

1 Einleitung

Wer sein Ziel nicht kennt, kann jeden Weg nehmen.

– Alice im Wunderland

1.1 Warum ein Buch über Requirements Engineering?

Alice fragte: »Könntest du mir bitte sagen, welchen Weg ich von hier aus nehmen soll?« »Das hängt vor allem davon ab, wohin du gehen willst«, sprach die Katze. »Ich weiß es nicht ...«, sagte Alice. »Dann ist es egal, wohin du gehst«, antwortete die Katze.

Dieser kurze Dialog aus dem Buch »Alice im Wunderland« beschreibt wunderschön, warum Anforderungen und Ziele eine Rolle spielen. Viel zu oft zerbrechen wir uns vorschnell den Kopf über eine Lösung – ohne verstanden zu haben, wie das Problem überhaupt aussieht. Wir laden zu einer Besprechung ein, ohne zu hinterfragen, was sie eigentlich bringen soll. Wir entwickeln Features für ein Softwaresystem und wissen nicht, welchen Wert sie für die Käufer und Benutzer darstellen. Wir optimieren und bemühen uns ständig, bessere Produkte zu entwickeln – ohne uns klarzumachen, was diese Produkte erreichen sollen, wenn sie auf den Markt kommen. In der Folge ändern sich Anforderungen an diese Produkte ständig, denn es ist uns zu keinem Zeitpunkt klar, was wir damit konkret erreichen wollen.

Prüfen Sie sich einfach einmal selbst, und beantworten Sie die beiden folgenden Fragen spontan und ehrlich. Hat Ihr derzeitiges Projekt einen expliziten Business Case (also nicht nur einige undurchsichtige und schwammige Marketingvorgaben)? Gibt es für jede einzelne Anforderung eine Begründung, die aus Benutzersicht (oder aus der Sicht derjenigen, die für das Produkt bezahlen sollen) beschreibt, was durch diese Anforderung besser wird? Falls nicht, ist das Buch das richtige für Sie. Falls ja, lesen Sie die Fragen nochmals und gehen aufrichtig in sich.

Klassisch sind in diesem Zusammenhang schon die Migrationsprojekte, die als wichtigste Vorgabe immer angeben, dass »alle Funktionen des existierenden Altsystems übertragen werden müssen«. Ein Fehler. Erstens kann sowieso keiner

mehr alle existierenden Altfunktionen im Zusammenhang beschreiben (und Archäologie gehört zu den wenigen Disziplinen, die nicht explizit im Software-Engineering verankert sind). Und zweitens ist gerade ein neues System die einzige Chance, gleichzeitig auch alte Prozesse und Workflows über Bord zu werfen.

Die Anforderungen an Software werden zunehmend komplexer. Abbildung 1–1 zeigt das Komplexitätswachstum von verschiedenen Softwaresystemen, die wir untersucht haben¹. Auf der waagrechten Achse sind die Jahreszahlen angegeben, während senkrecht das Softwarevolumen in tausend Objektcodebefehlen dargestellt ist. Diese Darstellung erschien uns als die einzig praktikable, wenn wir so unterschiedliche Systeme, wie Betriebssysteme, Vermittlungssysteme und eingebettete Software, vergleichen wollen. Während das Wachstum in den siebziger und achtziger Jahren eine Verdoppelung des Umfangs nach 5-7 Jahren brachte, verdoppelt sich heute der Umfang bereits alle 2-4 Jahre. Die Verdoppelungsrate hat sich verdoppelt! Mit diesem Wachstum steigt auch der Umfang der Spezifikationen an. Gab es Anfang der neunziger Jahre beispielsweise einige wenige Steuergeräte in einem Neuwagen mit ungefähr hundert Seiten an Spezifikationen, so sind es heute bereits knapp fünfzig Steuergeräte mit über 100.000 Seiten an Spezifikationen. Mit dieser zunehmenden Komplexität sind Funktionen immer häufiger korreliert und in unterschiedlichen Hardwaresystemen vernetzt, was zusätzliche Komplexität durch nichtfunktionale Anforderungen mit sich bringt.

