 [König-2002] 431
 [Krasemann-2001] 283, 431
 [Krasemann-2005] 289, 403, 431
 [Larman-2004a] 431
 [Larman-2004b] 13, 431
 [Lipp-1996] 368, 431
 [Lippitt-1999] 431
 [Lörz-2007] 302, 306, 431
 [Madden-1984] 13, 431
 [Manns-2005] 431
 [Maro-2002] 368, 431
 [Mayrshofer-2006] 370, 431
 [Mohaghegi-2005] 286, 431
 [OEP-2007] 19, 102, 117, 119, 209, 218, 256, 276, 413, 431
 [Oestereich-2001] 431
 [Oestereich-2004] 87, 92, 96, 243, 431
 [Oestereich-2005] 432
 [Oestereich-2006] 240, 432
 [Ohno-2005] 188, 335, 421, 432
 [PMBok-2004] 228, 293, 385, 386, 432
 [Putnam] 326, 432
 [Rieman-2003] 432
 [Royce-1970] 12, 432
 [Schulz von Thun-2002] 432
 [Schulz von Thun-2003] 432
 [Schulz von Thun-2004] 432
 [Schulz von Thun-2005] 432
 [Schwaber-2004] 18, 426, 432
 [Schwaber-2007] 18, 432
 [Seifert-2001] 368, 432
 [Seiwert-2003] 349, 432
 [Sneed-2005] 79, 80, 81, 432
 [Standish-Chaos] 13, 47, 432
 [Varga-2000] 432
 [Vigenschow-2003] 55, 403, 432
 [Vigenschow-2007] 42, 321, 330, 337, 339, 345, 432
 [Will-2002] 432
 [Wilson-1996] 418, 432
 [Wolf-2005] 18, 168, 272, 287, 419, 432
 [www.oose.de] 330
 [Zahrnt-2005] 261, 432
 [Zockoll-2005] 402, 432
 [Züllighoven-1998] 90, 319, 432
- Abbildungen
- Abbildung
- Abnehmende
 Entscheidungsspielräume bei
 agilem Vorgehen 53
- Abnehmende Schätzungenauigkeit
 74
- Anwendungsfälle überspannen
 mehrere Iterationen 85
- Anwendungsfall-Feature-Matrix . 94
- Anwendungsfallpriorisierung 143
- Anwendungsfallpriorisierung
 (klassifiziert) 144
- Beispiel eines Arbeitsauftrages 200
- Darstellungsformen eines
 Projektstrukturplans 153
- Defektraten und Testabdeckungen
 bei Wasserfall- und iterativem
 Vorgehen 46
- Der Weg vom Projektziel zum
 Arbeitsauftrag 6
- Iterationsdauer in Abhängigkeit
 von der Projektgröße 56
- Iterationsfeature als Karteikarte 151
- Iterationsfeatures im Iterationsplan
 146
- Iterationsmodell 160
- Iterationsmodell, prinzipieller
 Aufbau einer Iteration 9
- Iterationsplan mit
 Iterationskapazitäten 141
- Iterationsplan mit Produktfeatures
 138
- Iterations-Wolken-Metapher 3
- Iteratives Vorgehen vs.
 Wasserfallvorgehen 45
- Justierter und erweiterter
 Iterationsplan 148
- Metamodell der Featurebegriffe . 96
- Mikroprozessmodell einer Iteration
 4
- Mikrozyklen innerhalb einer
 Iteration 159
- Personalkapazitäten 140
- Personalkurve 111
- Personalkurve (Beispiel) 141
- Planungsebenen und Feedback-
 Schleifen 4
- Preis- und Vertragsmodelle 82
- Projekt- und Releaseplan 126
- Projekt- und Releaseplan mit
 Iterationsraster 7
- Projektstrukturplan mit
 Produktfeatures 154
- Projektstrukturplan mit
 Produktfeatures und
 Iterationsfeatures 155
- Prozentuale Aufwandsverteilung
 nach Phase und Disziplin im
 RUP 80
- Systemkontext (Beispiel) 89
- Typische Elemente einer
 Projektorganisation 36
- Typische zeitliche Relation der
 einzelnen Phasen 107
- Verlauf der Teambelastung
 innerhalb einer Iteration 60

- A
- Abbruch des Projektes 116
- Abhängigkeit 32, 167, 171
- Abhängigkeitsmanagement 53, 416
- Ablage 39
- Ablagestruktur 70
- Ablauforganisation 416
- Abnahme 106, 107, 118, 119, 120, 126, 220, 416
- Teil- bzw. Zwischen- 397
- Abnahmekriterium 221, 237
- Abnahmeplan 398
- Abnahmetest 416
- Abnahmeumgebung 119
- Abnahmeverhindernd 398
- Abnahmewesen 237
- Abschlussphase 8, 106, 119
- Abschlussprozesse 230
- Abstraktion 416
- Abzugsfähiger Mangel 398
- Actual Cost of Work Performed 291
- ACWP 291
- AFI 218
- Agiler Festpreis 400
- Agiles Manifest 15, 86
- Ahting, Jürgen 411
- Aktives Zuhören 335
- Aktueller Staffelhölzträger 302, 303
- Akzeptanztest 416
- Allgemeine Projekt-
 Geschäftsbedingungen (APGs) 394
- Alpha-Test 416
- Alpha-Version 185
- Alpha-Versionen 8
