Xpert.press

Manfred Broy
Marco Kuhrmann

Projektorganisation und Management im Software Engineering

Springer Vieweg

Die Reihe **Xpert.press** vermittelt Professionals in den Bereichen Softwareentwicklung, Internettechnologie und IT-Management aktuell und kompetent relevantes Fachwissen über Technologien und Produkte zur Entwicklung und Anwendung moderner Informationstechnologien.

Manfred Broy · Marco Kuhrmann

Projektorganisation und Management im Software Engineering

Springer Vieweg

Manfred Broy, Marco Kuhrmann
Fakultät für Informatik, Lehrstuhl für Software &
Systems Engineering
Technische Universität München
Garching, Deutschland

ISSN 1439-5428
ISBN 978-3-642-29289-7 ISBN 978-3-642-29290-3 (eBook)
DOI 10.1007/978-3-642-29290-3

Springer Vieweg

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier.

Springer Vieweg ist eine Marke von Springer DE. Springer DE ist Teil der Fachverlagsgruppe Springer Science+Business Media
www.springer-vieweg.de

Geleitwort

Projektarbeit ist anders als Routinebetrieb. Unternehmen, die einen Flughafen betreiben oder Stahl herstellen, sind nicht darin geübt, große Projekte durchzuführen. Das zeigt sich derzeit deutlich beim Bau des Hauptstadtflughafens und bei den Stahlwerken von ThyssenKrupp in Süd- und Nordamerika. In Berlin hat man gemeint, sich einen Generalunternehmer sparen und das Projekt als Flughafengesellschaft selbst managen zu können. Eine teure Fehleinschätzung.

Fortschritt bei Bauvorhaben ist augenscheinlich: man sieht das Gebäude wachsen. Nicht so bei der Softwareentwicklung, denn Softwaresysteme sind abstrakte Gebilde, sie gehören zum Komplexesten, was Menschen geschaffen haben. Selbst wenn schon viele Zeilen Code da sind, kann man nicht ohne weiteres erkennen, ob sie leisten, was sie sollen, und wie viel noch fehlt. Diese Unsichtbarkeit von Software macht es schwer, sie zu beurteilen, etwa hinsichtlich Qualität und Entwicklungsstand.

Das Management von Softwareprojekten ist keine leichte Aufgabe. Es ist schwierig, Software mit hoher Qualität, in meist knapp bemessener Zeit und zu akzeptablen Kosten herzustellen. Dafür braucht man hochqualifizierte Mitarbeiter, die im Team und auch selbständig arbeiten, die systematisch und kreativ, termingerecht und sorgfältig vielfältige Aufgaben bewältigen (Spezifizieren, Programmieren, Testen, Integrieren, Dokumentieren). Ihre Führung und Anleitung verlangt vom Projektmanager fachlich-technische Kompetenz und persönliche Qualitäten zugleich. Der ideale Projektleiter ist kommunikativ, motivierend und überzeugend, zielgerichtet und durchsetzungsstark.

Immer wieder habe ich Extreme beobachtet: einerseits formal-bürokratisches Vorgehen mit überzogener Struktur, Reglementierung, Kontrolle und Dokumentation, bei dem der Projektleiter sich hauptsächlich mit Planungstools beschäftigt, und andererseits Laisser-faire nahezu ohne Struktur und Dokumentation, wobei alle vermeintlich alles im Kopf haben und kommunikativ munter werkeln. In beiden Fällen mangelt es an Führung. Ein Projektleiter muss den richtigen Mittelweg finden. Selbstverständlich bedarf es in großen Projekten mit zig oder gar Hunderten von Entwicklern einer formaleren Vorgehensweise, Steuerung und Kontrolle als bei kleinen Projekten, die in freierem Stil geleitet werden können und sollten.

In erster Linie kommt es auf die Führung des Teams an, erst in zweiter Linie sind die stärker systematisierbaren Tätigkeiten des Planens, Kontrollierens und Verwaltens wichtig.

Dieses Buch behandelt beides. Es zeigt, wie Projekte zu organisieren und in die beteiligten Unternehmen einzubinden sind, welche Führungsformen es gibt und welche Wirkung sie zeigen. Eingehend beschrieben werden die verschiedenen Vorgehensmodelle und die Methoden, Techniken und Werkzeuge zur Projektplanung, -steuerung und -kontrolle, insbesondere zur Aufwandsschätzung, einer besonders heiklen Aufgabe der Projektleitung. Damit ist auch eine Grundlage geschaffen für betriebswirtschaftliche und vertragsrechtliche Aspekte.

Dieses Buch bietet eine umfassende Darstellung des Managements von Softwareprojekten. Es gibt wohlweislich keine konkreten Handlungsanleitungen, sondern zeigt das Instrumentarium auf, dessen sich der Projektmanager bedienen kann. Die Auswahl und Anpassung seiner Mittel muss er im konkreten Fall schon selber besorgen. Möge ihm das Buch eine Hilfe sein.

Grünwald, Dezember 2012 Ernst Denert

Vorwort

Mit dem Begriff *Software Engineering* adressieren wir ein ingenieurmäßiges Vorgehen bei der Entwicklung umfangreicher, leistungsstarker Softwaresysteme. Dies umfasst Aufgaben der Softwaretechnik, das heißt der fachlichen, methodischen und technischen Entwicklung und Realisierung von Software in allen Phasen der Systementwicklung und des Softwarelebenszyklus und Themen der Projektorganisation und des Managements von Softwareprojekten. Daneben finden sich projektübergreifende Aufgaben zur Sicherstellung der Fähigkeiten eines Unternehmens zur Entwicklung softwareintensiver Systeme wie Unternehmensorganisation, Schulung, Aus- und Weiterbildung, Bereitstellung der Infrastruktur für die Softwareentwicklung, aber auch die Erhöhung des Reifegrades in Unternehmen durch kontinuierliche Verbesserung der Entwicklungsprozesse für Software.