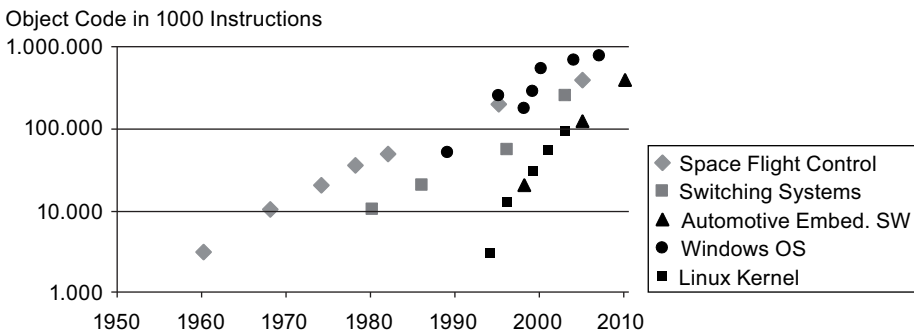


Abb. 1-1 Komplexitätswachstum von Softwaresystemen

Diese schnell wachsende Komplexität fordert systematisches RE, um die Qualität und Kosten nachhaltig kontrollieren zu können. Projektmanager und Entwickler wissen, dass es erprobte Methoden sowie werkzeugunterstützte Hilfsmittel für das Requirements Engineering gibt. Häufig fehlt ihnen aber der Überblick über die Theorie und Praxis des Requirements Engineering, um die für ihre Situation

1. Quellen der Daten: Eigene Studien des Autors zu Vermittlungssystemen von Alcatel-Lucent (S12) und Siemens (EWSD), NASA, Studien der HIS sowie Codeanalysen bei Windows und Linux. Konversionen soweit nötig gemäß den Korrekturfaktoren von Les Hatton (<http://www.leshatton.org/Documents/LOC2005.pdf>).

passenden Methoden, Verfahren und Hilfsmittel auszuwählen, sowie die notwendige Kenntnis im Detail, um sie produktiv nutzen zu können.

Das Buch füllt diese Lücke und liefert Theorie und Praxis des Requirements Engineering, sodass die Konzepte direkt umgesetzt werden können. Die gängigen Verfahren der Anforderungsanalyse und -verfolgung sind beschrieben. Die Leser erhalten Einblick in die Art und Weise, wie Anforderungen ermittelt, entwickelt, dokumentiert und im Projekt verfolgt werden. Die grundsätzlichen Methoden, Verfahren, Werkzeuge und Notationen des Requirements Engineering werden übersichtlich behandelt. Sie werden durch konkrete Beispiele aus der Projektarbeit illustriert. Notationen und Modelle sind in der Regel mit UML 2.0 beschrieben. Fallstudien demonstrieren die konkrete Umsetzung und Erfahrungen aus der Praxis.

1.2 Risiken des Requirements Engineering

Projekte scheitern aus verschiedenen Gründen. Die neueste Studie der Standish Group aus 2006 zeigt, dass nur 35% der Projekte zu einem erfolgreichen Abschluss kommen, 19% werden abgebrochen, und der Rest kommt zwar zu einem Abschluss, aber nur unter Aufgabe von ursprünglichen Zielen [Standish2007]. Unzureichendes Requirements Engineering ist in diesen Studien nach wie vor ein Hauptgrund für den Misserfolg.

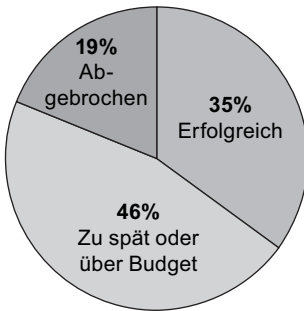
Die meisten Projekte, die abgebrochen wurden, hatten nur ungenügend geklärte Anforderungen und konnten Änderungen der Anforderungen nicht beherrschen [Standish2007, Ebert2007a, Charette2005]. Ein wichtiger Grund dafür, dass Projekte ihre Ziele nicht erreichen, liegt in nicht sauber definierten Zielen. Abbildung 1–2 zeigt diesen Zusammenhang und unterstreicht schon aufgrund der Datenlage die immense – und wachsende – Bedeutung eines guten RE. Nach wie vor ist unzureichendes RE der Hauptgrund für abgebrochene Projekte oder solche, die ihre Ziele nicht erreichen. Technologische Herausforderungen sind per se keine wichtigen Projektrisiken. Ihr Management dagegen schon.