- Always Alpha 185
- Always Beta 184
- Ampelstatus 186, 347, 349
- Analogiemethode 78, 426
- Analyse 178, 416
- Analysephase 45
- Änderung des Systems 240
- Änderungsantrag 257, 401
- Änderungsantragsverfahren 401
- Änderungskosten 53
- Änderungsmanagement . 40, 136, 258
- Änderungssteuerung 236
- Änderungswesens 265
- Änderungswunsch 395
- Anforderung 50, 416
- ausufernde 401
- instabile 400
- nicht funktionale 395
- stabile 54
- unbekannte 396
- Anforderungsänderung 193, 257
- Anforderungsbeitragende 360
- Anforderungseinheitspreis 411
- Anforderungserfüllung 259
- Anforderungsverantwortlicher 361
- Anforderungsworkshop 50
- initialer 69
- Angebot 391
- Angebotskalkulation 74
- Angebotsvorbereitungsworkshop... 69
- Angst 42
- Antoine de Saint Exupéry 244
- Anwendungsfall. 39, 94, 97, 114, 332, 416
- essenzieller 55
- Anwendungsfall-Feature-Matrix 94, 417
- Anwendungsfallpriorisierung 142, 143
- APG 129, 394
- API first 170
- APM 19
- Arbeitsauftrag.. 8, 162, 212, 274, 349, 417
- Status Entwurf 164
- Warteraum 163
- Arbeitsauftragsgutachter 166, 274
- Arbeitsauftragsreview 10, 198
- Arbeitsauftragsstatus 199, 275
- Arbeitsauftragsverantwortlicher ... 166
- Arbeitsauftragsverwaltung 189
- Arbeitsrhythmus 193
- Arbeitszeit 266
- Architektur 117
- Architekturabhängigkeitsmanagement
 54, 417
- Architekturmanagement 37, 136
- Architekturmuster 411
- Architekturteam 37, 193
- Arthur 127
- Asynchroner Lenkungsausschuss 210
- Audit 417
- Aufbauorganisation 417
- Aufgabe 417
- Aufgaben 8
- Aufgabenplanung 40
- Auftaktveranstaltung 177, 204
- Auftrag
- öffentlicher 401
- Auftraggeber 66, 360, 413
- Auftraggeber-Auftragnehmer-Jour fixe
 399
- Auftragnehmer 413
- Auftragsklärungsworkshop 69
- Aufwandschätzung 74, 166, 169
- Aufwandsfortschrittsindikator 218, 417
- Ausbaustufe 38
- Ausbildung 18
- Ausführungsprozesse 230
- Auslastungsgrad 266
- Auslieferung, stufenweise 106
- Ausschreibungssituation 401
- Aussitzen 374
- Ausufernde Anforderungen 401
- Auswertungen 211
- Autor-Kritiker-Treffen 316
- AW 417
- B
- Balanced Scorecard 417
- Barry Boehm 13
- Baseline 307, 333, 417
- Basisanforderung 259
- BCWP 291
- BCWS 291
- Bearbeitertag 418
- Bearbeitungsgrad 417
- Beck, Kent 419
- Begeisterungsanforderungen 260
- Belastung 60
- Benutzer 427
- Benutzergeschichte 154
- Benutzerverwaltung 40
- Benutzungskonzept 127, 417
- Berichtsweg 209
- Besitzstand 42
- Best-Practice-Modelle 229
- Beta-Test 417
- Beta-Version 185
- Betrieb 120
- Betriebseinführungsprozess 114
- Betriebsphase 120, 121
- Betroffene 360
- BGB 389
- Bid 391
- Bier trinken gehen 350
- Big Bang 418
- Boehm, Barry 13
- Broken Build 182
- Broken-Window-Theorie 418
- Brooks, Frederick P. 112
- BS 6079 227
- BT 418
- Buchhaltung 39
- Budgeted Cost of Work Performed
 291
- Budgeted Cost of Work Scheduled
 291
- Budgetobergrenze 400
- Budgetüberwachung 39
- Bugwellenplanung 8
- Build 110, 181, 397, 418
- Daily 181
- gebrochen 182
- Nightly 181
- privates 181
- Buildmanagement 40, 181
- Buildserver 70
- Bürgerliches Gesetzbuch 389
- Burnout 61
- Burn-up-Chart 293
- C
- C. Northcote Parkinson 76
- Capability Maturity Model Integration
 418
- CAPM 228, 418
- Carnegie Mellon University 418
- Cashflow 418

- Caveat emptor 390
 CCM 296, 303
 CCPM 296, 418
 Certified Associate in Project Management 228, 418
 Chance 418
 Change Control Board 232
 Change Request 395
 Change-Request-Verfahren 401
 Chaos-Report 13, 47
 Charisma 233
 ChefarchitektIn 37
 Critical-Chain-Projektmanagement 171
 CI 418
 CMMI 12, 50, 418
 Cockburn, Alistair 17
 COCOMO 81
 COCOMO II 81
 Codemanagement 70
 Continuous Integration 418
 Contractual Work Breakdown Structure 237
 Controlboard 209, 412, 418