Das Software Engineering hat in nur wenigen Jahrzehnten in der Praxis eine hohe wirtschaftliche und technische Bedeutung erlangt. Produkte wie Automobile, Flugzeuge, mobile Telefone oder Blue Ray Player enthalten heute in aller Regel eingebettete Systeme mit umfangreichen Softwareanteilen. Die inzwischen oftmals komplexen Verwaltungs-, Vertriebs- und Versorgungsprozesse der Industrie und Verwaltung stützen sich maßgeblich auf Informatikinfrastruktur ab, deren Funktionalität und Leistungsfähigkeit fast ausschließlich von der realisierten Software bestimmt wird. Somit kommt der Fähigkeit der Wirtschaft, schnell kosten- und nutzungsgerecht Software hoher Qualität zu entwickeln, entscheidende wirtschaftliche Bedeutung zu. Die Konkurrenzfähigkeit ganzer Branchen und die Existenz vieler Unternehmen hängen vital von der Qualität der Softwaresysteme ab und der Fähigkeit, diese angemessen weiter zu entwickeln. Immer stärker stützen sich die Infrastrukturen wie Stromversorgungsnetze, Verkehrssteuerung für Straße, Schiene, Luft, aber auch Geldüberweisungssysteme, Medizinsysteme und Verwaltungseinrichtungen kritisch auf die Funktionsfähigkeit von Software ab.

Software Engineering – Zwischen Wissenschaft und Praxis Gerade das Software Engineering hat – neben einigen technisch gesicherten Erkenntnissen – auch eine Reihe von Einsichten angesammelt, die rein pragmatisch sind, da sie sich auf der Erfahrung aus einer Reihe von Projekten stützen und nicht im eigentlichen Sinn wissenschaftlich korrekt nachgewiesen sind. Damit zeigt das Software Engineering eine für praktische Ingenieurwissenschaften typische Mischung aus Grundlagenerkenntnissen mit wissenschaftlich ge-

sicherten Theorien und praktischen Erfahrungen bis hin zum Fingerspitzengefühl. Bei der Beschäftigung mit dem Gebiet des Software Engineerings bieten sich daher zwei grundlegend verschiedene Vorgehensweisen an. In einem betont wissenschaftlichen Vorgehen werden die wichtigsten Methoden und Ansätze des Software Engineerings aufgezählt und wissenschaftlich bewertet. Im Vordergrund steht dann eine kritische Auseinandersetzung mit der Qualität und Relevanz der besprochenen Ansätze. Dies ist die Voraussetzung und Grundlage für weitergehende, wissenschaftliche Arbeiten zum Thema Software Engineering. Eine völlig andere Zielsetzung ist die Erstellung eines praxisorientierten Leitfadens für den Softwareingenieur, bei dem nicht möglichst viele Ansätze alternativ dargestellt, sondern vielmehr eine Folge von ineinander greifender Methoden behandelt werden, die den besten Kompromiss darstellen.

Dieses Buch versucht, einen Mittelweg zu gehen. Zum einen will es die wissenschaftlichen Grundlagen nicht vernachlässigen und die wichtigsten Ansätze darstellen, kritisch bewerten und zueinander in Beziehung setzen. Zum anderen beansprucht es doch einen gewissen Leitfadencharakter, sodass ein Projektmanager eine gewisse Anleitung erfährt, wie er bei der Durchführung eines Softwareprojekts vorgeht. Bewusst sind in den Text Aufgaben eingegliedert, da man die Organisation und das Management von Softwareprojekten nicht nur passiv erlernen kann, sondern selbst entsprechende Entwicklungsteilaufgaben aktiv durchführen muss. Für viele der aufgeführten Aufgaben gibt es dabei aber nicht *die* einzige richtige Lösung, sondern in fast allen Fällen eine Vielzahl von Lösungen, die unterschiedliche Qualitätsprofile aufweisen.

Auffällig für den heutigen Stand der praktischen Softwareentwicklung ist, dass es eine Vielzahl von Softwareprojekten gibt, in denen mittlerweile wohlbekannte, allgemein akzeptierte, grundlegende Prinzipien des Software Engineerings aus Unwissenheit oder Fahrlässigkeit sträflich vernachlässigt oder verletzt werden. Es ist ein Ziel dieses Buchs, solche vermeidbaren Fehler in Softwareprojekten auszuschließen.

Zu betonen ist schließlich, dass dieses Buch natürlich in keiner Weise Erfahrungen in der Softwareentwicklung ersetzen kann. Erst bei der aktiven Durchführung einer Reihe von Softwareprojekten werden viele der angesprochenen Themen dem Leser stärker bewusst.

Danksagung Das vorliegende Buch ist über einen Zeitraum von mehr als 20 Jahren im Rahmen einer mehrfach angebotenen Vorlesungsreihe zum Thema Software Engineering, mit den Schwerpunkten Softwaretechnik sowie Projektorganisation und Management, entstanden. Eingeflossen sind dabei Erfahrungen bei der Durchführung der Vorlesung und den begleitenden Übungen, aus den mehrfach stattgefundenen Softwaretechnikpraktika, in denen 10 bis 20 Studenten über ein Semester größere Softwareprojekte durchgeführt haben und nicht zuletzt aus Erkenntnissen in der Zusammenarbeit mit führenden Wirtschaftsunternehmen zu Themen der Softwareentwicklung.

Vor diesem Hintergrund ist es uns ein Bedürfnis, Studenten, Doktoranden, Mitarbeitern, und Kollegen sowie Gesprächs- und Projektpartnern aus der Industrie für ihre Beiträge und kritischen Anregungen zu danken. Insbesondere gilt unser Dank Prof. Dr. Helmut Krcmar für Input in Form der Folien, die er in seinen Veranstaltungen verwendet hat. Wei-

ter danken wir Dr. Herbert Ehler, PD. Dr. Bernhard Schätz, Veronika Bauer und Manuel Then für anregende Diskussionen und die Unterstützung bei der Erarbeitung der Vorlesungsmanuskripte, welche in dieses Buch mit eingeflossen sind. Weiterhin danken wir unseren stets kritischen Reviewern Dr. Jens Calamé, Dr. Oliver Linssen sowie Prof. Dr. Dr.-Ing. E.h. Ernst Denert und Alfons Demmler. Ohne sie wäre es nicht möglich gewesen, dem vorliegenden Text eine hinreichend tiefe wissenschaftliche Dimension und stark praktische Note zu geben. Nicht vergessen wollen wir auch Hermann Engesser, der uns dieses Buchprojekt ermöglicht hat und, vor allem, Dorothea Glaunsinger für ihre Geduld und Hilfe.

Zu betonen ist schließlich, dass dieses Buch natürlich in keiner Weise praktische Erfahrungen in der Softwareentwicklung ersetzen kann. Erst bei der aktiven Durchführung von Softwareprojekten kann das in diesem Buch beschriebene Wissen und die Methodik nachhaltig verstanden werden. Wir hoffen aber, dass dieses Buch dem Leser dabei nützlich ist, auch in der praktischen Projektdurchführung die eigenen Vorstellungen vom Thema Software Engineering zu festigen und weiter zu entwickeln.