Doch es gibt auch genügend Projekte, die ihre Ziele erreichen. Durch den systematischen Einsatz von besten Praktiken im Projektmanagement, in der Entwicklung und natürlich auch im RE hat sich die Zahl der erfolgreichen Projekte seit Mitte der neunziger Jahre verdoppelt. Abbildung 1–3 zeigt diesen Zusammenhang, in dem die Ergebnisse der ursprünglichen Studie der Standish Group aus 1994 mit jener aus 2006 verglichen werden.

Erfolg ist machbar. Viele Unternehmen haben ihre Produktentwicklung bereits erfolgreich verbessert. Die Erfolgsfaktoren sind in der Regel die gleichen:

- Ergebnisorientierte Vorgaben
- Zielorientierte Prozesse
- Kompetentes Produkt- und Projektmanagement
- Standardisierte und optimierte Infrastruktur
- Fokus auf Anforderungen, Änderungen und Risiken im Projekt

Erfolgsquote von SW- und IT-Projekten



Hauptgründe für Projektprobleme

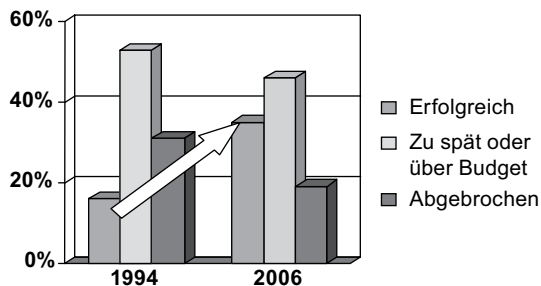
Prozessfähigkeit	91%
Organisationsmanagement	87%
Requirements Management	87%

Abb. 1-2 Unzureichendes Requirements Engineering reduziert Projekterfolge

Methodisch können diese Erfolgsfaktoren unterschiedlich erreicht werden, solange diszipliniert gearbeitet wird. Anhaltspunkte und konkrete Ansätze für die zielorientierte Verbesserung der Entwicklungsprozesse geben Modelle wie das CMMI [Chrissis2006]. Wir wollen in diesem Buch darauf eingehen, welche Techniken des RE Sie einsetzen sollten, um mit Ihren Projekten und Produkten zu den Gewinnern zu gehören.

Auf was muss man beim RE achten? Aus unterschiedlichen Praxiserfahrungen lassen sich die wichtigsten Risiken im RE ableiten. Die Risiken zu kennen, heißt, dass man sich darauf vorbereiten kann, um sie beim nächsten Mal zu vermeiden. Oder wie es Mao Zedong formuliert: »Die Niederlage zu verstehen ist der erste Schritt zum Sieg.« Die folgende Liste wurde ursprünglich von drei sehr erfahrenen RE-Praktikern erstellt [Lawrence2001] und wurde hier nochmals aktualisiert.