 Controlling 39
 Conway's Law 32, 418
 Corporate Identity 418
 CPM 296
 Critical-Chain-Management 303
 Critical-Chain-Projektmanagement 131, 172, 264, 296, 418
 Critical-Path-Methode 264, 296
 Crystal 18, 50, 418
 CV 291
 CWBS 237
- D**
- Daily Build 181
 Daily-Scrum-Meeting 159, 183
 Darley, John 25
 Datenschutz 39
 Datensicherung 40
 David Maibor 12
 Deadline Buffer 301
 Definition
 Anwendungsfall 97
 Arbeitsauftrag 162
 Build 110, 181
 Feature 87, 97
 Iterationsfeature 97
 Meilenstein 102, 108, 191
 Phase 108
 Produktfeature 97
 Release 110, 181
 Releasefeature 97
 Smoke-Test 181
 Teamaufgabe 162
 Timebox 103, 191
 Deliverables 237
 Delphi-Methode 77, 276
 DeLuca, Jeff 19, 419
- Deming, W. Edward 230
 Deming-Zyklus 231
 Design 178
 Designmetrik 411
 Designphase 45
 Detailprinzip 75
 Deutsche Gesellschaft für Projektmanagement 420
 Dialogprototyp 261
 Dienstvertrag 389
 DIN 69900 227
 DIN 69905 391
 Diskriminierung 233
 Disziplin 20
 DoD-STD-2167 12
 Dokumentation 232
 Domänenexperte 419
 Dreipunktschätzung 167, 265, 278, 296
 Druck
 psychologischer 60, 177
 Drum Buffer 301
 DV-Leitung 413
- E**
- Earned Value 233, 419
 Earned-Value-Analyse 289, 293, 306, 307, 419
 EBIT 419
 ECF 285
 Eclipse 170
 Eclipse-Projekt 223
 Eclipse-Way 50, 178, 419
 effektiv 419
 effizient 419
 Eigenverantwortung 22
 Einführungsphase 119
 einführungsverhindernd 398
 Einkaufsmanagement 81, 389
 Einkaufsmanagementplan 390
 Einladungsfrist 395
 Einpunktschätzung 297
 Einspruchsfrist 395
 Eintrittswahrscheinlichkeit 384
 Eisbergmodell 337
 Eisenhower, Ike 192
 Eisenhower-Prinzip 348
 Eliyahu M. Goldratt 296
 Endabnahme 221
 Endgame 58, 223, 324, 419
 Endphase 106, 119
 Endspiel 58, 223
 Engpass 167, 171, 419
 Engpassanalyse 172
 Entwicklungsarbeitsplatz 70
 Entwurfs- und Architekturphase 117
 Entwurfsmuster 419
 Entwurfsphase 106
 Environmental Complexity Factor 285
 Erfolgsfaktor 47, 69, 383
 Erfolgsfaktorworkshop 71, 375
- Erfolgskriterium 383
 Erfolgsrechnungsergebnistypen 419
 Ergebnis
 betriebswirtschaftliches 419
 Ergebniskennzahlen 419
 Ergebnisverantwortlicher 166
 Erledigungsgrad 419
 Erweiterbarkeit 409
 Essen gehen 350
 Essential Use Case Steps 397
 Essenzieller Anwendungsfall 55
 Essenzschritt-Verfahren 432
 EV 419
 EVA 289, 293, 419
 Externer Auftraggeber 66
 Extreme Programming 13, 18, 419
 Exupéry 244
 Exzellenz 20
- F**
- Fachabteilung 360, 419
 Fachbereichsleitung 413
 Fachexperte 361, 419
 Fachkonzept 90
 Fachliche Architektur 117
 Fachliche Teilabnahme 117
 Fachliches Featureteam 37
 Fallbeispiel 65
 Faustformel 287
 FDD 19, 419
 Feature 50, 73, 87, 97, 115, 249, 419
 Arten 92
 Beschreibung 149
 Iterations- 93, 399
 Produkt- 92
 Feature Creep 262, 401
 Feature Driven Development 19, 419
 Feature, planungsrelevantes 87
 Featureabschlussauftrag 164, 171
 Featurearten
 Metamodell 96
 featurebasiert 198
 Featurebasiertes Planen 51
 Featurekosten 200
 Featureliste 239
 Featureteam 37, 136, 193
 Feedback 317, 328, 339
 wöchentliches 159
 Feeding Buffer 301
 Fehlerbehebung 58
 Fehlerklasse 420
 Feinplanung 162
 Feldtest 420
 Fertigstellungsgrad 200, 212, 420
 Fertigstellungswert 290, 291, 306, 307
 Fertigstellungswertmethode 307, 420
 Festpreis 389, 405, 407
 agiler 400
 Finanzmanagement 39
 First-Level-Support 38
 Fishbowl 341

- Flow 329
 Fluktuation 329
 Fluktuationsrate 219
 Forming 328
 Fortlaufende Integration 420
 Fortschrittsmessung 193
 Fortschrittsabschnitt 8, 159, 168
 Fortschrittskontrolle 289, 293
 Fortschrittswert 233
 Fotoprotokoll 395
 Freigabe 114
 Freigabeverfahren 38
 Freud, Sigmund 337
 Führung 21, 233
 Führungsspanne 31
 Function Points 81, 281, 283, 397, 411
- G
- GAF 420