München, Garching, April 2013

Manfred Broy
Marco Kuhrmann

Inhaltsverzeichnis

Teil II Management im Projektlebenszyklus

Teil I
Grundlagen und Begriffsbildung

Grundlagen

1

Zusammenfassung

Software Engineering hat die ingenieurmäßige Entwicklung umfangreicher Softwaresysteme zur Aufgabe. Beim ingenieurmäßigen Vorgehen sind Kosten, Termine und Qualität die zentralen und kritischen Zielgrößen. Software Engineering umfasst zum einen die Softwaretechnik, die sich auf die fachlichen, methodischen und technischen Aufgaben der Softwareentwicklung konzentriert und zum anderen Fragen der Projektplanung, der Projektorganisation und der Projektdurchführung, die unter dem Oberbegriff *Management und Organisation von Softwareprojekten* zusammenfasst werden. Dieses Kapitel beschreibt die wesentlichen Aufgaben der Organisation und des Managements von Softwareprojekten und den Zusammenhang zu den unternehmerischen, fachlichen, methodischen und softwaretechnischen Aufgaben in Softwareprojekten.

1.1 Einleitung

Fragen der Projektorganisation und des Projektmanagements sind nicht nur für Softwareprojekte von Bedeutung, sondern für jegliche Art von Projekten. Lange bevor das *Software Engineering* entstanden ist, haben sich deshalb Ingenieure mit den Themen der Projektorganisation und des Managements auseinandergesetzt. Es gibt viele Probleme und Aufgaben, die unabhängig vom Umstand sind, ob das Projekt die Entwicklung von Software umfassen. Zu betonen ist hierbei, dass das allgemeine Gebiet der Projektorganisation und des Managements und das Gebiet der Softwareentwicklung in einer fruchtbaren, thematischen Wechselwirkung steten. Zum einen hat das Software Engineering viele gute und wichtige Ansätze und Methoden aus dem allgemeinen Gebiet der Projektorganisation und des Managements übernehmen können, zum anderen haben die Besonderheiten von Software zu ganz speziellen Techniken geführt, wie die der sehr sorgfältigen Behandlung von Vorgehensmodellen und allgemein Projekten der Produktentwicklung zu Gute kommen.

M. Broy, M. Kuhrmann, *Projektorganisation und Management im Software Engineering*,
Xpert.press, DOI 10.1007/978-3-642-29290-3_1, © Springer-Verlag Berlin Heidelberg 2013

Das Software Engineering selbst umfasst Vorgehensweisen, Methoden, Beschreibungsmittel, Werkzeuge und Prinzipien für die kostengerechte und die qualitativ angemessene Entwicklung von Softwaresystemen, die in einem organisatorischen oder technischen Rahmen spezifische, an Anwendungen orientierte Aufgabenstellungen zu bewältigen haben. Ziel des *Managements und der Organisation* von Softwareprojekten ist die ökonomische und kostengünstige Entwicklung von Software in geeigneter Qualität.

Themen Das Software Engineering stützt sich auf klassische Gebiete der Informatik und auch auf Aufgabenfelder der Betriebswirtschaftslehre (vgl. zum Beispiel Gadatsch et al. [82]). Die zentralen Inhalte des Software Engineerings können dabei grob in die folgenden zwei Bereiche unterteilt werden:

1. *Fachliches, Technik und Methodik*: Zu den fachlichen, technischen und methodischen Aufgaben des Software Engineerings gehören im Einzelnen: Problemanalyse (Systemanalyse), Anforderungsspezifikation, Systementwurf, Softwarearchitektur, Implementierung, Integration, Qualitätssicherung und Test, Installation, Wartung und Weiterentwicklung.
2. *Organisation und Management von Softwareprojekten*: Dies umfasst: Projektorganisation und Projektmanagement, darunter insbesondere Projekt- und Qualitätsplanung, Kostenschätzung und -kontrolle, Vertragsgestaltung, Vermarktung, sowie die Projekt- und Qualitätsüberwachung und -steuerung.

Beide Aufgabenfelder sind eng miteinander verknüpft und kaum trennbar. Die Vielfalt der Aspekte und deren wechselseitige Abhängigkeiten bedingen einen Großteil der Schwierigkeiten und der *Komplexität* bei der Entwicklung großer Softwaresysteme. Dabei erstrecken sich die Aufgaben nicht nur auf technische Fragen, sondern auch auf die Zusammenarbeit der Projektbeteiligten und -durchführenden in Teams.

Teams und Projekte Aktuelle Softwareprojekte haben längst einen Umfang erreicht, der die Bearbeitung durch nur einen oder einige wenige Entwickler ausschließt. Daher arbeiten in Projekten typischerweise umfangreiche, heterogene Projektteams mit unterschiedlichsten Qualifikationen, Fachwissen und Ausbildungen zusammen an der Erstellung eines Softwaresystems. Da Projekte üblicherweise unter Termin-, Kosten- und Qualitätsdruck stehen, ist die Teamorganisation für die Produktivität und den Projekterfolg mitentscheidend. Nur eine gute und zielgerichtete Organisation von Projektteams hilft, die termin- und sachgerechte Fertigstellung sowie die kosteneffiziente Projektdurchführung sicherzustellen.

Verteilte Teams und Projekte Die Anforderungen an eine tragfähige Projektorganisation steigen in aktuellen Projekten auch nicht nur wegen der steigenden Komplexität und Größe der Projekte, sondern auch aufgrund der zunehmenden räumlichen und organisatorischen Verteilung von Projektteams unter Einbeziehung von externen Dienstleistern rund um den

Globus (zum Beispiel Sangwan et al. [164] oder Välimäki et al. [191]). Die Planung, Organisation und Aufgabenverteilung in global verteilten Projekten ist hierbei eine der schwierigsten Aufgaben. Die Organisation und das Management muss insbesondere in verteilten Projekten nicht mehr „nur" die technischen, methodischen und die teambezogenen Aspekte berücksichtigen, sondern darüber hinaus auch organisatorische Herausforderungen wie zeitverzögertes/zeitversetztes Arbeiten durch unterschiedliche Zeitzonen, Verständigungs- und Sprachprobleme aufgrund unterschiedlicher Kulturen bewältigen [91].