Entwicklung der Erfolgsquote seit 1994

**Abb. 1-3** Projekte verbessern sich durch konsequente Nutzung der richtigen Techniken

Risiko 1: Kunden sind unzureichend repräsentiert

Unzureichende Einbeziehung der Benutzer führt zu nicht akzeptierten Produkten und zu unzufriedenen Kunden. Oftmals erscheint es einfacher, die Anforderungen zu Projektbeginn vom Kunden oder Benutzer – oder intern vom Vertrieb oder Marketing – zu ermitteln, und danach das Projekt zu beginnen, um diese Anforderungen zu implementieren. Häufig übernehmen interne Stellen die Rolle des Kunden, ohne ihn direkt einzubeziehen. Das Problem dabei ist, dass sich das Projekt in zwei getrennte Richtungen entwickelt. Schließlich lernen sowohl die Projektmitarbeiter als auch der Kunde ständig dazu. Da der Kunde allerdings nicht weiß, wie er damit umgehen soll, wartet er. Das Gleiche gilt für den Projektmanager, der eine Liste führt, was er beim nächsten »offiziellen« Projektreview auf den Tisch bringen will. Bei diesem nächsten Projektreview sind dann die Divergenzen sehr viel größer, als wenn es kontinuierliche Konsultationen gegeben hätte. Eine andere Facette ist, dass es beim Kunden verschiedene Rollen gibt, aber nur dessen Einkauf repräsentiert ist. Auch das führt später zu einem bösen Erwachen, wenn man feststellen muss, dass der Einkauf primär auf formale Kriterien achtete, aber nicht auf Benutzbarkeit oder Effizienz des Produkts. Ein dritter Aspekt, der Kundenmitarbeit nötig macht, ist der, dass viele Fragen und Unklarheiten erst auftreten, wenn das Projekt in der Entwicklung ist. Sie sollten ohne großen Zeitverlust geklärt werden. Zur Abschwächung empfehlen wir daher den Kunden, kontinuierlich am Projekt zu partizipieren, und dem Projektleiter, alle wichtigen Rollen auf Kundenseite zu berücksichtigen. Beides ist nicht in allen Projekten möglich. Schließlich wollen viele Kunden ihre Zeit nicht unbedingt mit Projektarbeit verbringen. In einem solchen Fall bietet es sich an, bestimmte Aspekte vertraglich zu regeln, damit dem Kunden von Anfang an bereits klar ist, dass er sehr viel präzisere Anforderungsreviews durchführen muss, wenn er nachher nicht aktiv mitarbeitet. Eine letzte Warnung: Zu viel Kundenmitarbeit ist für beide Seiten »tödlich«. Viele Projekte scheitern, da sie wachswich aufgesetzt werden. Man weiß ja, dass man nachher im Projekt sowieso eng zusammenarbeitet, und verschiebt Detailfragen auf später. Dann aber kann man kaum von Projekt sprechen, sondern eher von einem Versuchsballon. Und dass jene die Tendenz haben, zu platzen, das dürfte hinreichend bekannt sein. Machen Sie also im Vorfeld die Rollen der Kunden bei der Projektarbeit sehr deutlich und klären Sie wie bei jedem anderen Projekt (ohne Kundenmitarbeit) vor Projektstart, was das Projekt zu liefern hat.

Risiko 2: Kritische Anforderungen werden übersehen

Eine der Schlüsselregeln im RE besagt, dass man dem Kunden das liefern muss, was er will, und nicht das, was er braucht. So flapsig das klingt – ein wahrer Kern steckt darin. Im Zweifelsfall zählt, was vertraglich abgestimmt wurde. Allerdings sollte ein Lieferant im Interesse einer nachhaltigen Kundenbeziehung im Vorfeld klären, was der Kunde braucht, um dann *vor* Projektbe-

ginn eine Abstimmung zu erreichen zwischen dem, was gebraucht und gewünscht (also vertraglich festgehalten) wird. Eine wirksame Basis für erfolgreiches Kundenmanagement ist es, zuallererst den Business Case des Kunden zu verstehen. Dabei geht es darum zu erkennen, was der Kunde – anders – machen will, wenn er erst einmal das gewünschte Produkt in Händen hält. Den Business Case zu verstehen, bedeutet, dass man als Produkt- oder Projektmanager versteht, welche Funktionen oder Anforderungen an das Projekt den größten Nutzen bringen. Man versucht, aus der späteren Anwendung innerhalb der Geschäftsprozesse des Kunden heraus einzuschätzen, welche Anforderungen kritisch sind und ob vielleicht bestimmte Einschränkungen übersehen wurden.

Risiko 3: Nur funktionale Anforderungen werden berücksichtigt

Anforderungen haben verschiedene Ausprägungen, wie wir gesehen haben. Neben den funktionalen Anforderungen gibt es nichtfunktionale Anforderungen. Neben den Produkthanforderungen gibt es auch Prozessanforderungen. Nur die Hinterfragung aller dieser Typen macht die Anforderungsdokumentation vollständig. Wichtig wird diese Vollständigkeit vor allem auch bei der Testspezifikation. Testfälle müssen alle diese Kategorien von Anforderungen abdecken.