 Gebrochenes Build 182
 Gemeinnützige Organisation 420
 Gesamtarchitektur 136
 Geschäftsanwendungsfall 420
 Geschäftsbedingungen (APGs) 394
 Geschäftsprozess 39, 420
 Geschäftsprozessmodellierung 420
 Geschäftsprozessesteam 193
 Gesellschaft für Projektmanagement 29, 420
 Gewährleistung 390
 Gewichtungsmethode 78, 426
 Gewinnerzielungsabsicht 420, 428
 Goldratt 296
 Good Practice 229
 GP 420
 GPM 23, 29, 420
 Grobplanung 162, 164
 Großgruppen-Priorisierung 252
 Großprojekt 33, 65
 Gruppendynamik 27, 326
 Gruppendynamischer Prozess 27, 327
 Guide to the Project Management Body of Knowledge 228
 Gummibärchen 168
 Gutachter
 eines Arbeitsauftrages 166, 274
- H
- Hauptpersonalrat 413
 Hauptphase 7, 106, 118
 Heuristik 420
 Hindernisbericht 350
 Hochleistungsphase 329
- I
- IBM 13
 Idealer Tag 76, 168, 267, 420
 Identifikation mit dem Projekt 263
 identifizieren 420
- IF 146, 421
 Implementierung 178
 Inbetriebnahme 107
 Incentives 390
 Indikator Siehe Kennzahl
 Ineffizienz
 lokal 306
 Informationsverteilung 333
 Inhalt 236
 Inhaltsmanagement 40, 236
 Initiierungsprozesse 230
 Inkrement 151, 421
 Innenfinanzierungspotenzial Siehe Cash Flow
 Innovationsmodell 258
 Instabile Anforderungen 400
 Integration 421
 Integrationsbuild.. 106, 184, 220, 221, 421
 Integrationsfehler 184
 Integrationsmanagement 234
 Integrationsserver 70
 Integrationsteam 40
 Integrationstest 119, 421
 Integrierte Planung 232
 Integrität 233
 Interessenshalter 360
 Interessenkonflikte 170
 Interessenvertreter 68
 International Organization for Standardization 421
 International Project Management Association 228, 421
 Internationale Projektverträge 390
 Interner Auftraggeber 66
 Interventionssignal 186
 Intrinsische Motivation 61
 IPMA 228, 421
 ISO 421
 ISO-Norm 227
 Istaussgaben 291
 Istkosten 290
 Iteration 421
 Fortschrittsabschnitt 159
 Orientierungsabschnitt 159
 synchrone 58
 Iterationsabschlussauftrag 192
 Iterationsauftrag 204
 Iterationsauftragstreffen 177
 Iterationsdauer 55, 56
 Iterationsebene 5
 Iterationsfeature.. 8, 93, 97, 161, 192, 399, 421
 Planung 144
 planungsrelevantes 96
 Iterationsfeaturereviews 223
 Iterationsfeatures 5, 52, 146
 Iterationskapazität 136, 139, 421
 Iterationsmanagement 40, 136
 Iterationsplan 5, 8, 138, 421
 Iterationsraster 131
 Iterationsreview 201
- Iterations-Risikoliste 399
 Iterations-Wolken-Metapher 3
 Iterationsziel 161
- J
- Jour fixe 351, 399
 Jung, Carl Gustav 343
 Jung'sche Typenmodell 343
- K
- Kaizen 188, 421
 Kalkulationsworkshop 69
 Kano, Noriaki 258
 Kaufmännisches Projektmanagement 50
 Kennzahl 214, 242
 zur Zielerreichungsmessung 244
 Kennzahlen 218
 Kennzahlensystem
 Earned-Value-Analyse 291
 Kernfunktionalität 54
 Kern-Geschäftsanwendungsfall 421
 KFI 218, 422
 Kick-off 177
 Kleinprojekt 30
 Kommunikation 22, 42, 233, 363
 Kommunikationsmanagement 39, 333
 Kommunikationsmanagementplan 333
 Kommunikationsplan 333, 351, 422
 Kommunikationsproblem 335
 Komplexität 324
 Komponententeam 40
 Konfigurationsmanagement 40, 70
 Konflikt 42
 Konfliktmanagement 330
 Konfliktphase 328
 Konfliktvermeidung 313
 Konsolidierung 422
 Konstruktionsphase 106, 119
 Konstruktive Qualitätssicherung 316
 Konventionalstrafe 390
 Konzentrationsworkshop 308
 Kooperation 22
 Korrekturfaktor 168
 Korrekturrelease 107
 Korrespondenz 39
 Korruption 233
 Kosten 289, 419
 für Änderungen 53
 Kostenabweichung 291
 prozentuale 291
 Kostenbaseline 307
 Kostenfortschrittsindikator 218, 422
 Kostenmanagement 306
 Kosten-Nutzen-Analysen 74
 Kosten-Nutzen-Priorisierungsmatrix 309
 Kostenperformance 218, 422
 Kostenplanung 307
 Kostenschätzung 307
 KP 218

- Krankheit 61
 Krankheitsquote 219
 Krisensituation 183
