Peter **HRUSCHKA**

3. Auflage

# BUSINESS ANALYSIS UND REQUIREMENTS ENGINEERING

## Produkte und Prozesse nachhaltig verbessern

**Ihr Weg zur Requirements-Zertifizierung**

HANSER

Hruschka

# Business Analysis und Requirements Engineering

Peter Hruschka

# Business Analysis und Requirements Engineering

## Produkte und Prozesse nachhaltig verbessern

3., aktualisierte Auflage

HANSER

Der Autor:
*Dr. Peter Hruschka*, Aachen



Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.d-nb.de* abrufbar.


Lektorat: Brigitte Bauer-Schiewek
Copy editing: Petra Kienle, Fürstenfeldbruck
Umschlagdesign: Marc Müller-Bremer, *https://www.rebranding.de*, München
Umschlagrealisation: Max Kostopoulos
Titelmotiv: © istockphoto.com/3d_kot
Layout: Manuela Treindl, Fürth
Druck und Bindung: Hubert & Co. GmbH & Co. KG BuchPartner, Göttingen
Printed in Germany

Print-ISBN: 978-3-446-47692-9
E-Book-ISBN: 978-3-446-47819-0
E-Pub-ISBN: 978-3-446-47948-7

# Inhalt

# Vorwort

Als ich 1976 direkt nach dem Studium meinen ersten Arbeitsvertrag in Händen hielt, war ich überrascht. Als Berufsbezeichnung war „Systemanalytiker“ eingetragen. Ich war deshalb überrascht, weil ich „Programmierer“ erwartet hatte. Denn was wir an der Technischen Universität Wien im Informatikstudium gelernt hatten, war Programmieren, formale Sprachen, Mathematik und Logik, Hardware- und Elektrotechnikgrundlagen und vieles andere, aber keine Systemanalyse. Die Berufsbezeichnung „Systemanalytiker“ klang jedoch in meinen Ohren gut und ich fühlte mich aufgewertet. Ich habe mir jedoch vorgenommen, im Lauf meines Berufslebens herauszubekommen, was ein Systemanalytiker so tut.

In der Zwischenzeit haben wir die Berufsbezeichnung „Systemanalytiker“ oft durch „Business Analyst“ oder durch „Requirements Engineer“ ersetzt. Im Kern geht es immer noch um das gleiche Thema: herauszubekommen, wo unsere heutigen Produkte und Prozesse Schwachstellen haben, was Kunden und Nutzer wirklich brauchen, wie Prozesse durch neue Ideen effektiver gemacht werden können oder was IT-Systeme wirklich leisten sollten, um das Business besser zu unterstützen. Und das, was wir herausbekommen haben, wollen wir effektiv miteinander besprechen und verhandeln können, bevor wir es in Lösungen umsetzen (lassen), Produkte erstellen oder IT-Systeme bauen.

Jedem Vorhaben oder Projekt, d. h. jeder Korrektur, Verbesserung, Erweiterung, Änderung oder Diversifizierung Ihrer Produkte oder Ihrer betrieblichen Prozesse, sollten ein klares Verständnis des Ist-Zustands, eine gründliche Analyse von Schwachstellen und Risiken und die Identifizierung von Verbesserungsoptionen vorausgehen.

Das ist das Thema dieses Buchs. Sie können es gerne Business Analysis, Requirements Engineering oder auch Systemanalyse nennen. Das sind nur verschiedene „Jahrgänge“ von Worten, die das gleiche Thema behandeln: analysieren und artikulieren, welche Anforderungen wir haben, um Produkte und Prozesse innovativ und nachhaltig zu verbessern.

Wie Sie alle bin auch ich beruflich und privat Nutzer von sehr vielen Softwaresystemen und Produkten. Das beginnt beim Mobiltelefon, bei den Programmen auf meinem Rechner, mit denen ich Kursfolien gestalte oder Bücher und Artikel schreibe; die Systeme, die die Abrechnungen erstellen, die ich jeden Monat erhalte und bezahlen muss, aber auch viele andere technische Systeme im Auto, mit denen ich täglich unterwegs bin, oder in meinem Kaffeevollautomat, um mich mit verschiedenen Spezialitäten bei Laune zu halten. Mit diesem Buch möchte ich ganz einfach meinen Beitrag dazu leisten, dass diese Systeme und Produkte den Benutzern mehr Freude als Ärger bereiten.

Aachen, Januar 2014 *Peter Hruschka*

# Vorwort zur dritten Auflage

Mehr als acht Jahre sind vergangen seit der ersten Auflage. In weiten Teilen der Industrie und in vielen Behörden ist man inzwischen auf dem Weg zu agiler Business Analysis und agilem Requirements Engineering. Die Pandemie und der Trend zu Home-Office hat dafür gesorgt, dass auch in agilen Vorhaben die Anforderungen statt mit Kärtchen an der Wand elektronisch verwaltet werden – und die Kooperationswerkzeuge sind viel besser geworden.

Trotzdem gibt es sie natürlich noch: die klassischen Projekte – oft mit Ausschreibungen oder der Vergabe von kompletten Aufträgen an Offshore-Firmen, in denen die Auftraggeber noch sehr sorgfältig frühzeitig versuchen, die Anforderungen möglichst komplett aufzuschreiben.

Was mich persönlich sehr freut: Auch in der agilen Szene hat man inzwischen erkannt, dass Qualitätsanforderungen mindestens genauso wichtig sind wie die funktionalen Anforderungen, und man räumt ihnen in Veröffentlichungen, Schulungen und in den Werkzeugen den ihnen gebührenden Raum ein.

Vieles ist aber seit vielen Jahrzehnten einfach immer noch wahr und wichtig. Noch immer wollen wir als Analytiker die Probleme und Geschäftsideen verstehen und kommunizieren, bevor wir an Lösungen gehen. Nur ist unsere Welt immer schnelllebiger geworden: Wir wollen die Lösungen schneller und öfter. Wir arbeiten eher im Wochen- oder Monatsrhythmus statt in Jahren. Daran müssen sich auch die Analyseprozesse anpassen.

Eine meiner Hauptbotschaften, die sich im Titel „Business Analysis und Requirements Engineering“ ausdrückt, ist immer noch nicht ganz in der täglichen Praxis angekommen: die möglichst reibungslose Zusammenarbeit zwischen Fachabteilungen und IT-Abteilungen, zwischen Auftraggebern und Auftragnehmern, zwischen denen, die das Business verbessern wollen, und denen, die IT-Systeme dafür entwickeln und liefern können.

Aber ich gebe die Hoffnung nicht auf, die Grenze zwischen Firmen oder Abteilungen weiter abzubauen: Denn egal, auf welcher Seite Sie angesiedelt sind, wir ziehen doch (hoffentlich) am gleichen Strang (und auch am gleichen Ende des Strangs) und wollen, dass IT-Systeme unser Leben immer weiter erleichtern.

Aachen, März 2023 *Peter Hruschka*

# 1 Probleme, Ziele, Ideen und Visionen

Wenn man ein IT-Vorhaben aufsetzt, dann hat man entweder im heutigen geschäftlichen Alltag ein Problem oder eine Schwachstelle identifiziert, die man beseitigen möchte, oder man hat eine Idee oder eine Vision, wie ein Produkt oder ein Prozess etwas besser, schneller, effizienter, billiger etc. werden könnte. Nun ja, manchmal hat man vielleicht auch einfach noch Geld übrig und möchte es für irgendetwas ausgeben. Das alles sind Ausgangspunkte für Business Analysis und Requirements Engineering.

Im ersten Teil wollen wir uns mit einigen Grundbegriffen und einer Einführung in das Thema Business Analysis und Requirements Engineering auseinandersetzen. Um Sie zu motivieren, betrachten wir einige quantitative und qualitative Gründe für den Einsatz derartiger Methoden. Wir erwähnen aber auch Ursachen, warum es trotz aller guten Gründe manchmal nicht ernsthaft betrieben wird und was die Folgen von diesem Versäumnis sein können.

Danach werden wir die drei Säulen erfolgreicher Vorhaben betrachten, von denen Business Analysis und Requirements Engineering eine ist.

Wir besprechen Definitionen: Was sind überhaupt Requirements oder auf Deutsch, was sind Anforderungen? Und was bedeutet Business Analysis und was Requirements Engineering? Was gehört dazu, was gehört nicht dazu?

Dann diskutieren wir die Berufsbilder Business Analyst, Requirements Engineer und Product Owner. Was haben diese Personen zu tun? Was müssen sie machen? Aber auch: Welche Fähigkeiten sollten sie mitbringen, welche Soft-Skills und welche methodischen Fähigkeiten?

Anschließend besprechen wir die drei wichtigen Arten von Anforderungen. Und zum Schluss des Kapitels betrachten wir verschiedene Vorgehensweisen, wie man unter unterschiedlichen Randbedingungen zu Anforderungen kommen kann – vom Wasserfallmodell bis zu sehr agilen Vorgehensweisen.

## 1.1 Wovon sprechen wir?

In immer stärkerem Maß unterstützen heute technische Produkte oder IT-Systeme unser Leben. Sie helfen uns, Ziele schneller und effektiver zu erreichen, erleichtern oft vormals manuelle Tätigkeiten oder ermöglichen Dinge, von denen wir vor einiger Zeit nicht einmal zu träumen gewagt haben. Wir nehmen Geldtransaktionen von zuhause aus vor, wir überlassen das Schalten der Gänge und die Traktionskontrolle unseren Autos, wir telefonieren und

surfen mobil fast überall, wir nutzen medizinische Geräte zur Verbesserung der Diagnostik, wir gestalten Kundenprozesse benutzerfreundlicher und effektiver ...

Dazu ist es natürlich notwendig, unsere Probleme, Wünsche, Träume und Visionen so zu durchleuchten und aufzubereiten, dass IT-Entwickler und Produktentwickler dafür geeignete Lösungen erstellen können oder bestehende Produkte und Prozesse nachhaltig verbessern können.

Ohne klare Kenntnis des Ist-Zustands und ohne klare Ideen, was wir erreichen wollen, verzetteln wir uns sonst in Lösungen, die an den Marktbedürfnissen oder den Bedürfnissen einer Organisation vorbeigehen und mühevoll und aufwendig nachgebessert werden müssen. Deshalb brauchen wir Methoden und Techniken, um herauszubekommen, was das Geschäft wirklich braucht oder was Produkte leisten sollen. Diese Tätigkeiten nennen wir Business Analysis oder Requirements Engineering.

## 1.2 Quantitative Gründe

Wir können sicherlich sehr viele quantitative und qualitative Argumente anführen, warum wir uns mit dem Thema auseinandersetzen sollen. Sprechen wir zuerst mal über die Manager. Die wollen immer Zahlen hören. Sehr bekannt ist die Studie der Standish-Group über Fehler in Softwareprojekten. Software, die nie fertig wird, die viel zu teuer ist, die zu spät ausgeliefert wird und die weit überteuert ist. Oder sogar total scheitert. In dieser Studie, die jährlich erneuert wird, hat man die Top-10-Risiken, die zehn größten Risiken, in IT-Vorhaben festgehalten. In Bild 1.1 können Sie sehen, dass sich vier von diesen Top-10-Risiken mit dem Thema unseres Buchs beschäftigen, mit den Anforderungen.

| | | |
|---|---|---|
| | Executive Support | 18 |
| 1. | User Involvement | 16 |
| | Experienced Project Manager | 14 |
| 2. | Clear Business Objectives | 12 |
| 3. | Minimized Scope | 10 |
| | Standard Software Infrastructure | 8 |
| 4. | Firm Basic Requirements | 6 |
| | Formal Methodology | 6 |
| | Reliable Estimates | 5 |
| | Other | 5 |

**Bild 1.1**
Die Top-10-Risiken in IT-Vorhaben gemäß Standish-Studie

Erstens die wirkliche Beteiligung der Benutzer, die das System haben wollen, an dem Prozess (heute eher Stakeholder Involvement genannt – mehr dazu in Abschnitt 2.4); zweitens klare geschäftliche Zielsetzungen – heute oft auch Vision genannt; drittens eine eindeutige Festlegung des Scope und viertens eine vernünftige Formulierung der Anforderungen. Die Zahlen hinten den Punkten sind Gewichtungen. Zusammengenommen machen diese vier Themen 44 von 100 Punkten aus, fast den halben Projekterfolg!