Risiko 4: Unkontrollierte Änderungen von Anforderungen

Anforderungen, die sich ständig ändern und deren Änderungen nicht beherrscht werden, führen zu Kosten- und Terminüberschreitungen und reduzieren die Qualität. Häufig existiert eine gewisse Basis von Anforderungen, mit denen dann ein Projekt gestartet wird. Einige Punkte sind noch offen und sollen rasch geklärt werden. Da das Projekt läuft, hat der Kunde manches Mal kein großes Interesse mehr, diese Punkte zu klären. Schließlich könnte er bei der Abnahme davon profitieren, wenn nicht alles so läuft, wie abgesprochen, denn das ist die einmalige Chance, komplett neue Anforderungen als Kompensation für diesen Projektfehler kostenlos zu erhalten. Anforderungen ändern sich in beinahe jedem Projekt. Die Änderungsrate unterscheidet sich zwischen Projekten und ist abhängig vom Wiederholungsgrad der Technologie bei Lieferant und Benutzer und der Anwendung beim Benutzer. Zur Minderung dieses Risikos ist es wichtig, die Änderungsrate zu verfolgen. Zu bestimmten Meilensteinen muss die Änderungsmenge reduziert werden, um die nächste Phase erfolgreich durchlaufen zu können. Üblich ist es, mit einem Puffer zu arbeiten, der sowohl Schätzungenauigkeiten als auch Anforderungsänderungen abfangen kann. Eine weitere Maßnahme ist die sogenannte Rückwärtsplanung von der Übergabe an zurück ins Projekt, um zu erkennen, ab wann der kritische Pfad keine Parallelarbeit als Puffer mehr zulässt. Mancher Projektmanager und auch Kunde wird überrascht sein, wie früh im Projekt dies der Fall ist. Als Regel gilt, dass in der zweiten Projekthälfte nur noch sehr wichtige Änderungen zugelassen werden – in zeitkritischen Projekten bereits

früher als zur Hälfte. Eine dritte Maßnahme ist schließlich, Änderungen grundsätzlich nur zu diskutieren, wenn eine Analyse der Auswirkungen stattgefunden hat. Andernfalls verschwenden beide Seiten ihre Zeit. In diesem »Spiel« wird gern gepokert, nur um zu sehen, wie sich der Lieferant verhält. Oftmals genügt daher ein einfacher Projektplan, um zu zeigen, dass die vorgeschlagene Änderung Kosten und Projektdauer unzulässig erhöhen wird.

Risiko 5: Beschreibung von Anforderungen als Entwurf

Anforderungsbeschreibungen und Entwurfsbeschreibungen sind zwei grundlegend unterschiedliche Dinge. Das leuchtet jedem ein. Es geht bei den Anforderungen darum, was geliefert werden muss. Bei der Entwurfsbeschreibung geht es darum, wie die Lösung entwickelt wird. Trotzdem werden diese Was- und Wie-Perspektiven immer wieder vermischt. Häufig geschieht dies aus einer vermeintlichen Zeitersparnis heraus. Man beginnt Anforderungen zu notieren. Im Beschreiben kommen erste Ideen zu gegenseitigen Abhängigkeiten und zur möglichen Realisierung, die einfach mit der Anforderung zusammen notiert werden. Selbst wenn man dies in getrennten Teilen der Anforderung macht, sollte es klar sein, dass prinzipiell nie eine 1:1-Abbildung möglich ist. Die nächste Anforderung hat vielleicht überlappende Einflüsse und schon passt das Muster nicht mehr. Schlimm wird es vor allem, wenn sich Anforderungen später ändern oder gestrichen werden. Wohin mit der teilweisen Analyse, die vielleicht auch noch von anderen Anforderungen gebraucht wird? Hier gilt die klare Regel, immer zwei Spezifikationsdokumente zu halten, nämlich die Liste der Anforderungen (Lastenheft) und die Liste der Lösungsspezifikation (Pflichtenheft). Verfolgbarkeit durch eindeutige Bezeichner und eventuell eine angepasste Werkzeugunterstützung erlauben später, auch umfangreiche Änderungen schnell in trockene Tücher zu bekommen.