 Kritische Kette 171, 173, 303, 422
 Kritische-Pfad-Methode 297
 Kritischer Pfad 171, 264, 296, 422
 Kulanzrechnung 408
 Kurskorrektur 178, 193
- L
- Lastenheft 391
 Latane, Bibb 25
 Läufer 189, 349, 422
 Lean-Management 188, 230
 Lebenszyklus 410
 Leistungsanforderung 259
 Leistungsanreize 390
 Leistungsberichtswesen 333
 Leistungsbeschreibung 391
 Leistungserbringung 313
 Leistungsmerkmal 86, 92
 Leistungsmessung 307
 Leitbild 422
 Leitziel 422
 Lenkungsausschuss.. 209, 422, Siehe
 Lenkungsreis
 Lenkungsreis 209, 383, 398, 412,
 422
 Lieferantenauswahl 391
 Liefergegenstand 153, 237
 Linienorganisation 61, 412
 Liquidated Damages 390
 Lizenzverwaltung 40
 Load-Faktor 76, 168, 267, 422
 LOC 422
 Lokale Effizienz 427
 Lokale Ineffizienz 306
 Look-ahead-Plan 172
- M
- Macht 27, 42
 Machtkampf 27, 327, 328
 Maibor, David 12
 Makroschätzung 75, 216, 422
 Managementplanung 232
 Mangel
 abzugsfähiger 398
 zu behebbender 398
 Mängel 120
 Mängelbeseitigung 107
 Manifest
 agiles 15
 Manntag 423
 Marvin 127
 Maßnahme 243
 zur Zielerreichung 245
 Materialwesen 39
 Mediator 189
 Megaprojekt 34
 Meilenstein .. 102, 103, 108, 110, 126,
 181, 191, 253, 255, 264, 422
 der Abschlussphase 120
 der Entwurfsphase 117
 der Hauptphase 119
 Meilensteinauditor 109
 Meilensteinerreichung 109
 Meilensteinliste 264
 Meilenstein-Trendanalyse 294
 Meilensteinvereinbarung 255
 Melvin Conway 32
 Messung 74, 211
 Metakommunikation 320
 Methode 423
 Methodik 423
 Metriken 233
 Migrationsteam 38, 39, 193
 Mikroschätzung 75, 216, 423
 Mikro-Timebox 275
 Mission 423
 Missverständnis 335
 Mitarbeiterkapazität 266
 Mittleres Projekt 31
 Mitwirkungsleistung 395
 Moltke, Graf von 192
 Motivation 61, 177, 193
 Motivation, intrinsisch 169
 MT 423
 MTA 294
 Multiplikatormethode 78, 79, 426
- N
- Nachhaltigkeit 20
 NASA 13, 228
 Netzplan 297
 Netzplantechnik 264
 Newbie-Mentor 331
 Nicht funktionale Anforderung 115,
 395, 409
 Nightly Build 181
 Norming 328
 Northcote Parkinson 76
 Nulltoleranzstrategie 418
- O
- Object Points 81
 OEP 19, 50, 80, 102, 423
 Öffentlicher Auftrag 13, 401
 OMG 423
 oose Engineering Process 19, 102,
 423
 Operativ 423
 Operatives Ziel 241, 244, 423
 Organigramm 423
 Organisation 423
 gemeinnützige 420
 Organisationseinheit 423
 Organisationsentwicklung 233
 Organisationsplan 423
 Organisationsstruktur 30, 193
 Organisationsphase 328
 Orientierungsabschnitt 8, 159, 168
 Orientierungsphase 328
- P
- Pair-Programming 331
 Paralleles Releaseteam 130
 Paralleles Releaseteams 222
 Parallelrelease 10
 Parametrische Schätzgleichung 78, 81
 Pareto-Prinzip 423
 Parkinson
 C. Northcote 76
 Parkinson's Law 76, 220, 423
 PDCA-Zyklus 231
 Performance-Nachweis 117
 Performing 329
 Personalbedarfsplanung 235, 324
 Personalentwicklung 39
 Personalkurve 139
 Personalmanagement 39, 324
 Personalmanagementplan 324
 Personalwesen 39
 Personenjahr 266
 Personenmonat 266
 Personenstunde 266
 Personentag 266, 423, 425
 Personenwoche 266
 PERT 279, 291
 PERT-Formel 298
 PF 285, 423
 Pflichtenheft 235, 237
 PgMP 228, 423
 Ph 266
 Phase 108
 Phasenfestpreis 407
 Phasenmeilenstein 109
 Phasenmodell 45, 229, 230
 beim iterativen Vorgehen 47
 beim Wasserfallansatz 45
 Pilotumgebung 119
 PJ266
 Planabweichung 291
 prozentuale 291
 Planausgaben 291
 Plankosten 290
 Planung 232
 Planung aktualisieren 178
 Planungseinheiten 168, 265
 Planungskorrektur 10, 178
 Planungsphase 230
 Planungsprozesse 230
 Planungspuffer 219
 Planungsrelevantes Feature .. 87, 423
 Planungsrelevanz 87
 Planungsspiel 271
 Planungsunsicherheit 400
 Planungswand 166, 167, 173, 182,
 183, 189, 272
 Planungsworkshop 267
 PM v, 19, 266
 PMI 18, 23, 29, 227, 228, 292, 424,