Beobachtungen in der Praxis Die Komplexität und die Heterogenität großer Softwareprojekte implizieren beträchtliche *Risiken*. Nicht selten wird über das Scheitern von bedeutsamen Projekten berichtet, wobei der mißglückte Start der *Ariane 5* 1996 [65] das wohl am meisten zitierte Beispiel ist. Auch andere „leidende" Projekte werden häufig als Beispiele herangezogen, wie zum Beispiel die Deutsche *LKW-Maut* (geplanter Start: 31. August 2003 – tatsächlicher Zeitpunkt vollumfänglicher Funktion: 01. Januar 2006), die Lieferverzögerungen beim *Airbus A380* wegen durch inkompatible Softwareversionen zu kurz berechneter Kabel oder die *Arbeitslosengeld-II-Anwendung*, zu der die *Computerwoche* vom 15. März 2004 (Online) titelte: „Zeitbombe für die Bundesagentur?" Eine „Hitliste" der größten Softwarepannen hat das Unternehmen SQS erstellt [175]. Hier sind unter anderem verzögerte Gehaltszahlungen zu finden oder Bankautomaten, die Geld verschenken. Jedes verzögerte oder fehlgeschlagene Projekt zieht Schäden in Millionen- oder gar Milliardenhöhe nach sich – im schlimmsten Fall sogar Gefahren für von Leib und Leben beteiligter Menschen. Allerdings sind die Ursachen oft nicht technische Gründe, sondern ein Mangel in der „Projektfähigkeit" der Durchführenden. Umfragen und Studien, wie die SUCCESS Studie [41] oder der viel zitierte und kritisierte CHAOS Report [178] zeigen, dass die Missstände eher durch Schwächen in der Organisation der Projekte verursacht werden.

Die oben genannten Projekte sind die *spektakulären* Fehlschläge. Hinzu kommt eine beträchtliche Dunkelziffer von Softwareprojekten mit schmerzhafter Überschreitung von Termin- und Kostenvorgaben oder unzureichender Qualität – gar nicht zu reden von den riesigen Kosten, verursacht durch Fehler und Unzulänglichkeiten von Software im Betrieb. Dabei sei jedoch erwähnt, dass auch Projekte anderer Ingenieurdisziplinen mit Problemen zu kämpfen haben, wie beispielsweise die Elbphilharmonie in Hamburg oder der „Hauptstadtflughafen" Berlin-Brandenburg.

Erkenntnis – Menschliche Faktoren sind dominant
Nicht selten scheitern Softwareprojekte nicht aus technischen sondern aus organisatorischen Gründen. Ungeachtet der geschickten Projektorganisation bleibt der wohl wichtigste und entscheidende Faktor für den Projekterfolg die Kompetenz und Professionalität der *Mitarbeiter* auf allen Ebenen der Hierarchie. Der Auswahl der Mitarbeiter und Führungskräfte (Einstellungspolitik), ihrer Motivation, Expertise, Weiterbildung und Schulung kommt nach DeMarco [58] eine zentrale Bedeutung zu.

1.1.1 Ziele dieses Buchs

Gerade aufgrund der Tatsache, dass an der Entwicklung großer Softwaresysteme umfangreiche Teams beteiligt sind, ergeben sich weitreichende technische und insbesondere organisatorische Anforderungen an die Projektorganisation und das Management. Da die Aufgabenverteilung in großen Projekten, die unterschiedlichen Verantwortlichkeiten, sowie die fachliche und methodische Vielschichtigkeit der Aufgabenstellung bedingen, dass kein Projektverantwortlicher alle Details eines Projekts in vollem Umfang erfassen kann, ist es unerlässlich, den Projektverantwortlichen Methoden an die Hand zu geben, um den Projektstatus hinsichtlich Kosten, Qualität und Terminsicherheit zutreffend einzuschätzen und planerisch sowie steuernd darauf zu reagieren. Dies erfordert, dass aufgrund wissenschaftlich gesicherter Ansätze allgemeine Maßstäbe und aussagekräftige Metriken zu entwickeln sind, die es dem Management erlauben, den Status eines Projekts zu beurteilen und zu steuern.

Ein zentrales Bestreben ist die Entwicklung von Theorien und Methoden, die eine objektive Einschätzung der wesentlichen Bestandteile eines Projekts im Hinblick auf die Qualität und Fertigstellung der Ergebnisse aber auch auf die Qualität der Methoden, Prozesse und der beteiligten Mitarbeiter ermöglichen. Dies kennzeichnet eines der langfristigen Ziele der Disziplin *Projektorganisation und Management im Software Engineering*.

Mit diesem Buch fassen wir die wesentlichen Aspekte der Organisation und des Managements von Projekten als Teils des Software Engineerings zusammen. Dieses Buch gibt im Speziellen einen Einstieg in die für Softwareprojekte besonders relevanten Themen:

- Vorgehensmodelle im Software Engineering
- Unternehmens- und Projektorganisation
- Managementaufgaben im Lebenszyklus eines Softwareprojekts

Alle ausgewählten Themen werden beispielhaft durch Methoden, Erfahrungen, Anwendungsbeispiele und Übungsaufgaben flankiert. Ziel dieses Buchs ist es, das Thema *Projektorganisation und Management* für Informatiker und Softwareingenieure einzuführen und eine Hilfe für die Organisation und die Durchführung von Softwareprojekten zu geben.

1.1.2 Für wen ist dieses Buch?

Das vorliegende Buch ist als *Lehrbuch* konzipiert. An der Technischen Universität München wird am Lehrstuhl für Software & Systems Engineering seit vielen Jahren die Vorle-

sung „Projektorganisation und Management in der Softwareentwicklung“ angeboten, welche die Vorlage für dieses Buch liefert. Das Buch richtet sich an Studierende in Bachelor- und Master-/Diplom-Studiengängen der Informatik, Wirtschaftsinformatik und verwandter Disziplinen.

Dieses Buch ist jedoch auch eine Unterstützung für den Praktiker, insbesondere für Berufs- und Quereinsteiger, um Softwareprojekte zu organisieren und durchzuführen.

1.1.3 Erforderliches Vorwissen

Themen der Anforderungs- und Systemanalyse, des Architekturentwurfs, der Codierung oder des Testens, sind *nicht* Gegenstand dieses Buchs. Wir setzen Grundkenntnisse in dem oben genannten „handwerklichen“ Bereich der Technik und Methodik der Softwareentwicklung voraus. Ebenso ist es hilfreich, bereits grundlegende Kenntnisse der Betriebswirtschaftslehre und erste Projekterfahrung mitzubringen.