Risiko 6: Anforderungen werden nicht geprüft

Zu stark vereinfachte oder oberflächliche Spezifikationen resultieren später in fehlender oder falscher Funktionalität. Aus Zeitgründen und zur Vereinfachung werden Anforderungen häufig wortwörtlich von Kunden- oder Benutzerinterviews übernommen. Dies ist allerdings nur eine Rohfassung, die erweitert und präzisiert werden muss. Nehmen Sie sich die Zeit, Anforderungen mit Reviews und Inspektionen zu prüfen und prüfen zu lassen. Auf der Seite des Projekts erledigen das die Systemanalysten, Projektmanager, Produktmanager und Tester. Mehrdeutige Anforderungen bedeuten Mehrarbeit im gesamten Projekt. Gerade Tester finden sehr viele Ambivalenzen, die sonst erst sehr viel später entdeckt werden, wenn sie vielleicht vom Entwickler bereits falsch implementiert wurden.

Risiko 7: Perfektionierung von Anforderungen und Spezifikationen

»Gold Plating«, also Verschnörkelungen der Entwickler und Benutzer, bringt unnötige Funktionen, Verzögerungen und zu hohe Kosten ins Projekt. Entwickler versuchen oft, Anforderungen, die sie verstanden haben, mit Leben zu

füllen, und entwickeln weitere Funktionen, die nie vereinbart wurden. Im besten Fall passiert gar nichts, denn der Kunde wird sich hüten, solche verzichtbaren Funktionen zu würdigen. Im Normalfall allerdings wird diese Komplexität unbeherrschbar, weil ja keine Ressourcen dafür vorgesehen wurden. Jede weitere Funktion bringt Abhängigkeiten zu anderen Funktionen, Sonderfälle, Ausnahmesituationen – und damit zusätzlichen Entwicklungs-, Korrektur- und Testaufwand. Anforderungen sollen widerspiegeln, was der Kunde als relevant betrachtet. Wenn sich Kunde oder Benutzer nicht sicher sind, werden sie weggelassen. Das Gleiche gilt für Spezifikationsdokumente. Dies sind Dokumente, die straff beschreiben, was oder wie gearbeitet wird. Es sind keine detaillierten Entwurfsbeschreibungen. Hilfreich ist es, von vornherein mit verschiedenen Dokumenten zu arbeiten, sodass Entwurfsentscheidungen von Beginn an im richtigen Dokument – also der Architektur- oder der Entwurfsbeschreibung – dokumentiert werden, ohne den Umweg über ein Spezifikationsdokument, wo solche Informationen nicht hingehören. Im Unterschied zu diesen Verschnörkelungen sollten allerdings Ausnahmebehandlungen der regulären Anforderungen durchaus mit dem Kunden geklärt und als Erweiterung der Anforderung spezifiziert werden. Use Cases beispielsweise haben bereits in ihrem Template eine solche Ausnahmebehandlung vorgesehen. Wird die Behandlung von Ausnahmen nicht vorab abgestimmt, kann dies zu sehr verwickelten und inkonsistenten Realisierungen führen.

1.3 Der Business Case für Requirements Engineering

Die Einführung und systematische Umsetzung von RE erfordert Aufwand sowohl in der Entwicklung als auch an ihren Schnittstellen, also Produktmanagement, Produktmarketing und Vertrieb. Häufig wird dieser Aufwand als zu hoch und zu zeitraubend gesehen, sodass die Anforderungen weiterhin ad hoc in das Projekt hineinpurzeln und dort bruchstückhaft und mit vielen Nacharbeiten umgesetzt werden, bis einmal mehr das Projekt seine Ziele verfehlt oder abgebrochen werden muss. Aus unserer Beratungspraxis kennen wir das Dilemma: Verbesserungen in Methodik, Prozessen, Ausbildung und Werkzeugen werden nicht angegangen, da der Anfangsaufwand, um diese Verbesserungen anzustoßen, als zu hoch betrachtet wird. Daher wollen wir in diesem Kapitel die Nutzen eines systematischen RE quantitativ unterstreichen und vor allem konkrete Hinweise geben, wie Sie in Ihrer eigenen Umgebung den Nachweis führen können, dass sich der Aufwand für das RE lohnt. Eine weitaus umfangreichere Darstellung der ROI-Konzepte und zugrunde liegenden Projektaufwandsdaten findet sich in [Ebert2007a].