 432
 PMP 18, 228, 424
 Präparationsworkshop 68, 69

- Präventives Risikomanagement ... 376
 Preis des Systems 239
 Preisangebot 391
 PRINCE2 229, 424
 Prinzip 424
 Priorisierung 163, 252, 363
 vereinfachte 242
 zielgetriebene 242
 Priorisierungsstrategie 128
 Prioritätsgruppen für Interessenhalter
 364
 Privates Build 181, 424
 Procurement 389
 Product Backlog 18
 Product Scope 236
 Productivity Factor 285
 Produktfeature 51, 92, 97, 424
 planungsrelevantes 95
 Produktgedanke 86
 Produktgröße 288
 Produktionsfreigabe 120
 Produktionsumgebung 120, 220
 Produktivität 288, 325
 Produktivumgebung 424
 Produktkarton 72, 88, 238, 412
 Produktorientierung 39
 Produktstruktur 32
 Produktteam 38, 424
 Program Management Professional
 228, 424
 Programmiersprache 325
 Project Buffer 301
 Project Charter 234
 Project Management Institute .. 18, 29,
 227, 228, 424
 Project Management Professional. 18,
 228, 424
 Project-Velocity 349
 Projekt 424
 Projekt- und Releaseplan 5, 382
 Projektabbruch 116
 Projekttakte 235, 424
 Projektassistenz 39
 Projektauftrag 234, 238, 392, 412
 vorbereiten 69
 Projektbetroffene 360
 Projektbüro 34, 39
 Projektdauer 288
 Projektdurchführungsstrategie 13
 Projektebene 5, 116, 122, 135
 Projektende 220
 Projekt-Geschäftsbedingungen
 (APGs) 394
 Projektgröße 55, 56
 Projektintegrationsmanagement ... 234
 Projektkostenmanagement 306
 Projektleitung 351
 Projektleitungs-Jour fixe 351
 Projektleitungsstab 39
 Projektleitziel 241
 Projektmanagement, kaufmännisches
 50
 Projektmanagement-Jour fixe 349
 Projektmanagementplan 235
 Projektmarketing 263, 352, 378
 Projektname 263
 Projektorganisation 117, 332
 Projektparty 105, 354
 Projektplan 161, 267
 Projektplanung 321
 Projektprozesse 117
 Projektpuffer 424
 Projektrisiko 382, 384
 Projektrisikomanagement 371
 Projektsponsor 378
 Projektstab 34
 Projektstrukturplan 153, 235, 237,
 245, 246, 425
 Projekttagebuch 379
 Projektteam 324
 Projektzeitmanagement 264
 Projektziel 69
 Projektziele 233
 Proposal 391
 Prosecco-Event 177, 354, 355
 Prototyp 261, 387
 Prozentsatzmethode 78, 79, 426
 Prozentuale Kostenabweichung .. 291
 Prozentuale Planabweichung 291
 Prozess 230
 Prozessgruppe 230
 Prozesskette 425
 Prozessteam 39, 193
 Prozessverantwortlich 39
 Prozessverantwortlicher 331
 PSP 245, 425
 Psychologischer Druck 177
 PT 266, 425
 Puffer 214, 218, 219
 Putnam 287
 PW 266
- Q
- Qualität 425
 Qualitative Risikoanalyse 372
 Qualitätslenkung 314
 Qualitätsmanagement 39, 314
 Qualitätsmerkmal 425
 Qualitätsplanung 314
 Qualitätssicherung 314
 konstruktive 316
 Qualitätssicherungsmaßnahme .. 173,
 199
 Qualitätssteuerung 314
 Quality Control 314
 Quandantenmodell 344
 Quantitative Risikoanalyse 372
 Querschnittliches Team 193
 Quotation 391
- R
- Rahmenbedingungen 242
 Rational Unified Process 80
 Räumliche Verteilung 57
 Raumverwaltung 39
 RBS 265
 Realisierungsgrad 212, 218, 425
 Realisierungsphase 45
 Realisierungswettbewerb 380
 Rechtswesen 39
 Redundanz 425
 Refactoring 204, 425
 Reforming 329
 Regressionstest 425
 Reichsbedenkenträger 131, 425
 Reiseorganisation 39
 rekursiv 425
 Relationsmethode 78, 426
 Release. 10, 106, 110, 119, 126, 181,
 220, 425
 Releaseebene 5
 Releaseeinführungsteam 38
 Releaseentwicklung 221
 Releasefeature 97, 143, 425
 Releasemanagement 40, 116, 122,
 136
 Releasename 127
 Releasenummer 127
 Releaseorientierte
 Projektorganisation 425
 Releaseplan 161, 397, 425
 Releaseplanung 136, 323
 Releaseprojektteam 38
 Releaseteam ... 10, 38, 128, 137, 221,
 426
 paralleles 130
 Resignation 42
 Resource Breakdown Structure 265
 Ressourcenbedarfsschätzung 265
 Ressourcenengpass 167, 171
 Ressourcenkonflikte 170