1.1.4 Aufbau des Buchs

Dieses Buch besteht aus vier Teilen (Abb. 1.1). Im Teil 1 werden die Grundlagen behandelt und die Begriffsbildung vorgenommen. In diesem Rahmen werden die *Unternehmens- und Projektorganisation*, der *Lebenszyklus von Softwareprojekten* und *Vorgehensmodelle im Software Engineering* beschrieben. Im Teil 2 werden die Organisation und das Management in den einzelnen Abschnitten des Projektlebenszyklus dargestellt. Neben den übergreifenden Aufgaben des Managements werden die einzelnen Aufgaben während der *Projektentstehung*, der *Projektdefinition*, der *Projektdurchführung* und im *Projektabschluss* im Detail behandelt. Teil 3 des Buchs fasst die weiterführenden Themen *Metriken und Messung*, *Reifegradmodelle und Prozessverbesserung* und *Werkzeuge* zusammen. Im Anhang finden sich neben Ergänzungen und Erläuterungen zu COCOMO/COCOMO II und CMMI auch die Unterlagen für das Beispielprojekt „Code & Talk“, welches insbesondere in den Übungsaufgaben verwendet wird. Die Kapitel, die Übungsaufgaben enthalten, sind in Abb. 1.1 mit einem „Ü“ gekennzeichnet.

Dieses Buch befasst sich mit der Organisation und dem Management von Softwareprojekten. Es steht somit in Beziehung zur etablierten Literatur des Projektmanagements, wie etwa Patzak und Rattay [152], Burghardt [40] oder Hindel et al. [90]. Es beschreibt keine in sich geschlossene Vorgehensweise für das Projektmanagement, wie beispielsweise PRINCE2 [147] oder PMBOK [158]. Das vorliegende Buch ist *kein* Projektmanagementkompendium. Es baut vielmehr einen systematischen Rahmen auf, in dem wesentliche Themen zur Projektorganisation und zum Management vermittelt werden können. Alle Inhalte zum Projektmanagement können zu in der Praxis eingesetzten Standards in Beziehung gesetzt werden. Tabelle 1.1 zeigt das am Beispiel des PMBOK.

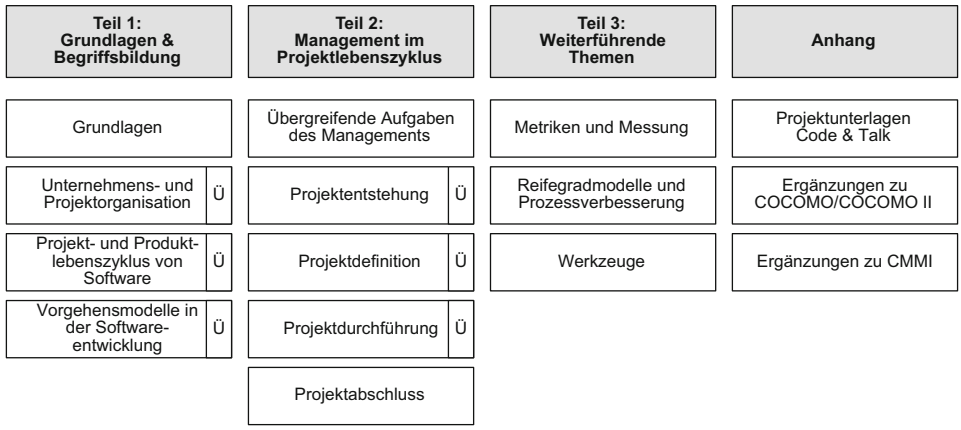

Abb. 1.1 Aufbau und Kapitelstruktur des Buchs

Tab. 1.1 Abbildung zum/Einordnung in das PMBOK

PMBOK	Buch
Initializing	Kapitel 6 Projektentstehung
Planning	Kapitel 7 Projektdefinition
Executing & Controlling	Kapitel 8 Projektdurchführung
Closing	Kapitel 9 Projektabschluss

1.2 Management von Softwareprojekten

Die Projektorganisation und das Management von Softwareprojekten verfolgen im Kern wirtschaftliche Ziele. Insbesondere die Optimierung des ökonomischen Kosten-/Nutzenverhältnisses in Softwareprojekten sind wesentliche Anliegen, die zwangsläufig Markt- und Qualitätsbetrachtungen einschließen.

Aus dem Spannungsfeld zwischen wirtschaftlichen und technischen Gesichtspunkten leiten sich die unterschiedlichen Ziele und Aufgaben des Managements im Software Engineering ab, aber auch klare Unterschiede zu Projekten, die nicht die Entwicklung von Software zum Gegenstand haben.

1.2.1 Grundlegende Begriffe

Zu Beginn dieses Abschnitts werden grundlegende Begriffe eingeführt:

▸ **Definition 1.1 (Organisation)** Dem Begriff Organisation (griechisch: órganon – „Werkzeug“) werden drei Bedeutungen zugeschrieben: Struktur, Funktion und Institution:

Struktur Die strukturelle Organisation steht für die organisatorische Gliederung von Institutionen (in Bereiche, Abteilungen, etc.) und Arbeitsgruppen (Aufbauorganisation) und von Tätigkeiten (Ablauforganisation).

Funktion Die funktionale Organisation steht für den Prozess des Organisierens, durch den fortlaufende unabhängige Handlungen zu sinnvollen Folgen zusammengefügt werden, so dass vernünftige Ergebnisse erzielt werden.

Institution Eine institutionelle Organisation ist ein soziales oder rechtliches Gebilde (etwa ein Unternehmen), das aus dem planmäßigen und zielorientierten Zusammenwirken von Menschen entsteht, sich zur Umwelt abgrenzt und – als kooperativer Akteur – mit anderen Akteuren interagieren kann.

Wir verwenden im Weiteren den Begriff Organisation im Wesentlichen im Sinne der Strukturierung von Unternehmen und Projekten.

▸ **Definition 1.2 (Management)** Management (engl. to manage → italienisch: maneggiare, „handhaben“) kann sowohl *Leitungsaufgaben* in Projekten und Unternehmen bezeichnen, als auch die *Gruppe der Personen*, die diese Aufgaben ausüben und entsprechende Managementkompetenzen benötigen. Typische Aufgaben des Managements sind: Planung, Delegation, Organisation, Führung und Kontrolle (im Sinne von Fortschritts- und Erfolgskontrolle).

Die Gesamtheit aller Tätigkeiten, die mit der erfolgreichen Abwicklung eines Projektes zusammenhängen, mündet im sogenannten *Regelkreis des Projektmanagements* (Abb. 1.2) zur Überwachung und Steuerung von Projekten. Dieser Regelkreis wird in einem Projekt kontinuierlich durchlaufen.