Systematisches RE bringt klare Vorteile für die Softwareentwicklung. Die zwei am häufigsten zitierten empirischen Studien untersuchten in Hunderten von Projekten den Zusammenhang von RE und dem Projekterfolg. Die Termintreue von Projekten wurde dabei als primärer Erfolgsfaktor betrachtet, denn in den meisten Branchen geht es heute aufgrund von Hyper-Wettbewerb und Time-to-

Profit auf der Kundenseite eines IT- oder Softwarelieferanten darum, Softwarelösungen präzise zum gewünschten Termin zu liefern [Hamel2007, Graham2005].

Die umfassendste Untersuchung zum Nutzen von RE kommt von Alcatel-Lucent. Über mehrere Jahre hinweg wurden in einer longitudinalen Feldstudie unterschiedliche Projektdaten systematisch erfasst und analysiert (Abb. 1–4) [Ebert2006a]. Gute Termintreue wird nur dann erreicht, wenn die vier folgenden Techniken gleichzeitig umgesetzt werden (Abb. 1–4, rechte Seite). Wurde eine oder mehrere dieser Techniken vernachlässigt, führte das sofort zu Terminverzug (Abb. 1–4, 0-3 Techniken eingesetzt).

- Ein aktives Kernteam mit Produktmanagement, Marketing, Entwicklung und Produktion, das das gesamte Projekt (oder das Produktrelease) von der Strategie bis zur Evolution steuert
- Konsequente Nutzung eines definierten Lebenszyklus für die Entwicklung mit Meilensteinen, Checklisten und Vereinbarungen
- Transparenz aller Projektvereinbarungen (z.B. Anforderungen) im Intranet
- Gemeinsame Anforderungsanalyse durch das Kernteam mit Produktmanagement, Marketing, Entwicklung und Produktion

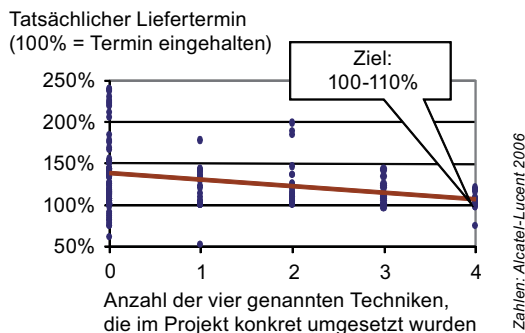


Abb. 1–4 Der gleichzeitige Einsatz von vier wesentlichen Techniken des RE (Kernteam, Lebenszyklus, Transparenz, gemeinsame Analyse) verbessert den Projekterfolg

Eine weitere umfassende Studie zum Nutzen von RE in Entwicklungsprojekten kommt von der NASA (Abb. 1–5) [Forsberg1997]. Der Zusammenhang ist auch in dieser Studie offensichtlich. Wenn die NASA in ihre Projekte weniger als 5 % Vorbereitungsaufwand (insbesondere auch Anforderungen) investiert, kommt es zu starken Verzögerungen. Bei einem Anteil von 10-20 % am gesamten Projekt-aufwand reduzieren sich die Terminverzögerungen auf unter 30 %.

Eine dritte longitudinale Feldstudie zum Nutzen von RE in IT-Projekten kommt ebenfalls von der NASA (Abb. 1–6) [Hooks2001]. Im Unterschied zu den beiden vorigen Studien wurde hier die Kosteneinhaltung in Abhängigkeit vom Aufwand für RE untersucht. Projekte mit 5 % Aufwand für RE führen zu einer Kostenüberschreitung von 80 % bis knapp 200 %. Wird dieser Aufwand in Rich-

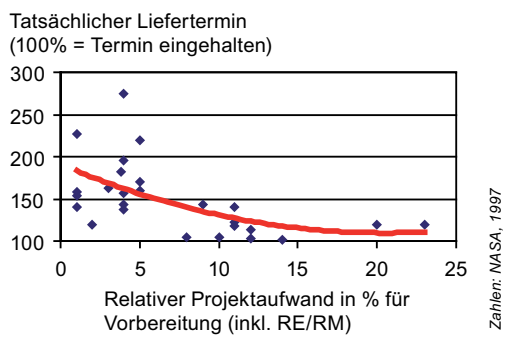


Abb. 1-5 Termineinhaltung in Abhängigkeit vom Aufwand für die Projektvorbereitung