 Ressourcennivellierter kritischer Pfad
 426
 Ressourcenstrukturplanung 265
 Restaufwandschätzung 77, 191
 Restrukturierung 58, 178, 204, 426
 Retrospektive. 10, 159, 161, 201, 319,
 321
 Review 426
 Arbeitsauftrag 198
 RG 218
 Risiko 171, 243, 426
 Risikobericht 399
 Risikobewältigung 386
 Risikobewältigungsplanung 372
 Risikoidentifikation 372
 Risikoliste 382, 399
 Risikomanagement 23, 39, 54, 233,
 321, 363, 371, 375, 378, 379, 380,
 383, 397
 präventives 376
 Risikomanagementplanung 371
 Risikoregister 371
 Risikoüberwachung 372
 Risikoworkshop 382

- initialer 69
 Robustheit 409, 426
 ROCE 426
 ROI 426
 Rolle 426
 Rollout 38
 Rollout-Team 38
 Royce, Walker 12
 Royce, Winston 12, 431
 Rückmeldungszyklus 187
 RUP 80, 428
- S
- SAF 426
 Schadenersatzforderung 390
 Schadenshöhe 384
 Schätzen 276
 Delphi-Methode 276
 Dreipunkt-Schätzung 278
 Faustformel 287
 Schätzkurve 280
 Softwaregleichung 287
 Wetter-von-Gestern 286
 Widget-Point-Verfahren ... 281, 283
 Schätzfehler 215
 Schätzgenauigkeit 75
 Schätzgleichung 81
 Schätzkurve 280
 Schätzmethode 77, 426
 Schätztechnik 74, 77
 Schätzung 211, 216
 Schätzung, personenspezifisch ... 169
 Schätzverfahren 74
 Schedule Variance 291
 Schwaber, Ken 426
 Schwimmbahndiagramm 299
 Scope 236
 Scrum 18, 50, 154, 172, 174, 183, 200, 426
 Second-Level-Support 38
 SEI 418
 Selbstorganisation 20, 24
 Sequenzielle Releaseteams 221
 Server-Administration 40
 SEU 288, 325
 Shewhart, Walter A. 230
 Sicherheit 39
 Six Sigma 230
 Smoke-Build 181, 182, 221, 426
 Smoke-Test 117, 181, 426
 Sneed, Harry 79
 Soft Skills 335
 Softwarearchitektur 32, 181
 Softwareentwicklungsumgebung 288, 325
 Softwaregleichung... 69, 81, 111, 139, 267, 287
 Softwarelebenszyklus 410
 Sorgfaltspflicht 390
 soziale Spannungen 183
 Space-Shuttle 13
- SPICE 426
 Spike 261, 426
 Spiralmodell 13
 Sponsor 378
 Sprint 18, 427
 Sprint Backlog 18
 Sprintplanungssitzung 162
 Sprint-Rückschau-Sitzung 174, 200
 Stab 427
 Stabilisierungsiteration 129, 159, 223, 323, 427
 Stabilisierungsiterationen 58, 223
 Stabilität 54
 Stabsfunktion 39, 427
 Staffelläuferprinzip 427
 Stakeholder 360, 427
 Workshop 68
 Stakeholder-Analyse 68, 74, 359
 Stakeholder-Diagramm 27, 69, 362
 Stakeholder-Management 333
 Stakeholder-Orientierung 233
 Stakeholder-Priorisierung 69, 363
 Stakeholder-Profil 366
 Standish Group 13
 Standup-Meeting 183
 Startphase 7, 106, 113
 Statement of Work 391
 Status
 eines Arbeitsauftrages 275
 Statusbericht 185, 347, 349
 Statusverteilung 212
 STD-2167 12
 Steering Committee 209, 412, 427
 Stehung 183, 345
 Steuerung der Kosten 307
 Steuerungsausschuss 427
 Steuerungskreis. 209, 412, 427, Siehe Lenkungskreis
 Steuerungsprozesse 230
 Steuerungstreffen
 tägliches 159
 Storming 328
 Strategiekarte 427
 Strategisch 427
 Strategisches Ziel 242, 244, 427
 Stresstest 427
 Studentensyndrom 427
 Subiteration 26
 Subsystemstruktur 39
 Subsystemteam 40, 136
 SV 291
 Synergie 427
 Systemadministration 40
 Systemakteur 427
 Systemanwendungsfall 427
 Systembetreffende 360, 361
 Systemidee 72
 Systemintegration 40
 Systemische Ordnung 365
 Systemkontext 89
 Systemmigration 39
 Systemvoraussetzungen 72, 239
- T
- Tägliches Teamtreffen 159
 Task-Card 166
 TCF 284
 Team- und Iterationsebene 5
 Teamaufgabe 162
 Teamaufgaben 8
 Teambildung 324
 Teamentwicklung 39, 330
 Teamkonflikte 325
 Teamkoordination 332
 Teamleitung 31
 Teamorientierung 232
 Teamsteuerungstreffen
 tägliches 159
 Teamstruktur 32
 Teamtraining 324, 330
 Teamtreffen
 tägliches 159
 Technical Complexity Factor 284
 Technische Architektur 117
 Technischer Prototyp 127
 Technisches Featureteam 37
 Technologie-Coach 37