Aber ich glaube, noch deutlicher hat es der Großmeister der Zahlen, Capers Jones, ausgedrückt. Er hat auch untersucht, wie viele Fehler Teams machen, die keine guten Analysemethoden,

keine guten Requirements-Methoden oder andere Qualitätsmethoden für Anforderungen haben [Jon08]. Sein Ergebnis: Teams machen 0,23 Fehler pro Function Point. Sie brauchen jetzt nicht genau zu verstehen, was ein Function Point ist. Vergleichen Sie den Wert nur mit dem Ergebnis beim Einsatz guter Analysemethoden.

Mit guten Requirements-Methoden kann man die Fehlerrate von 0,23 auf 0,08 pro Function Point reduzieren. Das ist ein Drittel davon. Dreimal weniger Fehler. Und sicherlich kennen die meisten von Ihnen die Statistiken, was es uns kostet, Fehler erst im Betrieb eines Produkts oder eines Systems zu korrigieren. Es ist aufwendiger, es ist viel teurer, als wenn wir Fehler zu dem Zeitpunkt finden, wo sie gemacht werden, nämlich beim Beschreiben des Problems. Manche Chefs fangen an zuzuhören, wenn man sagt, dass drei Mal weniger Fehler gemacht werden und drei Mal weniger Kosten hinterher für die Korrektur dieser Fehler aufgewendet werden müssen. Das sind schlagende betriebswirtschaftliche Argumente für den Einsatz guter Analysemethoden.

## 1.3 Qualitative Gründe

Aber es gibt auch qualitative Argumente, warum wir Business Analysis und Requirements Engineering betreiben sollten.

Falsche Anforderungen → falsche Systeme

Der Hauptgrund ist: Falsch verstandene oder nicht korrekte Anforderungen führen zu falschen Systemen. Wenn also die Anforderungen ungenügend klar sind, erhalten wir die falsche Software und Produkte und meistens merken wir das erst im Betrieb. Also, ich brauche gute Anforderungen, um zu guten Systemen zu kommen.

Implizite Anforderungen

Ein zweiter Punkt, warum das Thema so wichtig ist, sind die vielen impliziten Annahmen. Das ist ohnehin klar, da brauchen wir nicht drüber sprechen, das weiß doch jeder. Naja, scheinbar doch nicht. Denn wenn es nicht vorher klar gesagt wurde, kommen Designer und Programmierer auf ihre eigenen Ideen und entwickeln wieder vorbei an den Bedürfnissen des Markts.

Sehr viele Anforderungen sind impliziter Natur und es ist gar nicht leicht, diese aus den entsprechenden Personen herauszulocken. Wenn man es aber nicht tut, bleibt es eine implizite Annahme und jemand trifft Entscheidungen.

Interpretierbare Anforderungen

Selbst wenn Anforderungen explizit gemacht werden, ist der Projekterfolg noch nicht gewährleistet. Denn viele Anforderungen sind interpretierbar. Der Auftraggeber hält eine Aussage für absolut klar und verständlich; der Auftragnehmer interpretiert die Aussage jedoch ganz anders – und schon wieder haben wir unter Umständen eine falsche Lösung.

Requirements Creep

Zuletzt wollen wir noch auf ein Problem hinweisen, das uns oft betrifft. Anforderungen ändern sich im laufenden Projekt. Wir verfolgen ein bewegliches Ziel. Saubere Methoden und Vorgehensweisen dieses „Requirements Creep“ sollten Ihnen daher einiges wert sein.

## 1.4 Warum macht es nicht jeder richtig?

Man könnte sich jetzt die Frage stellen, warum es nicht jeder richtig macht, wenn wir doch wissen, dass schlampige Analyse sehr viele Fehler produziert und sehr teuer ist. Ich glaube, der Hauptgrund ist: Wir haben Kommunikationsprobleme miteinander. Es ist nicht leicht, Anforderungen in einer Weise auszudrücken, die alle Beteiligten verstehen. Das wird umso schwieriger, je weltweit verteilter unsere Projekte sind, wenn wir nicht mehr die gleiche Sprache sprechen oder in anderen Kulturen groß geworden sind – dann ist auch das gegenseitige Verständnis nicht so gut.

Das ist aber nur ein Argument. Ein weiteres Argument, warum es nicht jeder richtig macht, ist: Wir wollen einfach Geld sparen. Gute Analyse kostet Zeit und Aufwand. Zeit und Aufwand, den man vielleicht erst ersetzt bekommt, indem man keine Nachbesserungen machen muss. Das ist aber *nach* Abwicklung des Projekts. Mancher Projektleiter ist heilfroh, wenn er seinen Abnahmetest bestanden hat und fertig ist, und nach ihm die Sintflut. Was es dann noch kostet, die Fehler auszubessern, ist unter Umständen nicht mehr in der Verantwortung eines Projektleiters. Und deshalb versucht man, möglichst wenig Analyse zu machen, um möglichst schnell zu Lösungen zu kommen, was aber vielleicht dann teure Nachbesserungen erfordert.

Vielleicht noch ein weiteres Argument: Es geht mitunter nicht nur um das Sparen von Geld, sondern manchmal auch von Zeit. Wir wollen einfach rasche Lösungen und nehmen uns nicht die Zeit, vorher festzulegen, was es sein soll, das wir hier produzieren. Das ist aber ein falsch verstandenes Zeitsparen, denn die Zeit brauchen Sie hinterher mehrfach wieder.

Aus vielen dieser Gründe werden Business Analysis und Requirements Engineering nicht von allen Leuten ernst genommen, aber in den letzten 20 Jahren haben viele Unternehmen erkannt, dass es ein sehr wichtiges Thema ist, haben viel in dieses Thema investiert und es ist deutlich besser geworden.

Natürlich sind auch die Methoden in den letzten 35 Jahren deutlich besser geworden. Als ich angefangen habe zu arbeiten, gegen Ende der 1970er-Jahre, gab es nicht viel an Analysemethodik. In den letzten 35 Jahren ist hier eine Menge dazugekommen an Verständnis, wie man systematisch Geschäftsprozesse untersucht und Anforderungen erheben kann, wie man sie dokumentieren, prüfen und im Lauf der Zeit verwalten kann.

## 1.5 Zwei weitere Gründe

Wogegen wollen Sie denn Ihr System testen, wenn es gar keine expliziten Anforderungen gibt? Die Antwort ist klar: Die Tester denken sich einfach etwas aus. Und das trifft oftmals nicht die wahren Wünsche der Stakeholder.

Außerdem sind Anforderungen – auch wenn sie anfänglich vielleicht grob sind – die Basis für Aufwandsschätzungen – und somit eine wertvolle Grundlage für Ihre Vorhabensplanung. Mehr dazu im Abschnitt 11.8.

# 1.6 Standardisierung und Zertifizierung

Business Analyst und Requirements Engineer sind heute keine geschützten Berufsbezeichnungen. Jeder, der möchte, kann sich diese Titel selbst verleihen, kann sich Systemanalytiker nennen oder Business Analyst oder auch Requirements Engineer, ohne dafür einen bestimmten Nachweis zu erbringen.

Zur Verbesserung dieser Situation hat sich 2006 eine Gruppe von Fachleuten zusammengeschlossen, um das IREB zu gründen, das International Requirements Engineering Board (*https://www.ireb.org*).

Die Hauptaufgabe dieses Boards war, das Wissen, das man zur Ausübung dieses Berufsbilds benötigt, zusammenzuschreiben und einen Lehrplan zu formulieren, zunächst für Basiskenntnisse, die man unbedingt nachweisen sollte, um den Beruf gut ausüben zu können. Neben diesem Lehrplan über das Basiswissen wurden Fragenkataloge erstellt, mit denen man nachweisen kann, das Wissen wirklich erworben zu haben. Sie können sich dieses Wissen bei unabhängigen Stellen zertifizieren lassen.

Als Gründungsmitglied dieses Vereins habe ich das Buch natürlich IREB-konform gestaltet. Es bietet alle Grundlagen, die man kennen muss, um die Prüfung zum CPRE (Certified Professional for Requirements Engineering) auch erfolgreich bestehen zu können. Bis 2022 wurden mehr als 84 000 Personen nach diesem Programm weltweit zertifiziert. In den letzten Jahren (mit Ausnahme der Corona-Jahre) waren es mehr als 7 500 Personen pro Jahr, die die IREB-Ausbildung durchlaufen haben. Der Verein bietet seit einigen Jahren auch entsprechende Aufbaumodule an, um das Basiswissen zu vertiefen. Das Buch deckt auch große Teile von Advanced Level Requirements Modeling und RE@Agile ab (Bild 1.2).

Unabhängig davon ist auch in Kanada ein Programm entstanden: das IIBA (International Institute of Business Analysis, *https://www.iiba.org*), das ähnliche Ziele wie das IREB verfolgt. In diesem Buch lernen Sie auch alle relevanten Teile des BABOK (Business Analysis Book of Knowledge) kennen. Auch IIBA bietet ein mehrstufiges Zertifizierungsprogramm an.

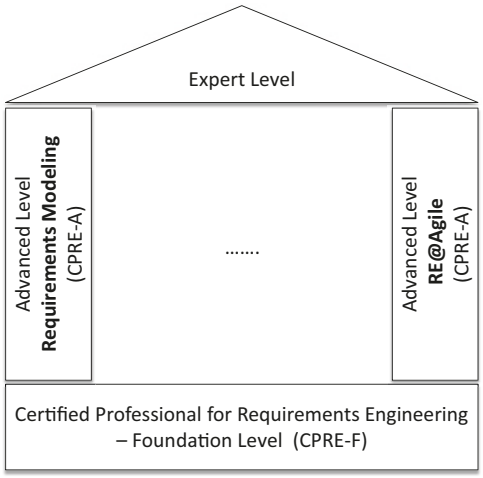


**Bild 1.2**
Die drei Ebenen des IREB-Zertifizierungsprogramms

## 1.7 Drei Säulen erfolgreicher Vorhaben

Beginnen wir mit einer Gesamtbetrachtung von Projekten oder Produktentwicklungsvorhaben. Wie passt unser Thema „Business Analysis & Requirements Engineering“ in den gesamten Lebenszyklus hinein? Vielleicht hat Ihre Firma, Ihr Unternehmen, Ihre Organisation ein vorgeschriebenes Vorgehensmodell. Besonders populär im Behördenumfeld ist zum Beispiel das V-Modell, das von einigen Ministerien für staatliche Projekte vorgeschrieben wird. Auch viele Firmen haben für sich ausführliche Vorgehensmodelle erstellt, nach denen alle internen Projekte abgewickelt werden sollen. Wenn Sie nach einem dieser Vorgehensmodelle arbeiten müssen, haben Sie sicherlich festgestellt, dass Sie sehr, sehr viele Tätigkeiten machen müssen, viele Dokumente erzeugen und verschiedenste Rollen besetzen. Die Zahl der Dokumente bei diesen Vorgehensmodellen ist oft dreistellig; es sind auch zirka 150–170 getrennte Tätigkeiten.