In der *Planung* werden die Soll-Vorgaben für die Projektdurchführung festgelegt. In der Projektdurchführung werden Kennzahlen ermittelt, welche als Eingabe für die *Überwachung* des Projekts dienen. Auf Basis der Kennzahlen werden zum Beispiel Soll-Ist-Vergleiche durchgeführt, um den aktuellen Projektstatus zu ermitteln. Auf Abweichungen oder sonstige Ereignisse wird reagiert, indem in der *Steuerung* Maßnahmen festgelegt und in Gang gesetzt werden. Diese haben in der Regel Auswirkungen auf die Planung des Projekts, welche entsprechend anzupassen ist. Im Berichtswesen werden abschließend alle In-

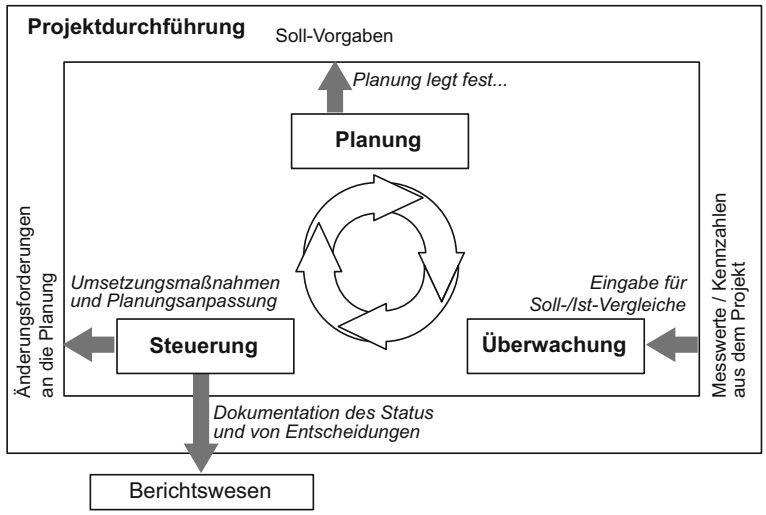

Abb. 1.2 Regelkreis des Projektmanagements

formationen zusammengetragen und in einer für alle Stakeholder nachvollziehbaren Weise dokumentiert.

▸ **Definition 1.3 (Projekt, ISO)** Ein Projekt[1] ist nach [64][2] ein Vorhaben, bei dem innerhalb einer vorgegebenen Zeitspanne ein spezifiziertes Ziel erreicht werden soll und das sich dadurch auszeichnet, dass es im Wesentlichen ein einmaliges (individuelles) zeitlich begrenztes Vorhaben ist.

▸ **Definition 1.4 (Projekt, PRINCE2)** Ein Projekt ist nach [147] eine für einen befristeten Zeitraum geschaffene Organisation, die mit dem Zweck eingerichtet wurde, eine oder mehrere Erzeugnisse in Übereinstimmung mit einem vereinbarten Business Case zu liefern. Es zeichnet sich darüber hinaus durch eine fortlaufende wirtschaftliche Rechtfertigung aus.

▸ **Definition 1.5 (Projekt, PMBOK)** Ein Projekt ist nach [158] eine zeitlich beschränkte Anstrengung zur Erzeugung eines einmaligen Erzeugnisses (Produktes) oder Dienstes.

[1] Projekt in der allgemeinen Bedeutung eines besonderen Vorhabens geht auf das lateinische Verb proicere: „nach vorn werfen“, „projizieren“ zurück.
[2] Die Serie DIN 6990x bezieht sich auf eine Reihe von Normen zum Thema *Projektmanagment*. Wir beziehen uns in der Regel auf die Norm 69901, weisen aber explizit darauf hin, dass Begriffe auch aus anderen Teilen der Norm entstammen können.

Arbeitsdefinition
Verkürzt und vereinfacht ausgedrückt lässt sich ein Projekt wie folgt charakterisieren:

- Ein Projekt zielt auf die Lösung eines eigenständigen Problems.
- Ein Projekt zielt auf die Erarbeitung eines Ergebnisses.
- Ein Projekt hat einen Anfang und ein Ende – es ist zeitlich begrenzt.
- Ein Projekt hat ein klares Ziel.
- Ein Projekt ist ein einmaliges Vorhaben.
- Ein Projekt grenzt sich von anderen Vorhaben ab.
- Einem Projekt steht eine begrenzte Menge von Ressourcen zur Verfügung.
- Ein Projekt benötigt einen organisatorischen Rahmen, der alle Aufgaben unterstützt, um das Projektziel zu erreichen.

Mithilfe dieser Charakteristika umschreiben wir den Projektbegriff, den wir diesem Buch zugrunde legen.

Ein Projekt fasst somit verschiedene Aspekte, wie Ressourcen, Ergebnisse oder Rollen zusammen. Für die Einordnung in die Inhalte und das weitere Verständnis dieses Buchs, dient der Ordnungsrahmen aus Abb. 1.3. Die hier aufgeführten und in Beziehung gesetzten Begriffe spiegeln die grundlegende Terminologie wider, die im Weiteren im Detail erläutert werden.

Projekte zielen auf die Erarbeitung von Ergebnissen. Diese Ergebnisse sind in der Regel der primäre Grund für die Durchführung eines Projekts. Je nach Projekt können diese Ergebnisse sehr unterschiedlich sein. Beispiele sind

- ein neu entwickeltes, lauffähiges Softwaresystem
- Teilergebnisse für die Entwicklung von Software (Anforderungen, Architekturentwurf, Testkonzept, etc.)
- die Migration einer Software auf eine andere Hardwareplattform

Typischerweise erarbeiten Projekte primäre Ergebnisse. Dies sind die Resultate des Projekts, welche der Grund für die Durchführung eines Projekts sind. Daneben werden in Projekten noch eine ganze Reihe von sekundären Teilergebnissen erarbeitet, die Mittel zum Zweck und auf dem Weg der Erarbeitung der primären Ergebnisse nützlich sind. In beiden Fällen sprechen wir von *Artefakten*, die wir in Anlehnung an Méndez et al. [138] wie folgt definieren.