 Teilabnahme . 10, 106, 220, 221, 397, 428, 429
 Teilabnahmen 106
 Teilprojekt 40
 Telefonzentrale 39
 Termindruck 60
 Terminkalender 264
 Terminorganisation 39
 Terminperformance 218, 428
 Terminplan 235, 265
 Terminplanung 297
 Test 178
 Integration 119
 Testautomatisierung 54, 181, 428
 Testdefinition 178
 Testgetriebene
 Anforderungsdefinition 179
 Testgetriebenes Design . 54, 179, 428
 Testphase 45
 Testumgebung 119
 Theory of Constraints 297, 428
 Tiefpassfilter 193
 Time and Material 389
 Timebox 5, 103, 191, 428
 Timeboxing 191
 Top-10-Liste 382, 384
 Toyota-Produktionssystem 188
 TP 218, 428
 Training 18
 Transaktion 428
 Transition 428
 Transparenz 22
 Trillian 127
 Truck-Faktor 428
 T-Stich-Prototyp 387
 Typenmodell 343

- U
- UAW 284
- UC 428
- Umfang 236
- Umfangsmanagement 236
- UML 428
- Umorientierungsphase 329
- Umsatzrendite 244, 245
- Unadjusted Actor Weight 284
- Unadjusted Use Case Points 284
- Unadjusted Use Case Weight 284
- Unbekannte Anforderung 396
- undiszipliniertes Arbeiten 183
- Unified Modeling Language 428
- Unified Process 50, 428
- Unit Test 184
- unproduktive Arbeitszeit 422
- Unternehmensidentität 428
- Unternehmensleitziel 241
- Unternehmenspolitik 42
- Unternehmenszweck 428
- Unterstützender
Geschäftsanwendungsfall 428
- Unterstützer 378
- Ursachenkategorien 73
- Use Case 429
- Use Case Points 397, 411
- Use-Case-Point-Methode 283
- User Story 154
- US-Verteidigungsministerium 228, 418
- UUCP 284
- UUCW 284
- V
- Verantwortung 19, 20, 23, 26
zurückgeben 24
- Verantwortungsdiffusion 25, 429
- Verbesserungsprozess 319
- Vereinfachte Priorisierung 242
- Versionierungsrichtlinie 70
- vertikaler Prototyp 388
- Vertrag 313
- Vertrags- und Preismodell 411
- Vertragsabwicklung 391
- Vertragsänderung 391
- Vertragsbeendigung 391
- Vertragsgestaltung 407
- Vertragsstrafe 390
- Vertrauen 20, 22, 24, 42
- Viewpoint Resolution 359
- V-Modell 12
- V-Modell XT . 13, 19, 44, 50, 102, 229
- Vorabnahme 10, 106, 126, 221
- Vorabnahmen 118, 129
- Voraussetzungen für Software 239
- Vorbedingung 429
- Vorbereitungsphase 6, 105
- Vorschlag 391
- W
- Wahrscheinlichkeitsabfragen 185
- Walker Royce 12
- WAM-Methode 90, 319
- Warnschilder 184
- Wartbarkeit 409
- Warteraum 165, 173
- Warteraum für Arbeitsaufträge 163
- Wartung 120, 122, 409
- Wartungsorganisation 107
- Waschmaschinenmodell 179
- Wasserfallmodell 178
historischer Irrtum 13
Nachteile 45
- WBS 237, 246, 264
- Weak Matrix 233
- Weglassworkshop 308
- Weiterentwicklung 122, 409
- Weiterentwicklungsorganisation .. 120
- Werkvertrag 389
- Werkzeugkasten 229
- Wert 429
- Werte 20
- Wertschätzung 22, 24
- Wertschöpfungsbeitrag 54
- Wette 217
- Wetter-von-gestern 286
- Widget Points 81, 397, 411
- Widget-Point-Verfahren 281
- Wiederverwendungsprinzip 75
- Wiki 429
- Winston Royce 12
- Wirtschaftsjahr 58
- Wissensgebiete 232
- Wöchentliches Feedback 159
- Work Breakdown Structure... 237, 246
- Workflow 429
- Workshop
Anforderungen 69
Angebotsvorbereitung 69
Auftragsklärung 69
Kalkulation 69
Risiko 69
Stakeholder 68
- X
- XP 13, 18, 50, 154, 429
- Z
- ZAK-Karte 241
- Zaphod 127
- Zeit- und Inhaltsmanagement 40
- Zeitereignis 427
- Zeitmanagement 264
- Zero Tolerance 418
- Ziel 40, 324, 366
Beschreibungsschema 244
operatives 244, 423
Projektziel 69
strategisches 244, 427
- Zielart 242
- Zieldefinition 241, 243
- Ziele 22, 26
aktiv kommunizieren 177
- Zielerreichung 352
- Zielformulierung 72
- Zielgetriebene Priorisierung 242
- Zielgewicht 242
- Zielumgebung 119, 120
- Zielvereinbarung 241, 243
- Zielvereinbarungen 136
- Zielwert 242
- Zu behebbender Mangel 398
- Zulieferkette 429
- Zulieferleistung 77, 429
- Zulieferpuffer 429
- Zwischenabnahme 397