Ich möchte versuchen, Ihnen ein einfacheres Weltbild nahezulegen, das leichter zu merken ist und aus nur drei Grundpfeilern besteht (vgl. Bild 1.3). Einer von diesen dreien ist, herauszufinden, welches Problem der Auftraggeber, ein Kunde oder eine Organisation hat; das Problem zu definieren, das Problem zu durchleuchten und zu verstehen. Das ist unser Thema Business Analysis und Requirements Engineering. Vielleicht sagen Sie auch Geschäftsprozessanalyse dazu; vielleicht verwenden Sie auch das Wort Systemanalyse. Wir meinen auf jeden Fall: herauszubekommen, was unser Geschäft oder unsere Kunden brauchen bzw. was unsere Anwender von IT-Systemen erwarten oder mit den Produkten machen wollen.

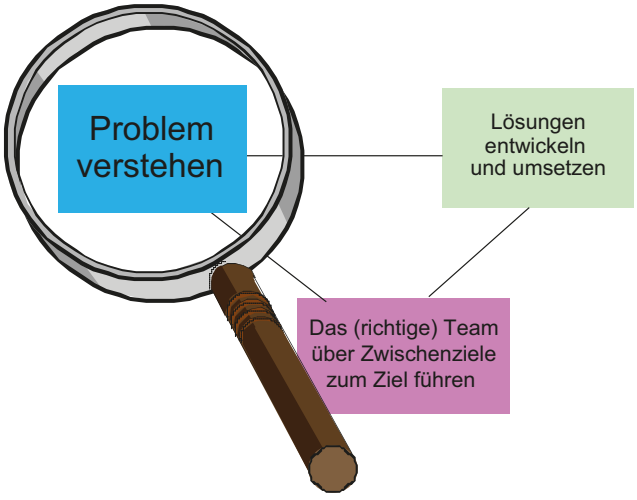


**Bild 1.3**
Drei Säulen für erfolgreiche Vorhaben

Die zweite Säule meines einfachen Weltbilds ist es, die Anforderungen in eine Lösung umzusetzen. Wenn diese Lösung eine IT-Lösung ist, dann gehören dazu der Aufbau einer Softwarearchitektur, vielleicht auch einer Hardwarearchitektur, und die Entwicklung, der Test und die Inbetriebnahme von Software. Wenn die Lösungen organisatorisch bewerkstelligt werden sollen, dann umfasst dieser Block die detaillierten Festlegungen und die Umsetzung in Aufbau- und Ablauforganisation. Kurz gesagt: eine komplette Lösung für ein vorgegebenes Problem erstellen.

Und die dritte Säule in meinem einfachen Weltbild ist Projektmanagement: das richtige Team zu haben; diesem Team gute Ziele zu geben, vernünftige Pläne aufzustellen, Zwischenziele festzulegen, deren Erreichung man prüfen kann, und das Team erfolgreich über diese Zwischenziele zum Ziel zu führen.

Die Tätigkeiten in den drei Säulen werden heutzutage hochgradig iterativ durchgeführt. Ständig werden Sie gleichzeitig Probleme analysieren, Lösungen designen und Ihre Vorhaben oder Produktentwicklungen managen.

Mit diesen drei Säulen können Sie Vorhaben erfolgreich abwickeln, Ihr Business effektiver gestalten und Ihre Produkte erfolgreicher machen. Dieses Buch handelt hauptsächlich von der ersten dieser drei Säulen: von Business Analysis und Requirements Engineering. Wir werden aber auch die Schnittstellen zwischen diesen beiden Themen und der Umsetzung in Lösungen sowie die Schnittstellen zwischen Business Analysis/Requirements Engineering und Projektmanagement ausführlich behandeln.

## 1.8 Definition: Business Analysis und Requirements Engineering

Bevor wir uns mit den Inhalten von Business Analysis und Requirements Engineering näher beschäftigen, sollten wir uns auf Definitionen einigen. Was also verbirgt sich hinter den beiden Begriffen?

Das International Requirements Engineering Board [IREB], dessen Programm diesem Buch zugrunde liegt, hat die Gesamtheit des Themas als Requirements Engineering bezeichnet. Die beiden Hauptteile, die dazugehören, sind einerseits die Requirements-Analyse – von vielen auch als Systemanalyse bezeichnet – und andererseits Requirements-Management, d. h., wenn ich einmal Anforderungen habe, diese auch zu managen, zu verwalten und zu pflegen. Zu den Unterpunkten der Requirements-Analyse gehört einerseits, Requirements aus Kunden herauszulocken, und andererseits, diese auch zu Papier zu bringen – also Requirements ermitteln und spezifizieren.

Sie sehen an dem einfachen Mengendiagramm in Bild 1.4 die drei Teile, aus denen dieses Gebiet besteht:

- Anforderungen ermitteln (oder herauslocken),
- Anforderungen spezifizieren (auf irgendeine Art skizzieren, niederschreiben oder zeichnen) und
- Anforderungen managen (verwalten, pflegen, versionieren, priorisieren).

Die ersten beiden davon werden oft auch als „Requirements-Analyse“ oder „Systemanalyse“ zusammengefasst. Welche Wörter Sie für die Summe aller dieser Tätigkeiten verwenden, überlasse ich Ihrer Entscheidung. Das IREB hat sich für „Requirements Engineering“ als Oberbegriff entschieden; ob Sie das in Ihrem Umfeld als Requirements Engineering bezeichnen oder Requirements-Management als Oberbegriff verwenden oder vielleicht Problemanalyse oder Business-Analyse, überlasse ich Ihnen.

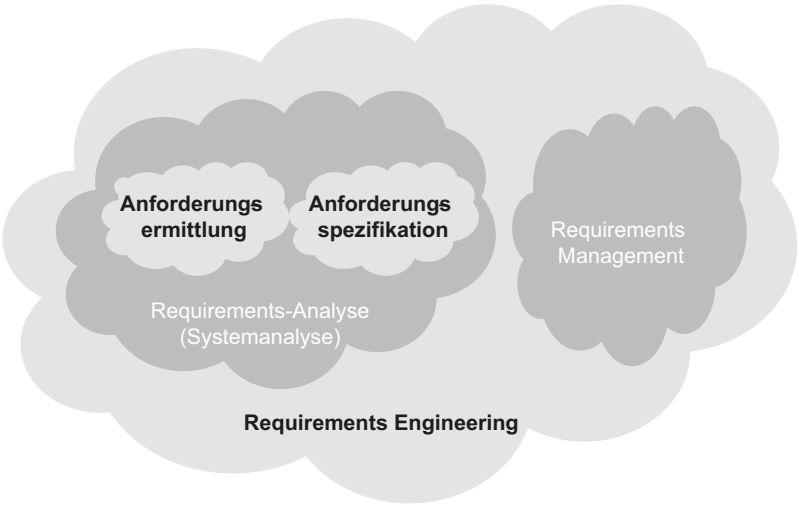


**Bild 1.4** Die Hauptthemen im Requirements Engineering

Definition: Requirements Engineering

Ich möchte Ihnen aber neben diesem Mengendiagramm doch noch eine ausführliche Definition geben. Requirements Engineering im heutigen Sinne ist ein kooperativer, iterativer und inkrementeller Prozess mit folgenden drei Zielen:

1. Alle relevanten Anforderungen sollen bekannt und im erforderlichen Detaillierungsgrad verstanden sein.
2. Die involvierten Personen und Organisationen (Stakeholder) sollen ausreichende Übereinstimmung über die bekannten Anforderungen erzielen.
3. Die Anforderungen sollen konform zu den Dokumentationsvorschriften der Organisation spezifiziert sein.

Alleine in der Einleitung dieser Definition stecken drei Aussagen drin. Ein kooperativer Prozess heißt: Wir arbeiten zusammen und nicht gegeneinander. Wir wollen nicht Anforderungen über den Zaun werfen zu anderen Leuten, sondern wir wollen sie gemeinsam entwickeln zwischen Auftraggeber und Auftragnehmer oder in Scrum: zwischen Product Owner und Development Team.

Die Wörter „iterativ“ und „inkrementell“ richten sich gegen das alte Wasserfallmodell. Ich muss nicht alle Anforderungen am Anfang perfekt kennen. Ich kann in Iterationen (Releases oder Sprints) – schrittweise – Teile davon, weitere Teile davon, noch mehr Teile davon zeitgerecht entwickeln.

Danach lernen wir durch die Definition die drei wesentlichen Ziele dieses kooperativen Prozesses kennen.

Das erste Ziel ist: Ich möchte alle **relevanten** Anforderungen so gut und im **erforderlichen Detaillierungsgrad** kennen, dass ich eine Lösung darauf basieren lassen kann. Sehen Sie die Weichmacher in dieser Definition? Alle relevanten Anforderungen in hinreichendem Detaillierungsgrad? Was ist relevant? Was bedeutet hinreichender Detaillierungsgrad? Wir werden uns ausführlich mit beiden Fragen auseinandersetzen.

Das zweite Ziel besagt, dass die beteiligten und betroffenen Personen, die Stakeholder, die Organisationen, die mitspielen müssen, hinreichende Übereinstimmungen über diese Anforderungen erzielen sollten. Nun ja, in der Diktatur reicht es, wenn der Diktator ja sagt. Wenn Sie etwas demokratischer aufgestellt sind, brauchen Sie vielleicht die Zustimmung von ein paar mehr Leuten. Aber selbst in einer Demokratie sind 50,01 % eine Mehrheit. Es wird also immer wieder Personen geben, die dagegen sind, die andere Wünsche haben. Wir wollen aber, dass der Prozess dafür sorgt, dass wir eine hinreichende Übereinstimmung über die bekannten Anforderungen erzielt haben.

Und das dritte Ziel für diesen Prozess Requirements Engineering ist noch einfacher. Wir wollen, dass die Anforderungen nach den Regeln des Hauses, nach Ihren Spielregeln, nach Ihren Dokumentationsvorgaben aufgeschrieben werden. Wenn Sie in Ihrem Haus gar keine Vorschriften haben, wie man ein Pflichtenheft oder ein Lastenheft erstellt oder den Product Backlog erfasst, sind Sie auf jeden Fall konform dazu ☺. Aber die meisten Firmen haben natürlich Vorgaben, was alles festzuhalten ist und wie es festgehalten werden sollte.

Requirements Engineering ist also ein kooperativer Prozess, der in Zyklen durchgeführt wird. In jedem Zyklus muss ich genügend herausfinden für das, was ich demnächst umsetzen möchte. Ich brauche genügend Übereinstimmung und ich muss es hausgerecht festhalten, nach unseren Dokumentationsvorschriften.

Wenden wir uns nun dem etwas weiter gefassten Begriff „Business Analysis“ zu. Das International Institute of Business Analysis (IIBA) definiert in seinem Business Analysis Body of Knowledge (BABOK) Folgendes:

Definition: Business Analysis

Business Analysis ist eine Menge von Aufgaben und Techniken, die genutzt werden, als Liaison zwischen Stakeholdern zu arbeiten, um die Struktur, die Direktiven (Policies) und die Arbeitsweise einer Organisation zu verstehen und (fachliche) Lösungen vorzuschlagen, die es der Organisation ermöglichen, Ihre Ziele zu erreichen.