▸ **Definition 1.6 (Artefakt)** Ein Artefakt ist ein primäres oder sekundäres Arbeitsergebnis, das in einem Projekt durch bestimmte Projektaktivitäten erstellt, bearbeitet oder genutzt

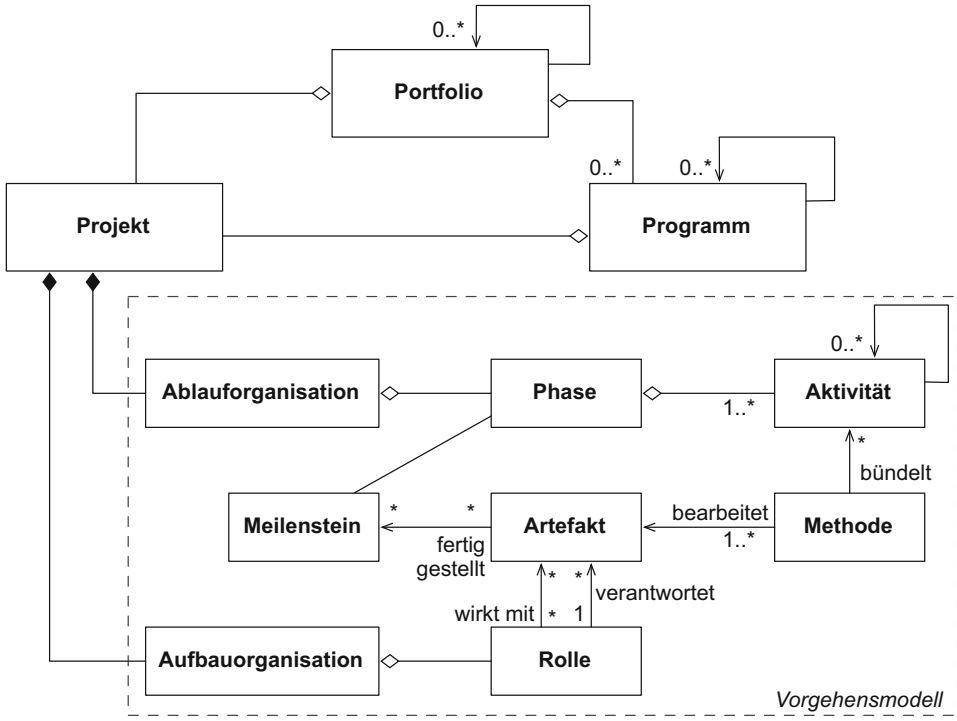

Abb. 1.3 Ordnungsrahmen (Ontologie) für Projekte in Form eines Metamodells

wird. Rollen sind verantwortlich für die Erstellung von Artefakten. Artefakte sind Gegenstand der Qualitätssicherung, der Versionskontrolle, haben einen Typ, einen Inhalt und eine innere Struktur.

Artefakte sind für uns äußerst wertvolle Konzepte, da sie es erlauben, über konkrete Ergebnistypen und Ergebnisse zu sprechen, ohne auf die Details ihrer Erstellung, etwa Methoden oder konkrete Aktivitäten, eingehen zu müssen. Beispiele für Artefakte sind Ausarbeitungen (Anforderungsdokumentation, Architekturentwurf, Testfallbeschreibung, Programme, etc.) aber auch Dokumente wie Verträge, Protokolle etc. Ein Artefakt kann durch folgende Aspekte gekennzeichnet werden:

- sein Inhalt, Zweck
- durch wen und durch welche Aktivitäten es geschaffen wird
- durch wen und in welchen Aktivitäten es genutzt wird
- wie es repräsentiert wird (Papier, Datei, Inhalt einer Datenbank)
- wie es strukturiert ist (Gliederung)
- mit welchen Mitteln sein Inhalt dargestellt wird (Text, Formel, Diagramm)

Die Menge aller Artefakte eines Projekts beschreibt alle primären und sekundären Ergebnisse des Projekts. Wesentlichen Projektergebnisse sollten sich grundsätzlich in Artefakten (und seien es nur Aktennotizen) niederschlagen.

1.2.1.1 Unterschiede zu Projekten anderer Ingenieurdisziplinen

Dieses Buch konzentriert sich auf Projekte, die im weiten Sinn auf Themen der Softwareentwicklung ausgerichtet sind. Die Durchführung solcher Projekte fällt in die Disziplin „Software Engineering".

Software Engineering ist eine Ingenieurdisziplin
Das Software Engineering weist mit dem Ziel der Entwicklung nutzbarer Produkte und Systeme alle Merkmale einer *Ingenieurdisziplin* auf, nämlich der Erreichung formalisierter technischer Zielsetzungen unter Einhaltung technischer und ökonomischer Rahmenbedingungen.

Viele Aufgaben der Projektorganisation und des Managements von Softwareprojekten unterscheiden sich nicht grundlegend von denen in Entwicklungsprojekten, die nicht Software, sondern Maschinen, elektrotechnische Erzeugnisse, Bauwerke oder auch allgemein Produkte zum Ziel haben. Allerdings gibt es eine Reihe *signifikanter Unterschiede*. Der wahrscheinlich wichtigste betrifft die *Immaterialität* von Software. Die Folgen dieses wesentlichen Unterschieds sind:

1. *Es gibt nur den Entwicklungsprozess und die Inbetriebnahme* und keinen wirklichen Produktions- bzw. Fertigungsprozess.
2. Es gibt Unterschiede in den *Kostenstrukturen*.
3. *Vertriebs- und Wartungsstrukturen* unterscheiden sich deutlich.
4. Software greift tief in menschliche, betriebswirtschaftliche und technische Prozesse ein und erfordert ein umfassendes Verständnis der Anwendungsfelder und der Kundenbedürfnisse.
5. Es gibt *zusätzliche Chancen und Risiken* durch zeitige Kundeneinbindung und Änderung von Kundenanforderungen während der Entwicklung.
6. *Software ist ein abstraktes Artefakt*. Die Erarbeitung und Analyse von Kundenanforderungen ist oft nur iterativ (manchmal auch nur experimentell) möglich.
7. Software beschreibt *dynamisches Verhalten* und unterliegt einer ständigen *Evolution*, womit sie eine inhärent hohe Komplexität aufweist.

Folgen Software wird entwickelt, nicht produziert. Dass kein Fertigungsprozess wie bei allen materiellen Produkten erforderlich ist, führt zu einer völlig anderen Organisations- und Kostenstruktur. Insbesondere ist ein unterschiedlicher Informations- und Erkenntnisfluss

die Folge. Werden bei materiellen Produkten Fehler in der Entwicklung unter Umständen noch in der Planung und Durchführung der Fertigung erkannt, fehlt diese zusätzliche Überprüfung in der Softwareentwicklung gänzlich.