Das entsprechende Überblickbild über Business-Analyse identifiziert acht Kernbereiche, die dort als Wissensgebiete (Knowledge Areas) bezeichnet werden (vgl. Bild 1.5).

Sicherlich sehen Sie die starken Überlappungen zu der Definition von Requirements Engineering. Auch hier stehen die Erhebung (von Anforderungen) und das Managen von Anforderungen rechts und links im Fokus. Analysieren wurde hier dreigeteilt: Enterprise-Analyse und Requirements-Analyse, natürlich nicht ohne beides zu bewerten und zu validieren. Eingerahmt werden diese Kerntätigkeiten durch die Planung und Überwachung des Gesamtvorgehens sowie die Forderungen nach grundlegenden Kompetenzen der Mitwirkenden, die wir später in diesem Kapitel genauer betrachten werden.

Was haben die beiden Definitionen gemeinsam? Sowohl bei Business Analysis wie auch beim Requirements Engineering geht es um das Verstehen eines Systems (d. h. eines Geschäfts, einer Organisation, eines Produkts oder eines Systems), mit dem Ziel, Schwachstellen zu identifizieren, Ideen für Verbesserungen zu entwickeln, damit zu besseren oder neuen Produkten bzw. zu optimierten Ablauf- und Aufbauorganisationen zu kommen. Bild 1.6 fasst die wesentlichen Aspekte von Business Analysis und Requirements Engineering zusammen:

- Wir gehen davon aus, dass es jemanden gibt, der mit dem Ist-Zustand von Produkten oder Organisationen nicht zufrieden ist. Jemand, der ehrgeizige Ziele hat, Wünsche und Ideen und Visionen, oder aber durch äußere Zwänge wie Gesetze und andere Auflagen gezwungen wird, am Status-quo etwas zu verändern.

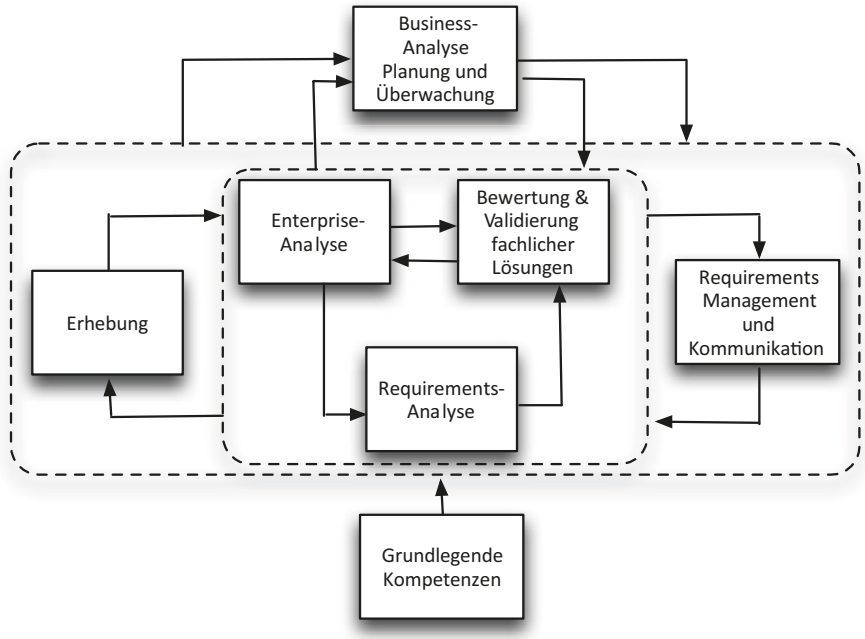


**Bild 1.5** Wissensgebiete der Business-Analyse

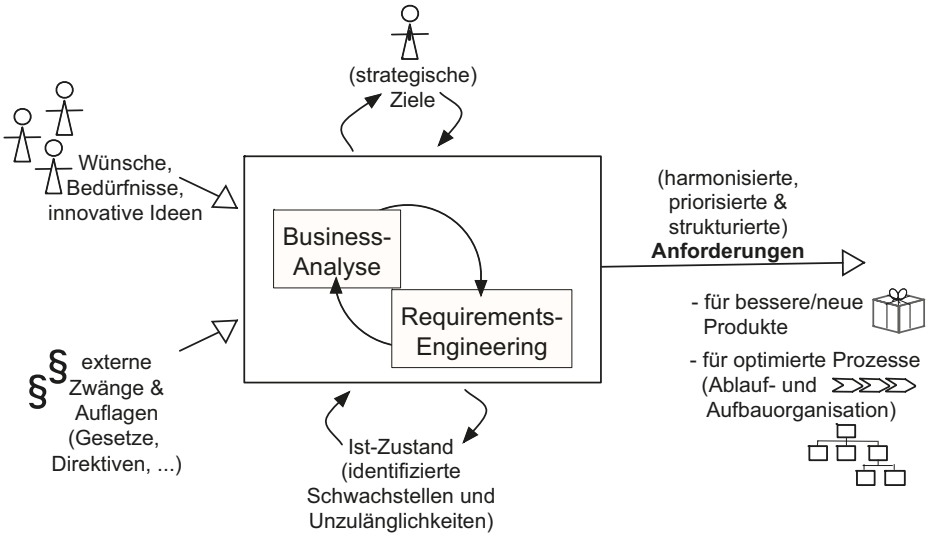


**Bild 1.6** Business-Analyse und Requirements Engineering

- Durch Analyse des heutigen Zustands von Organisationen und Produkten identifiziert man Schwachstellen und entwickelt Lösungsideen und Verbesserungsvorschläge, die man in Form von möglichst präzisen Anforderungen an jemanden übergeben kann, der in der Lage ist, die Lösung in die Praxis umzusetzen. Entweder mittels IT-Systemen oder aber auch durch organisatorische Maßnahmen.

- Diese Tätigkeiten werden gemeinsam von allen Stakeholdern durchgeführt, die ein Interesse an der Sache haben, in einem kooperativen Prozess, wobei wir ausgewogen über kurz-, mittel- und langfristige Lösungen nachdenken. Je kurzfristiger wir etwas ändern wollen, desto genauer müssen wir Anforderungen spezifizieren. Je langfristiger Themen sind, desto vager können wir im Ergebnis bleiben. Durch iteratives und inkrementelles Vorgehen sichern wir jederzeit die Konzentration und Bündelung unserer Kräfte auf das zurzeit Notwendige.

## 1.9 Definition: Requirement

Anforderungen sind identifizierbar und halbwegs formalisiert

Jetzt haben wir so häufig das Wort „Requirements“ erwähnt. Was überhaupt ist eine einzelne Anforderung, ein Requirement? Nicht alles, was uns Projektbeteiligte erzählen, ist automatisch eine Anforderung. Sie sprechen über viele Bedürfnisse, über Wünsche. Sie geben uns zahlreiche Hintergrundinformationen. Sie erklären uns vielleicht sogar, warum sie das Ganze haben wollen, und geben uns Erläuterungen aus dem bisherigen Leben, aus Altsystemen oder was in Zukunft sein soll. Zur Anforderung wird es erst dann, wenn Sie es halbwegs formalisieren und identifizierbar machen. Identifizierbar, denn im Zweifelsfall wollen wir sagen, das war eine Anforderung oder das war keine Anforderung. Es nur gesagt zu haben oder nur erwähnt zu haben, reicht oft nicht aus. Wir wollen eine identifizierbare Aussage haben und sie sollte auch halbwegs formalisiert sein.

Sie brauchen aber keine Angst zu haben. Sie müssen nicht unbedingt komplexe Diagramme lernen. Ein sauber geschriebener deutscher Satz mit Subjekt, Prädikat und Objekt oder ein sauber geschriebener englischer Satz oder eine gut formulierte User-Story erfüllt auch das Kriterium, halbwegs formalisiert zu sein. Aber die beiden Punkte brauchen wir, denn wir brauchen eine Einigung, ob irgendwas eine Anforderung war oder nicht, denn wir schreiben Anforderungen hauptsächlich deshalb, damit wir beim Übergeben einer Lösung, beim Abnahmetest, beim Sprintende feststellen können, ob die Anforderung auch erfüllt ist. Wenn wir zu diesem Zeitpunkt nichts hätten, wogegen wir prüfen können, nichts, was wir identifizieren können, nichts, was halbwegs eindeutig formuliert ist, dann hätten wir Probleme; wir könnten nicht feststellen, ob das geliefert wurde, was wir bestellt haben oder haben wollten.

Für Formalisten hat IEEE, die amerikanische Ingenieursvereinigung, natürlich formale Definitionen festgehalten, was Anforderungen sind. Gemäß IEEE 610.12-1990 ist eine Anforderung

1. eine Bedingung oder Fähigkeit, die von einem Benutzer (Person oder System) zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird.
2. eine Bedingung oder Fähigkeit, die ein System oder Teilsystem erfüllen muss, um einen Vertrag, eine Norm, eine Spezifikation oder andere, formell vorgegebene Dokumente zu erfüllen.
3. eine dokumentierte Repräsentation einer Bedingung oder Fähigkeit gemäß (1) oder (2).

Umgangssprachlich ausgedrückt heißt der erste Punkt: etwas, was wir **wollen**, was irgendjemand will. Der zweite Teil heißt umgangssprachlich ausgedrückt: etwas, was wir haben **müssen**, ob wir wollen oder nicht, weil ein Gesetz oder eine Norm oder eine Vorschrift dahintersteckt. Und egal, ob es ein Wunsch war oder ob wir es haben müssen, jede Niederschrift einer solchen Aussage zählt natürlich auch als Anforderung. Also nicht nur das gesprochene Wort, sondern vor allem die dokumentierte Form von solchen Wünschen oder Zwängen zählen als Anforderung.

Der neue IEEE-Standard 29148 ist etwas pragmatischer. Er definiert: Eine Anforderung ist eine Aussage, die einen Wunsch samt seinen Einschränkungen und Randbedingungen ausdrückt oder übersetzt.

Bild 1.7 bringt diese Aussagen auf den Punkt: Anforderungen sind das Bindeglied zwischen zwei Parteien: jemanden, der etwas haben will (Auftraggeber, Kunde, Anforderer, Product Owner, ...), und jemanden, der es liefern kann (Auftragnehmer, Solution Provider, Development Team, ...).

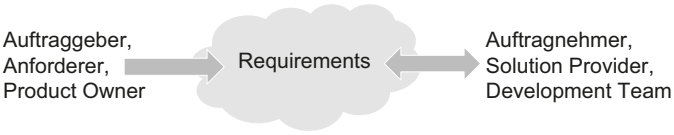


**Bild 1.7** Anforderungen – das Bindeglied zwischen den Beteiligten

Es ist egal, wer der Auftraggeber ist, Ihre interne Marketingabteilung, ein Kunde oder ein wirklicher Endnutzer. Es ist auch egal, ob Sie im gleichen Unternehmen sitzen oder unterschiedlichen Unternehmen angehören. Auftragnehmer ist jemand, der das Ganze umsetzen kann; jemand, der diese Anforderung nimmt und daraus eine gewünschte (organisatorische oder technische) Lösung erstellen kann. Anforderungen sind das Bindeglied zwischen diesen beiden Welten, egal ob sich dazwischen Firmengrenzen, Abteilungsgrenzen oder Systemgrenzen befinden. Es ist das, was die beiden Parteien zusammenbringt.

## 1.10 Arten von Anforderungen

Wir haben jetzt geklärt, was Anforderungen sind. Als Nächstes wollen wir über unterschiedliche Arten von Anforderungen sprechen. Die Klassifizierung soll uns helfen, Anforderungen leichter zu finden und zu erfassen. Wir wollen zwei bis drei Arten von Anforderungen unterscheiden (vgl. Bild 1.8). Die klassische Zweiteilung erfolgt in funktionale und nichtfunktionale Anforderungen.

Funktionale Anforderungen sind eher einfach. Das sind die Funktionen, die das System leisten soll; die Informationen, die es verarbeiten soll; das gewünschte Verhalten, welches das System an den Tag legen soll.

Nichtfunktionale Anforderungen ist eigentlich ein Unwort. Auch im Englischen: Non-functional Requirements. Nichtfunktionale Anforderungen? Was soll das sein? Die Dinge, die hinterher nicht funktionieren? Nein. Das ist nicht damit gemeint. Es ist einfach das Gegenstück

zu den funktionalen Anforderungen: alles andere außer den Funktionen und Informationen des Systems. Und es wird wesentlich klarer, wenn wir sie noch in zwei Teile unterteilen: in die Qualitätsanforderungen und in Randbedingungen.

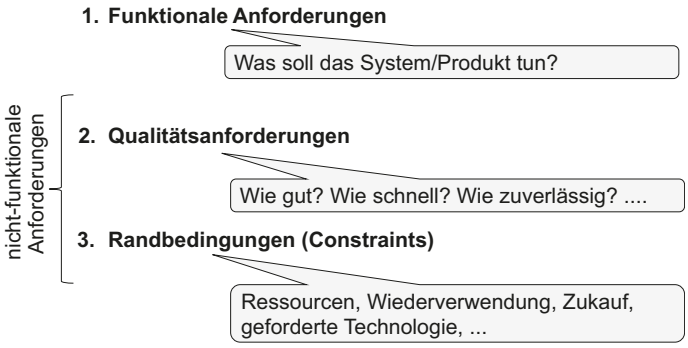


**Bild 1.8**
Kategorisierung von Anforderungen

Eine Bitte: Streichen Sie das Wort „nichtfunktionale Anforderungen“ aus Ihrem Sprachschatz. Verwenden Sie stattdessen die viel aussagekräftigeren Begriffe „Qualitätsanforderungen“ und „Randbedingungen“.

Qualitätsanforderungen sind Ergänzungen zu den Funktionen; präziser gesagt: zu einer oder mehreren Funktionen. Sie sagen uns, wie gut die Funktionen ausgeführt werden sollen, wie schnell, wie zuverlässig, wie gesetzestreu, wie sicher, wie wartbar ... Es reicht nicht, zu sagen: „Mein Auto soll bremsen können.“ Sie wollen, dass es einen Meter vor der Mauer stehen bleibt und nicht erst einen Meter hinter der Mauer. Das ist eine Qualitätsanforderung an die Funktion „bremsen können“. Und diese Qualitätsanforderungen sind genauso wichtig und wir wollen sie genauso sorgfältig erforschen und erfassen wie die funktionalen Anforderungen.

Die dritte Kategorie Randbedingungen, im englischen Constraints genannt, ist etwas schwieriger. Denn viele Leute sind der Meinung, dass das eigentlich keine Anforderungen sind. Aber zu dieser Kategorie gehören technische und organisatorische Vorgaben, die ein Auftraggeber natürlich machen kann. Wenn der Auftraggeber sagt: „Du darfst maximal 300.000,– € ausgeben“, so ist das weder eine funktionale Anforderung noch eine Qualitätsanforderung. Es ist einfach eine Randbedingung. „Du musst bis Dezember fertig sein“ ist eine zeitliche Randbedingung. „Du darfst die Aufbauorganisation nicht verändern“ ist eine organisatorische Randbedingung. „Du sollst Rechner von Hersteller X einsetzen“ oder „Du sollst diese Datenbank verwenden“ sind technische Randbedingungen. „Du sollst dieses Framework verwenden“. Alles Randbedingungen. Viele betrachten sie als unechte Anforderungen; sie gehören aber definitiv zu der Menge der Anforderungen dazu.

Etwas formaler ausgedrückt: Funktionale Anforderungen definieren Funktionen, die vom Gesamtsystem oder von einem Teilsystem, von einem Modul, von einer Komponente ausgeführt werden sollen.

Qualitätsanforderungen definieren qualitative Eigenschaften, die diese Funktionen haben sollen. Wiederum sind das entweder qualitative Eigenschaften vom Gesamtsystem oder von einem großen Teilsystem oder von einer einzelnen Funktion; aber sie stehen niemals im

luftleeren Raum. Sie gehören immer zu einer oder mehreren funktionalen Anforderungen dazu. Und die Randbedingungen, die Constraints, sind organisatorische oder technische Vorgaben, die dem Designer, den Lösungsverantwortlichen, die Hände fesseln. Sie haben nicht jegliche Freiheitsgrade, eine Lösung zu entwickeln, sondern sie müssen sich bei dieser Lösung an solche Randbedingungen halten, an Zeitvorgaben, an Budgetvorgaben, an Technologievorgaben. Freiheitsgrade, die dem Designer durch den Auftraggeber entzogen werden, aus gutem Grund. Das sind unsere drei Kategorien von Anforderungen.

## 1.11 Hauptaufgaben von Analytikern und Produkt-Ownern

Im nächsten Abschnitt wollen wir uns die Hauptaufgaben eines Business Analyst, eines Requirements Engineer oder eines Product Owners ansehen. Bild 1.9 zeigt, dass wir sie auf vier wesentliche Punkte reduzieren können.

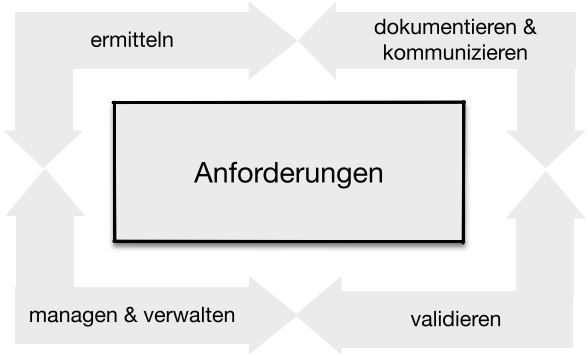


**Bild 1.9**
Hauptaufgaben eines Analytikers

**Ermitteln**

Erster wesentlicher Punkt: Ich muss Anforderungen ermitteln können. Ich muss sie herauskitzeln, ich muss sie begreifen und verstehen können, ich muss wissen, was der Kunde haben möchte, und Methoden haben, um darauf zu kommen. Als ich vor 30 Jahren gelernt habe, waren Interviews die einzige Technik, um aus Kunden Wünsche herauszubekommen. Wir werden später im Buch (in Kapitel 10) diskutieren, dass das bei Weitem nicht die einzige Technik ist. Wir haben heute Dutzende von Erhebungstechniken, um Anforderungen aus Stakeholdern herauszulocken, nicht nur Interviews.

**Dokumentieren und Kommunizieren**

Die zweite wesentliche Tätigkeit ist, verstandene Anforderungen zu dokumentieren. Ich habe sie begriffen, ich muss sie aber noch irgendwie schriftlich festhalten. Und dazu stehen uns im Wesentlichen drei Möglichkeiten zur Verfügung: schreiben, malen oder simulieren. Schreiben heißt, Sätze schreiben. Deutsche, englische, natürlichsprachige Sätze schreiben, die die Anforderung wiedergeben. Malen heißt, formale grafische Modelle einzusetzen: BPMN-Diagramme, Use-Case-

Diagramme, Datenmodelle, Aktivitätsdiagramme; die vielen Modelle, die uns die Methoden heute zur Verfügung stellen, um vielleicht etwas präziser als nur in Umgangssprache auszudrücken, was gewollt wird. Und simulieren heißt, wir machen Screenshots von IT-Lösungen, wir zeigen Masken, wir zeigen Maskenfolgen, um zum Beispiel die Akzeptanz zu prüfen, ob eine derartige Umsetzung den Wünschen des Auftraggebers gerecht würde. Oder wir bauen Prototypen und Mock-Ups, die wir als Diskussionsgrundlage vorführen können. Sie haben also drei unterschiedliche Arten zur Verfügung, um zu spezifizieren, was Anforderer wollen oder brauchen. Wir werden uns sehr ausführlich mit allen drei Arten beschäftigen und die alternativen Dokumentationstechniken sowie ihre Vorteile und Nachteile genauer erläutern.

**Trend: weniger formale Dokumentation, mehr Kommunikation**

Früher galt es, Anforderungen so präzise und eindeutig wie möglich aufzuschreiben. Mit dem Aufkommen der agilen Methoden hat sich das etwas gewandelt. Oft dient die Dokumentation nur als „Gedächtnisstütze“ dessen, was man miteinander geklärt hat. Wir werden das bei den User Storys noch genauer besprechen. So ein Story-Kärtchen ist nur der Platzhalter für eine Anforderung. Was genau gemeint ist hat der Product Owner mit dem Team unter Umständen mündlich geklärt.

Einen ähnlichen Trend beobachten wir bei den grafischen Modellen. Statt ganz präzise UML-Modelle (oder andere) zu erstellen, reicht oft eine Skizze, denn ein Bild sagt mehr als 1 000 Worte. Auch darauf gehen wir im Kapitel über Requirements-Modelle noch genauer ein.

Doch bedenken Sie: Je weiter PO und Entwickler auseinandersitzen, je mehr Anforderungen an Unterauftragnehmer als Vertragsgrundlage übergeben werden, desto präziser müssen sie auch dokumentiert sein. Den Luxus, weniger genau zu dokumentieren, kann man sich nur leisten, wenn man verspricht, dafür mehr und öfter miteinander zu sprechen.

Validieren

Jetzt haben wir Anforderungen ermittelt und niedergeschrieben. Als Nächstes müssen wir sie validieren, d. h. prüfen und eventuell konsolidieren. Gerade wenn viele Leute gleichzeitig arbeiten und unabhängig voneinander Anforderungen niederschreiben, kann es schon vorkommen – ja nicht nur kann, es wird wahrscheinlich vorkommen –, dass wir Widersprüche haben, dass unterschiedliche Wünsche von unterschiedlichen Leuten nicht zusammenpassen. Und genau daran müssen wir arbeiten. Wer sollte jetzt Recht bekommen? Welche Anforderung wollen wir wirklich erfüllen? Sind wir schon widerspruchsfrei, sind wir in uns konsistent? Wir wollen den Lösungspartnern ja keine widersprüchlichen Wünsche übergeben. Das sicherzustellen, ist die dritte Haupttätigkeit.

Verwalten

Und für die vierte Haupttätigkeit habe ich leider noch eine sehr schlechte Nachricht für Sie. Anforderungen ändern sich mit ein bis drei Prozent pro Monat. Wenn wir also ein längeres Projekt machen, über mehr als einen Monat hinweg, ist die Wahrscheinlichkeit sehr groß, dass Anforderungen auch geändert werden – mit ein bis drei Prozent Wahrscheinlichkeit. Eine Projektdauer von zwölf Monaten und Sie haben die Chance auf 12 × 2 % – im Schnitt also 24 % – geänderte Anforderungen. Deshalb brauchen wir als weitere Tätigkeit Requirements verwalten. Das Verwalten von Requirements fasst alle Tätigkeiten mit einmal geschriebenen Anforderungen zusammen.