Hinweis
Der Begriff Produkt wird in unterschiedlicher Weise verwendet:

- Statt von Artefakten wird auch bedeutungsgleich von Produkten gesprochen – dann bezeichnet ein Produkt ein primäres oder sekundäres Arbeitsergebnis eines Projekts.
- Oft wird das Gesamtergebnis eines Projekts als Produkt bezeichnet.
- Bei vertriebenen Dienstleistungen oder Waren spricht man ebenfalls von Produkten. So gibt es Software, die als Massenprodukt vertrieben wird.

Im Sinne der letzten Definition ist ein Produkt ein Wirtschaftsgut.

Software ist aufgrund ihrer Immaterialität auch wenig anschaulich – sie ist ein *abstraktes Artefakt*. Zudem sind die Erfahrungswerte in der Softwareentwicklung noch deutlich eingeschränkter als in den klassischen Ingenieurdisziplinen mit ihren langen Traditionen und Erfahrungen. Erschwerend kommt hinzu, dass im Softwarebereich immer noch in schneller Folge technische Innovationen erfolgen. Innovationen ziehen oft auch ein breites Spektrum von technischen und methodischen Änderungen im Entwicklungsprozess nach sich. Aus diesen Gründen ist auch bei den für die Projektorganisation Verantwortlichen solides Wissen der Regeln und Prinzipien der Softwareentwicklung unabdingbar. Anders als in Gebieten, in denen gesicherte Erfahrungsdaten vorliegen und klare Regeln und Faustformeln zur Projektplanung und Durchführung vorhanden sind, erfordern Softwareprojekte von Managern fundierte Kenntnisse über die Besonderheiten der Softwareentwicklung.

▸ **Anmerkung** Softwareprojekte sind weniger als Projekte anderer Art nach rein wirtschaftlich orientierten und messbaren Methoden plan- und steuerbar. Dort wird nach bewährten Regeln vorgegangen, wohingegen es in der Softwareentwicklung nur eingeschränkt gesicherte Erfahrungen gibt.

1.2.2 Ziele des Managements im Software Engineering

Ziel des Software Engineering ist die *kostengünstige* und *termingerechte* Erstellung *zuverlässiger* und *nutzergerechter* Softwaresysteme in *angemessener Qualität* zur Lösung anwendungsspezifischer Aufgabenstellungen unter vorgegebenen Rahmenbedingungen. Daraus leiten sich die folgenden Teilziele ab:

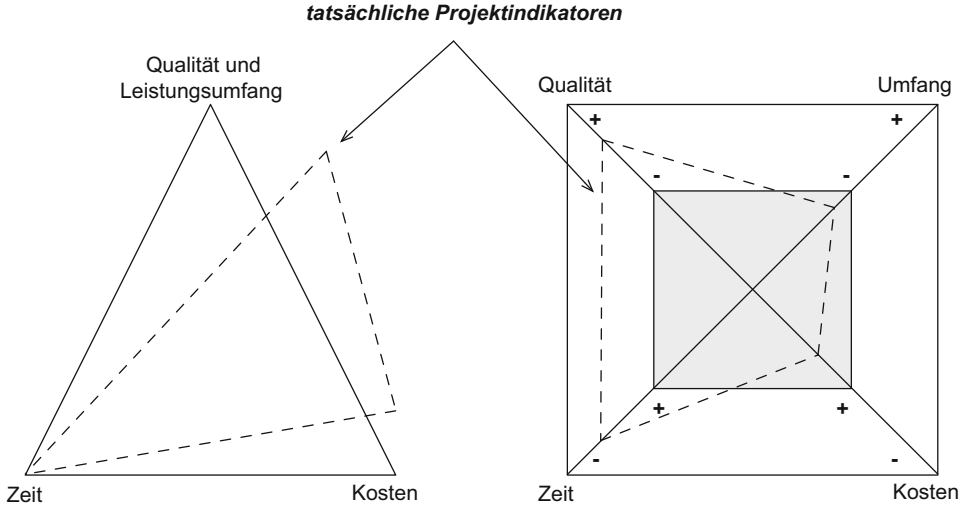

Abb. 1.4 Das „Magische Dreieck des Projektmanagements“ und das „Teufelsquadrat“ nach H. Sneed. Beide Darstellungen illustrieren die wechselseitigen Abhängigkeiten zwischen Zeit, Umfang, Qualität und Kosten.

Beherrschung von Zeit und Kosten Unter den oben genannten Besonderheiten von Softwareprojekten ist die Beherrschung der Kosten eine der größten Herausforderungen. Alle Parameter des sogenannten „Magischen Dreiecks“ bzw. des „Teufelsquadrats“ nach Sneed [173] (Abb. 1.4) wirken direkt oder indirekt auf die Projektkosten. Die Beherrschung der Projektkosten hängt hierbei wesentlich von der Zuverlässigkeit der Kostenschätzung unter Berücksichtigung der Schätzungen für Entwicklungsaufwand und -dauer ab. Die Kostenstruktur muss konkurrenzfähig sein, damit sich ein Unternehmen im wirtschaftlichen Wettbewerb am Markt behaupten kann. Hierzu ist es unterlässlich, die Einhaltung der geschätzten Kosten, Termine und Aufwände kontinuierlich zu kontrollieren.

Abbildung 1.4 (rechts) beschreibt das Teufelsquadrat der Projektgrößen Zeit, Qualität, Umfang und Kosten (Aufwand). Grob erklärt gilt: Die Fläche des gestrichelten Vierecks ist immer gleich groß; wählt man drei Größen, ergibt sich die vierte daraus. Weniger Zeit wirkt sich kritisch auf die Höhe der Kosten, die Qualität und den Leistungsumfang aus.

Erreichung von Qualitätszielen Neben Zeiten und Kosten ist die Sicherstellung einer möglichst hohen Qualität der Software ein wesentliches Ziel. Das Management muss die Erreichung von Qualitätszielen in der Regel aus den drei Perspektiven der Nutzer, der Entwickler und des Betriebs betrachten (zum Beispiel nach der Norm ISO/IEC 25000 [100], dem Nachfolger der ISO 9126 [99]). Hierbei sind aus Nutzersicht zum Beispiel die Anforderungen im Hinblick auf Angemessenheit und der Umsetzung der Funktionalität, der Effizienz, Performanz, Korrektheit und Zuverlässigkeit zu erfüllen. Aus der Perspektive des