Wie versionieren wir sie? Wie stellen wir fest, welche momentan gültig ist? Wie gehen wir mit Change-Requests, mit Änderungsanträgen, um? Und wie machen wir Traceability zwischen Analyse und Design und Sourcecode und Testdaten? Alle diese Themen fallen in den Bereich „Requirements verwalten“ oder „Requirements managen“.

Herauskitzeln, Niederschreiben und Kommunizieren, Validieren (Prüfen und Abstimmen) und Verwalten sind die vier Haupttätigkeiten, die ein „klassischer“ Systemanalytiker durchzuführen hat.

**Vom Verwalten zum Managen**

Mit dem Wechsel vom klassischen Requirements Engineer zum Product Owner haben sich auch die Aufgaben geändert. Jetzt kommen zu den vier Aufgaben noch einige managementorientierte Aufgaben hinzu. Ein klassischer Business-Analyst oder Requirements-Engineer durfte (und sollte) alles mit den Anforderungen erledigen, durfte aber keine Entscheidungen über den Vorhabensablauf treffen und Anforderungen nicht aus Sicht des Geschäfts priorisieren, sondern höchstens Abhängigkeiten feststellen. Diese Aufgaben waren dem Projektmanagement vorbehalten. Im agilen Umfeld ist es sogar die Hauptaufgabe eines Product Owner, die wertbringenden Funktionen höher zu priorisieren, früher genau zu spezifizieren (oder von dem Team spezifizieren zu lassen) und dann auch früher im Vorhaben zur Umsetzung zu bringen. Mehr über diese Schwerpunktverlagerung lesen Sie im nächsten Abschnitt.

*Lassen Sie mich eine kleine Geschichte erzählen. Als ich 1976 meinen ersten Job angetreten habe und – wie im Vorort erwähnt – von meinem Chef meinen Arbeitsvertrag mit der Berufsbezeichnung Systemanalytiker überreicht bekam, wollte ich das, was wir an der Technischen Universität Wien nicht gelernt hatten, natürlich nachholen. Wir haben damals Kollegen zu 5-Tage-Kursen geschickt, um über Systemanalyse zu lernen. Als die zurückkamen, habe ich neugierig gefragt: Was habt ihr denn gelernt? Was ist und tut denn ein Systemanalytiker? Und die haben mir die Essenz des Kurses mitgeteilt: „Zieh' Dich anständig an, wenn Du zum Kunden gehst, sei höflich, tritt dem Kunden nicht auf die Füße, hör zu und schreib auf, was er sagt. Und das wichtigste Handwerkszeug sind Radiergummi, Bleistift und gesunder Menschenverstand.“ Nun, die Welt hat sich seither kräftig weiterentwickelt. Wir wissen heute viel mehr über dieses Berufsbild, und das alles sollen Sie im Rest des Buchs erfahren.*

## 1.12 Aufgabenverteilung im Team

Wir haben bis jetzt immer über die Person eines Requirements Engineer, Business Analyst oder Product Owner gesprochen. Sehen wir uns die Aufgabenverteilung im Team einmal an, wer alles eine Rolle in Zusammenhang mit Systemanalyse hat (vgl. Bild 1.10). Wenn Sie ein explizites Rollenbild eines Requirements Engineer oder eines Business Analyst haben, dann wird diese Person hauptsächlich vier Tätigkeiten durchführen (ermitteln, dokumentieren, verwalten und auch prüfen und abgleichen). Ich kenne aber Projekte, wo es diese Rolle gar nicht explizit gibt. Und trotzdem muss irgendjemand diese Tätigkeiten durchführen. Irgendjemand im Team macht das dann. Ob das ein Entwickler oder der Product Owner oder wer auch immer übernimmt, ist egal.

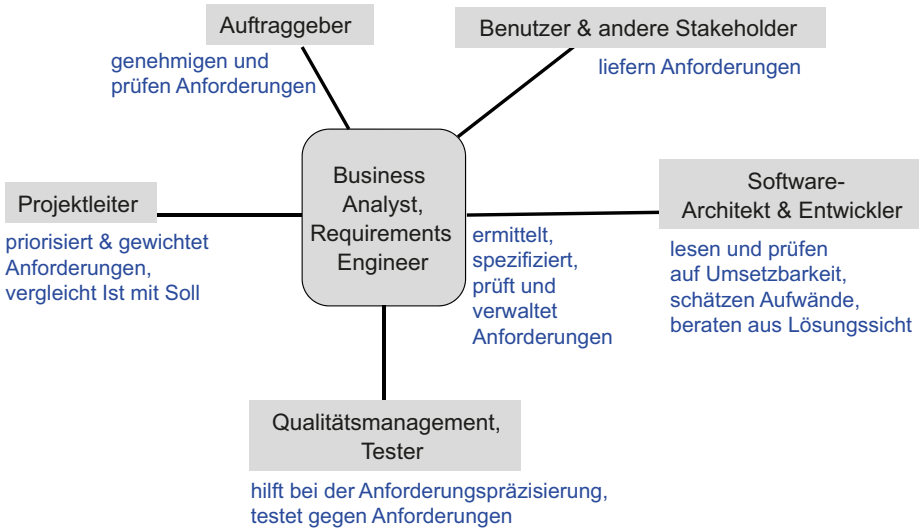


**Bild 1.10** Aufgaben im Umfeld von Business Analysis und Requirements Engineering

Trotzdem, selbst wenn Sie diese Spezialrolle haben, haben auch andere Personen Aufgaben im Zusammenhang mit Business Analysis und Requirements Engineering, wie zeigt.

Nehmen wir einmal den Auftraggeber. Der trägt schließlich die Verantwortung für das Ganze; der will es haben; der ist der Geldgeber dafür und der wird vielleicht Ziele und grobe Anforderungen vorgeben. Und unsere Benutzer, die Personen, die später die Lösung nutzen sollen. Von denen – zusammen mit anderen Stakeholdern – wird der größte Teil der Anforderungen kommen, die Sie als Systemanalytiker aufschreiben sollen.

Als Analytiker haben Sie meistens nicht sehr viel Macht, denn die Entscheidung über Prioritäten, wie wichtig diese Anforderung ist, ob sie frühzeitig oder später erledigt werden sollte, trifft sehr oft der Projektleiter in Abstimmung mit dem Auftraggeber. Sie bereiten vor, Sie geben Abhängigkeiten an, Sie sagen, das eine kann nicht gemacht werden ohne das andere; aber die Entscheidung darüber trifft ein Projektleiter. Also hat auch ein Projektleiter eine Rolle in Zusammenhang mit Anforderungen. Und Projektleiter werden meistens auch Soll und Ist überwachen. Wie weit sind wir fortgeschritten? Und was ist das Nächstwichtige. Sie werden – vielleicht mit vielen anderen Leuten – Schätzungen vornehmen, was das Ganze kosten kann. Sie helfen natürlich als Systemanalytiker bei diesen Aufgaben.

Auch das Team, das später die Umsetzung machen soll (Design, Implementierung, Test, ...), hat bereits in der Business-Analyse oder im Requirements Engineering eine Rolle. Denn das Team muss Aufwände schätzen und aus Sicht technischer oder organisatorischer Risiken beraten. Je früher Sie solche Personen sehen lassen, was gemacht werden soll, desto früher können die sagen: Das wird viel zu aufwendig, das ist hoch risikobehaftet. Versucht mal eine andere Lösung stattdessen, wir hätten hier gute Ideen. Das sind nicht die Leute, die hauptsächlich Anforderungen vorgeben, aber sie tragen sehr gut dazu bei, dass die Anforderungen machbar sind, dass sie umsetzbar sind, dass sie nicht zu teuer werden. Also schalten Sie bitte auch das Entwicklungsteam ein, wenn Sie Pflichtenhefte oder Lastenhefte schreiben oder Ihren Product Backlog befüllen.

Und – last but not least – Qualitätssicherung und Tester: In vielen Projektorganisationen kommen diese erst viel zu spät zum Zug, wenn das System fertig entwickelt ist. Sie sollten sie frühzeitig einschalten. Dazu ein Zitat von Dorothy Graham, eine berühmte Testerin, Buchautorin und Rednerin auf vielen Konferenzen. Sie hat einmal gesagt: „Testen ist zu wichtig, um es den Testern zu überlassen." [DeM22] Was sie damit gemeint hat: Testen beginnt viel früher! Wenn Sie Anforderungen ermitteln, ziehen Sie rechtzeitig Tester hinzu. Lassen Sie diese die Anforderungen lesen und zwar unter der Vorgabe: „Kann ich das prüfen? Wüsste ich, wie man das Ganze umsetzt?" Denn wenn die Anforderungen noch zu vage sind, wird Ihnen der Tester sagen: „Das passt noch nicht. Ich weiß nicht, wie ich's prüfen sollte." Auch zum Zeitpunkt der Requirements-Erstellung spielen die Tester also schon eine bedeutsame Rolle. Ihre Hauptrolle kommt natürlich später. Es sind die Personen, die nach Fertigstellung des Systems prüfen, ob die Anforderungen auch wirklich erfüllt wurden.

Sie sehen also, fast jeder im Projekt muss früher oder später seinen Beitrag leisten zum Thema Anforderungen.

Was ich Ihnen gerade vorgestellt habe, ist eher die klassische Aufteilung. Wir haben jemanden, der das Projekt verantwortet, den Projektleiter. Und der trifft auch Geldentscheidungen, Prioritätsentscheidungen. Wir haben jemanden, der die Systemanalyse erarbeitet und hauptsächlich für Pflichtenhefte, Lastenhefte zuständig ist. Wir haben jemanden, der testet. Jemanden, der umsetzt, designt, implementiert, und wir haben Auftraggeber. In neueren Verfahren, in agilen Methoden, werden diese Rollen ganz anders aufgeteilt (vgl. Bild 1.11).

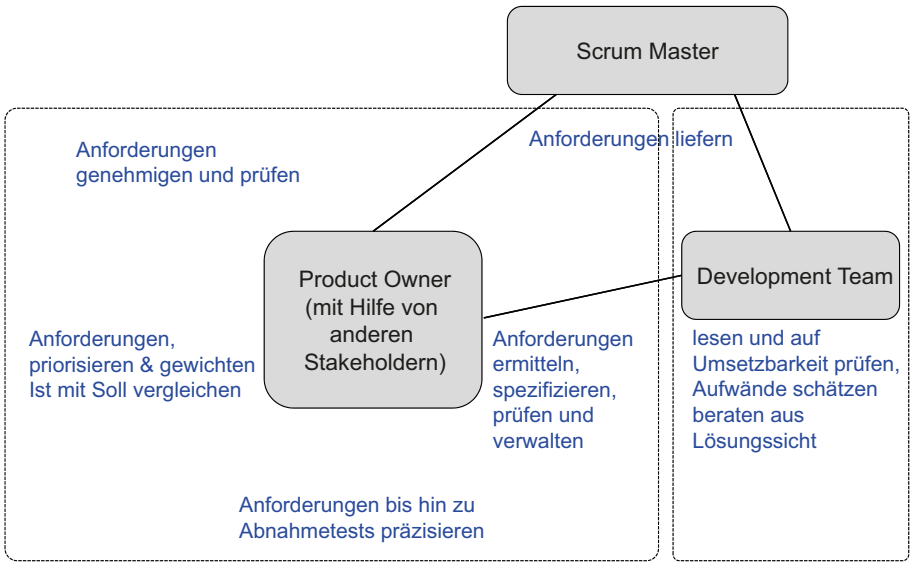


**Bild 1.11** Neue Rollenverteilung für die Aufgaben

Wenn Sie Scrum als agiles Vorgehensmodell nehmen, gibt es viele von den oben genannten Rollen nicht mehr. Wir haben neben dem Scrum-Master, der für die Einhaltung der Spielregeln sorgt, im Wesentlichen einen Product Owner (einen Produkteigner) und ein Entwicklungsteam. Der Produkteigner ist quasi Projektleiter und Systemanalytiker und evtl. sogar Auftraggeber in einem. Sie sehen: Viele Rollen werden hier zusammengefasst unter einer

Verantwortung. Ich habe also als Product Owner nicht nur die Aufgabe, die Anforderungen vorzugeben und zwar so detailliert vorzugeben, dass das Team sie versteht und umsetzen kann; sondern ich habe auch die Aufgabe, alle beteiligten Personen zu koordinieren, eine einheitliche Meinung dem Entwicklungsteam gegenüber zu vertreten, was gemacht werden soll und was nicht, und gleichzeitig auch Prioritäten aus Business-Sicht vorzugeben, was wie wichtig ist. Ich betrachte das als einen sehr guten Trend, weil damit das Dilemma Projektleiter gegen Analytiker gegen Auftraggeber vielleicht aus der Welt geschafft wird.

Das neue Rollenbild des „Product Owner" ist nicht einfach, aber sehr zukunftsträchtig. Da ist jemand, der das große Ganze, die Vision im Auge behält und dann – Stück für Stück – iterativ, inkrementell Produktversionen entwickeln lässt, die ständig den Mehrwert des Produkts erhöhen. Natürlich werden wir noch eine Zeitlang mit Mischformen leben. Dann müssen Projektleiter und Requirements-Verantwortliche einfach klare Absprachen haben, wer wofür die Verantwortung übernimmt.

## 1.13 Benötigte Fähigkeiten

Vorher wollen wir uns allerdings ansehen, welche Eigenschaften ein Systemanalytiker, Business Analyst, oder ein Requirements Engineer haben sollte, um den Job wirklich gut ausführen zu können.

Analytisches Denken

Ich glaube, allem voran ist analytisches Denken gefragt. Ein Problem durchleuchten zu können, abstrahieren zu können, auf den Punkt zu bringen, das Wesentliche herauszufinden; ein gerütteltes Maß an analytischem Denken ist Voraussetzung für die Ausübung dieses Berufs.

Selbstbewusstes Auftreten und Kommunizieren

Außerdem sollte ein Systemanalytiker ein halbwegs selbstbewusstes Auftreten haben. Als Programmierer können Sie sich hinter dem Rechner verstecken. Sie müssen nur mit Ihrem Rechner kämpfen. Sie müssen nicht mit Menschen sprechen. Ich übertreibe bewusst. Als Systemanalytiker haben Sie ständig Kontakt mit vielen Leuten und Sie müssen diese auch koordinieren. Dazu gehören Moderationsfähigkeiten, aber auch ein gesundes Einfühlungsvermögen und auf jeden Fall gestandenes Selbstbewusstsein. Sie müssen sich mit größeren Gruppen von Leuten auseinandersetzen. Sie brauchen Überzeugungsfähigkeiten und Kommunikationsfähigkeiten. Sie müssen auf die Ausdrucksweise, die Sprechweise, das Verständnis Ihrer Gesprächspartner eingehen können. Manchmal kann man nur über Prototypen kommunizieren, manchmal über deutsche Sätze, manchmal mit grafischen Modellen. Nicht Sie bestimmen, was gute Kommunikation ist. Meistens bestimmen Ihre Projektpartner, wie man am besten und am effizientesten kommuniziert. Sie sollten all diese Variationen der Kommunikation beherrschen. Dazu gehört auch sprachliche Kompetenz. Sie müssen sich in der Projektsprache vernünftig ausdrücken und klare, eindeutige Sätze formulieren können.

Methodische Kompetenz

Zu den Fähigkeiten gehört auch methodische Kompetenz – das ist der Schwerpunkt, den Ihnen dieses Buch vermitteln kann. In den letzten 35 Jahren wurden viele Methoden entwickelt, wie man Anforderungen ermittelt, do-

kumentiert, prüft und verwaltet. Die können Sie lernen. Viele von den anderen Fähigkeiten sollten Sie mitbringen.

**Domänenkenntnisse?**

Lange diskutiert haben wir im IREB über Domänenkenntnisse. Muss ein Systemanalytiker gute Domänenkenntnisse haben, also zum Beispiel Versicherungsexperte sein, wenn wir ein Versicherungsprojekt durchführen, oder Getriebeexperte, wenn Sie über Getriebesteuerungen sprechen? Wir haben uns schließlich darauf geeinigt, Domänenkenntnisse helfen, sind aber nicht unbedingt Voraussetzung für einen guten Systemanalytiker. Sehen Sie mich an. Ich bin das beste Beispiel dafür. Ich arbeite in vielen Branchen, von denen ich anfangs wenig Ahnung habe. Irgendjemand in der Gruppe von Personen, mit denen Sie arbeiten, sollte allerdings Ahnung haben. Und Ihr Geschick als Systemanalytiker ist es, dieses Wissen aus den anderen herauszuholen und in geordneter Form zu kommunizieren. Sie müssen nicht Topspezialist in dieser Domäne sein. Aber ein gesundes Verständnis des Umfelds, in dem Sie arbeiten, hilft auf jeden Fall und macht Systemanalyse leichter.

Überlegen Sie einmal, was ein Systemanalytiker überhaupt nicht können muss, um seinen Job gut auszuüben: C# zum Beispiel, die neuesten NOSQL-Datenbanken, die letzten Versionen von irgendwelchen Hardwaresystemen. Dafür haben wir Designer, Programmierer, Hardwareingenieure. Deren Job ist es, eine Lösung zu bauen. Also, Sie brauchen kein Technologiespezialist sein. Es schadet natürlich auch nicht, wenn Sie die Sprache der Leute, die hinterher weiterarbeiten, verstehen. Aber das ist nicht Ihre Hauptaufgabe.

**Managementfähigkeiten**

In der Rolle eines Product Owner kommen noch zwei weitere Fähigkeiten hinzu: Erstens sollte man definitiv das Business, in dem man tätig ist, sehr gut – auch langfristig – verstehen. Denn man trifft Entscheidungen darüber, welche Anforderungen frühzeitig umgesetzt werden sollten, weil sie hohen Wertgewinn versprechen und welche noch zurückgestellt werden können. Dazu muss man Kosten-Nutzen-Abwägungen gut im Griff haben. Und zweitens sollte man als Product Owner ausgeprägte Empathie und Managementfähigkeiten haben. Man ist verantwortlich, alle Stakeholder (die oft unterschiedliche, ja widersprüchliche Wünsche äußern) so zu koordinieren, dass möglichst alle glücklich werden. Das heißt oft: Kompromisse schließen bzw. harte Entscheidungen gut an alle Beteiligten zu „verkaufen“.

Ein paar andere Punkte noch: Sie müssen nicht auf jedem Gebiet, auf dem Sie Anforderungen erfassen, Topspezialist sein. Sie werden juristische Anforderungen behandeln. Für deren Präzisierung holt man sich einen Rechtsanwalt oder Hausjuristen zu Hilfe. Sie werden Sicherheitsanforderungen behandeln. Dazu haben Sie im Haus vielleicht einen Sicherheitsbeauftragten oder einen Sicherheitsspezialisten. Sie müssen nur in der Lage sein, mit solchen Personengruppen vernünftig umzugehen und deren Wissen transparent zu machen. Also bitte suchen Sie sich für Spezialgebiete Unterstützung. Sie müssen nicht alle Themen alleine beherrschen.

Ist also ein Systemanalytiker eher eine extrovertierte Person oder eine introvertierte Person? Ich glaube, Sie haben es herausgehört. Introvertiert können Sie als Programmierer sein; als Systemanalytiker sollten Sie extrovertiert sein und mit Menschen gerne und gut umgehen können.

# 1.14 Der Aufwand für die Analyse

Betrachten wir im nächsten Abschnitt den Aufwand, den wir für die Problemanalyse veranschlagen müssen. Ich habe dafür eine einfache Faustformel entwickelt von 4 × 25 % (vgl. Bild 1.12).

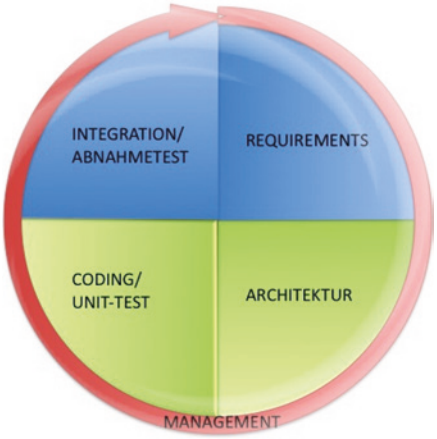


**Bild 1.12**
Aufwandsverteilung für Geschäftsoptimierung und (Teil-)Automatisierung

Der Aufwand für Business Analysis und Requirements Engineering ist für mich in vielen Vorhaben ein Viertel des gesamten Aufwands des Vorhabens. 25 % der Zeit und 25 % der Kosten des gesamten Verbesserungsaufwands gehen in das Herausfinden, Kommunizieren und Verwalten von guten Anforderungen. Das erscheint Ihnen viel? Sie müssen nicht so viel machen. Wenn Sie weniger vorgeben, dann treffen anschließend andere Personengruppen (z. B. Ihre Designer und Programmierer) die Entscheidung über das, was das System genau tut. Deshalb seien Sie von Anfang an ehrlich. Es ist viel Arbeit, herauszubekommen, was genau gemacht werden soll.

Die tröstliche Nachricht ist aber: Sie müssen diese 25 % nicht am Anfang eines Vorhabens erbringen. Der Aufwand erstreckt sich über den gesamten Vorhabensverlauf. Ich kann anfangs mit viel weniger auskommen; kann schon erste Teile einer Lösung zuführen (z. B. in Form eines ersten Release eines IT-Systems, eines Minimal Marketable Product – MMP). Denn Sie erinnern sich an die Definition von Requirements Engineering? Es ist ein iterativer, inkrementeller Prozess. Aber der Gesamtaufwand liegt bei 25 %.

Die einfache 4 × 25 %-Formel stimmt nicht für alle Vorhabensgrößen. Sie ist aber durchaus schon anwendbar bei kleinen Projekten, bei denen Sie zu zweit oder zu dritt über drei Monate arbeiten. Selbst dann können Sie bereits mit dieser Faustformel arbeiten. Die Faustformel endet bei sieben bis zehn Personen über ein bis zwei Zeitjahre, also Vorhaben in der Größenordnung zwischen zehn und 20 Personenjahren.

Nur kurz zum Rest des Aufwands: Weitere 25 % gehen in das Design, in Grob- und Feinentwurf, nochmal 25 % in die Codierung, in die Umsetzung und den Unit Test und die letzten 25 % in Integrationstest, Abnahmetest, User Acceptance Test, wie immer Sie es nennen wollen.

Das ist nur eine Faustformel. Die realen Werte können davon abweichen, aber für mittelgroße Projekte zwischen einem halben Personenjahr und zehn bis 20 Personenjahren funktioniert

diese Faustformel. Was ist, wenn die Projekte noch kleiner sind? Eine Person, die in 14 Tagen ein Problem lösen muss? Da überwiegt deutlich die linke Hälfte des Kreises. Sie werden hauptsächlich implementieren, testen und wieder ausliefern und integrieren. Wenn Ihr Projekt aber größer ist als zehn bis 15 Personen, dann überwiegt die rechte Hälfte des Kreises. Und es kann statt 50 % auch 60 %, 65 %, 70 % ausmachen, das Problem zu verstehen und Lösungsansätze im Großen zu konzipieren – zu Lasten der Umsetzung und Inbetriebnahme.

Ihnen fehlt das Projektmanagement in dem Bild? Na gut. Sehen Sie den äußeren Rand an. Ziehen Sie 10 % von dem Ganzen ab. Dann bleiben in jedem Quadranten 22,5 % statt 25 % übrig. Ich habe zur Vereinfachung das Management in die jeweilige Tätigkeit mit hineingerechnet.

Ihnen fehlt die Qualitätssicherung? Wir machen alle vier Bereiche inklusive Qualitätssicherung! Wir werden in den 25 % Requirements-Aufwand die Requirements qualitätssichern. Wir werden in den 25 % Design das Design qualitätssichern und wir liefern garantiert kein Produkt aus, das nicht getestet ist. Qualitätssicherung habe ich als integralen Bestandteil in die diversen Blöcke mit hineingerechnet.

Das ist die Aufwandsverteilung über ein komplettes Vorhaben. Damit ist nicht gesagt, dass Sie das alles in Ihrem Projektbudget machen müssen. Einen Teil der Anforderungen sollte ja der Auftraggeber beisteuern. In Bild 1.13 sehen Sie eine Überblendung mit den Verantwortungen von Auftraggeber und Auftragnehmer. Je nachdem, wie viel Prozent der Auftraggeber selbst an Requirements vorgibt, bleibt der Rest dann als Projektarbeit für den Solution-Provider. Im Idealfall bekommen Sie als Auftragnehmer Ihren Auftraggeber dahin, dass er mehr und größere Teile davon macht. Wenn Sie gute Requirements hereinbekommen, brauchen Sie im Projekt nicht mehr so viel Aufwand zu betreiben. Eine typische Verteilung der 25 % Requirements-Aufwand liegt jedoch eher bei 5 % Business- oder User-Requirements und 20 % Requirements-Präzisierung durch den Auftragnehmer. Seien Sie also nicht allzu optimistisch bezüglich der Qualität der auftraggeberseitig gelieferten Anforderungen. Mehr dazu in Kapitel 8.

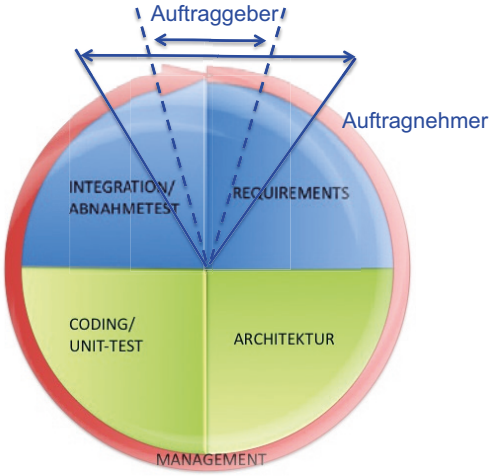


**Bild 1.13**
Arbeitsverteilung zwischen Auftraggeber und Auftragnehmer

Der Aufwand, der von irgendjemandem insgesamt erbracht werden muss, liegt bei 25 %. Ob der auftraggeberseitig oder auftragnehmerseitig geleistet wird, können wir offen lassen. Sicherlich wird der Endkunde einen Teil des Abnahmetests und den endgültigen Akzeptanztest erbringen. Sie liefern als Auftragnehmer eine gut getestete Lösung aus, aber ob diese gefällt, wird beim Auftraggeber festgestellt.

Die Aufwandsverteilung ist also nicht eine Projektkostenverteilung, sondern eine reine Tätigkeitsverteilung: 25 % Requirements, 25 % Architektur, 25 % Umsetzung und 25 % Endarbeiten inklusive Test und Auslieferung.

## 1.15 Was erleichtert die Analyse?

Betrachten wir in diesem Abschnitt, was den Prozess der Systemanalyse erleichtern kann und was ihn erschweren kann. Unter bestimmten Randbedingungen ist es viel leichter, zu Anforderungen zu kommen, als unter anderen. Was erleichtert also die Analyse, Spezifikation und Abstimmung unserer Wünsche?

Ein offensichtlich erster Punkt: wenn sich die beteiligten Parteien schon gut und bereits länger kennen. Anforderungen sollten nicht interpretierbar sein; es sollten keine Missverständnisse möglich sein. Wenn man sich schon lange kennt und weiß, wie der andere denkt, wie der andere fühlt, welches Weltbild der Partner hat, ist es leichter, sich auch mit informellen Äußerungen, mit unpräzisen Äußerungen zu verständigen. Man kann davon ausgehen, dass der andere einen schon versteht. Wesentlich schwerer ist es, wenn man sich gerade kennengelernt hat.

Sie haben einen Auftragnehmer ins Projekt genommen, den Sie als Billigstanbieter kennengelernt haben? Er hat das günstigste Angebot abgegeben. Sie haben keine Ahnung, wie diese Firma arbeitet. Sie haben noch nie zusammengearbeitet. Das wird es natürlich schwieriger machen. Jetzt müssen wir uns über den Inhalt wesentlich eingehender verständigen, als wenn wir gegenseitiges Verständnis voraussetzen können. Denken Sie einmal an ein altes Ehepaar – 30, 35, 40 Jahre verheiratet. Die kommen ohne Worte aus. Man sieht sich gegenseitig an und weiß ganz genau, was der andere jetzt denkt oder sagen möchte. Worte sind kaum notwendig. Ja, wenn Sie sich gerade kennengelernt haben, haben Sie sich noch viel zu erzählen.

Der zweite Punkt: Requirements Engineering wird dann leichter, wenn wir in der gleichen Muttersprache arbeiten, wenn wir die gleiche Sprache sprechen. Denn in jeder Sprache gibt es Nuancen von Wörtern, die jemand, der die Sprache als Fremdsprache gelernt hat, bestimmt nicht so gut interpretieren kann wie jemand, der mit der gleichen Sprache aufgewachsen ist. Es hilft also schon, ein Projekt so mit Personal auszustatten, dass alle die gleiche Sprache sprechen. Denken Sie jedoch einmal dran, was wir heute in unseren Projekten machen. Wir verteilen sie weltweit zwischen vielen Kulturen über Erdteile hinweg!

Kulturen sind das dritte Thema. Mein Musterbeispiel dafür ist die Sendung „Wer wird Millionär?“. Haben Sie diese einmal in einem anderen Land gesehen? Haben Sie mal Armin Assinger statt Günther Jauch gesehen? Die ersten fünf Fragen sind in einem anderen Kulturkreis die schwierigen. Jemand, der in dem Kulturkreis aufgewachsen ist, kann die meist völlig locker beantworten. Jemand, der auch nur aus dem Nachbarland kommt, mit noch immer gleicher Muttersprache, aber etwas anderer Kultur, hat unter Umständen genau mit diesen einfachen Fragen Schwierigkeiten. Sobald es zu den Wissensfragen geht, können wir wieder gut zusammenarbeiten. In Deutschland reicht schon der Unterschied zwischen Westdeutschland und Ostdeutschland. Jemand, der im Osten groß geworden ist und Fragen bekommt, die sich

eher auf den Westen beziehen, wird Schwierigkeiten in den Antworten haben genau wie umgekehrt Personen aus dem Westen, die die Kultur des Ostens nicht so detailliert kennengelernt haben. Sie sehen schon: Abstammung aus dem gleichen Kulturkreis würde wirklich beim Umgang mit Anforderungen das Leben erleichtern.

Der vierte Punkt ist ein gleicher Domänenhintergrund. Wenn Sie schon lange Zeit in einer Bank arbeiten oder Autos bauen, kennen Sie die Terminologie dieser Branche, Sie kennen die Begriffe; Sie wissen, was die meinen, wenn sie ein bestimmtes Wort verwenden. Das hilft gewaltig, sich gegenseitig zu verstehen, ohne alles bis zum i-Tüpfelchen definieren zu müssen. Aber Vorsicht! Zu viel gleiche Branchenkenntnisse führen manchmal zu dem Effekt: Alles klar, brauchen wir nicht definieren, jeder versteht das Gleiche darunter. Und in Wirklichkeit hat doch jeder eine andere Meinung dazu. Hin und wieder ist ein Branchenfremder ganz nützlich, um Ideen zu hinterfragen, bei denen alle Beteiligten glauben, sich zu verstehen, und sich doch nicht wirklich verstehen. Also gemeinsamer Domänenhintergrund, gemeinsamer Branchenhintergrund kann schon hilfreich sein.

Der letzte Punkt ist noch heikler. Wenn Sie zueinander Vertrauen aufgebaut haben, ist die Systemanalyse leichter. Anders ausgedrückt: wenn untereinander keine politischen Spielchen gespielt werden. „Wir haben das absichtlich so formuliert, dass wir später sagen können, das ist doch klar, dass das und das und das und das auch noch dazugehört." Für mich war das vielleicht klar. Jemand anderer hat den Satz vielleicht gelesen, viel zu gering eingeschätzt, die Komplexität, die dahintersteht, unterschätzt und kommt erst sehr spät im Projekt darauf, dass das mit Absicht so schwammig formuliert wurde, nur damit man billig einen Auftrag bekommt. Also Vorsicht: Politische Spielchen machen den Analyseprozess auch sehr schwierig. Wenn man sich gegenseitig verstehen will, wenn man keine Spielchen spielt, wenn man positiv vertrauensvoll zusammenarbeitet, ist der Umgang mit Anforderungen viel leichter.

Vergleichen Sie das mit unserer Realität. Wir wissen heute, dass Projekte an einem Standort mit eng zusammenarbeitenden Teams, die alle die gleiche Sprache sprechen, die alle aus der gleichen Kultur kommen, sicherlich erfolgreicher wären. Aber im realen Leben verteilen wir Projekte oft weltweit; wir nehmen Auftraggeber, Auftragnehmer, die weit auseinandersitzen, unterschiedliche Sprachen sprechen, und müssen trotzdem erfolgreich Projekte durchführen.

Ich möchte geografische Verteilung und Kulturunterschiede definitiv nicht verdammen. Ganz im Gegenteil. Multikulturell besetzte Projekte sind hoch erwünscht, wenn es um innovative Ideen, um andere Weltbilder, andere Anschauungen geht. Das kann einen wesentlichen Beitrag dazu leisten, nicht zu engstirnig zu werden, Lösungen zu sehen, die kein anderer gesehen hat. Aber meine Aussage ist: Über Anforderungen zu sprechen, wird dadurch schwieriger. Wenn wir uns nicht selbstverständlich verstehen, müssen wir hart daran arbeiten, keine Missverständnisse aufkommen zu lassen. Und das ist immer schwieriger über Sprachgrenzen hinweg, über geografische Grenzen hinweg, über kulturelle Grenzen hinweg oder zwischen Parteien, die sich misstrauen und absichtlich Spielchen spielen.

Überlegen Sie, wie weit Sie durch organisatorische Maßnahmen und Projektbesetzung die Requirements-Arbeiten erleichtern können. Wie weit können Sie dafür sorgen, dass Sie gute Randbedingungen haben? Alles, was uns keine zusätzlichen Schwierigkeiten macht, reduziert das Risiko im Projekt und erhöht die Erfolgschance, auch mit etwas weniger als perfekten Requirements zum Ziel zu kommen.