tung 8-14% verdoppelt, liegt die Kostenüberschreitung bei unter 60%. Offensichtlich sind IT-Projekte sehr anfällig für eine unzureichende Anforderungsanalyse und -spezifikation, denn die Anforderungen werden sich im Projektverlauf zunehmend ändern und zu beträchtlichen Zusatzaufwänden führen. Auch hier gilt, dass die absoluten Zahlen für Überschreitungen von Kosten natürlich durch viele Faktoren bestimmt werden. Aber ein unzureichendes RE hat einen starken Anteil an überbordenden Kosten.

Das deckt sich auch mit einer Studie von Borland mit 348 IT-Verantwortlichen in den USA [Borland2006]. Über 90% der Antwortenden unterstrich, dass eine Verbesserung der RE-Prozesse einen klaren Wettbewerbsvorteil bringen würde. Mehr als die Hälfte der Antwortenden erklärte, dass mit einem besseren RE eine Senkung der Entwicklungskosten um 30% möglich wäre.

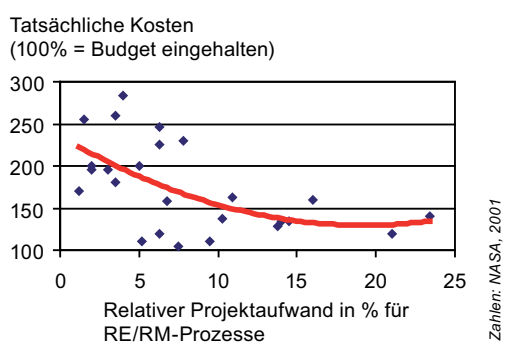


Abb. 1-6 Kosteneinhaltung in Abhängigkeit vom Aufwand für RE

Der Nutzen eines guten RE kann an verschiedenen Faktoren festgemacht werden. Im Folgenden geben wir Anhaltspunkte für die eigene Nutzenrechnung, die wir aus verschiedenen Kundenprojekten in unterschiedlichen Branchen konkret realisiert hatten [Ebert2007a, Standish2003, Stevens1998, Leffingwell1997].

- Produktivitätsverbesserung. Typischerweise werden 30-50 % des Entwicklungsaufwands für Fehlerbehebung und nicht entwicklungsbezogene Aktivitäten eingesetzt. Die Hälfte der Fehler kommen direkt aus unzureichenden Anforderungen und unkontrollierten Änderungen. Im Systemtest sind es 80 % der Fehler, die von unvollständigen (31 %) oder falschen (49 %) Anforderungen resultieren.
- Verbesserte Projektplanung und Ressourceneinteilung, weniger Verzögerungen vor Projektstart, eine schnellere Anlaufphase sowie Termintreue aufgrund von bekannten Anforderungen und klaren Verantwortungen im Projektteam und im Vertrieb. Mehr Aufwand für Entwicklung und konsequente Umsetzung und Test der Anforderungen schaffen eine Verbesserung der Termintreue und reduzieren Verzögerungen auf unter 20 %.
- Kürzere Durchlaufzeiten durch Fokus auf jene Aktivitäten und Inhalte, die Wert schaffen. Über die Hälfte aller Funktionen eines Softwaresystems werden nicht genutzt. Das gilt sowohl in der IT wie auch in technischen Produkten. Weniger Anforderungen reduzieren die Komplexität und machen damit die Projekte verlässlicher sowie die Qualität besser.
- Weniger Nacharbeiten von inkonsistenten Anforderungen durch Einflussanalyse bei sich ändernden Anforderungen und Verfolgbarkeit zu den betroffenen Arbeitsergebnissen. Werden die Anforderungen so umgesetzt, wie sie vom Kunden oder Benutzer beschrieben werden, führt das zu Nacharbeiten, die im Schnitt 45 % des Projektaufwands ausmachen.
- Wiederverwendung von Anforderungen und davon abgeleiteten Arbeitsergebnissen (z.B. Mechanismen zur Systemsicherheit)
- Bessere Kundenzufriedenheit durch konsistentes Verständnis über die wirklichen Anforderungen

Einige dieser Faktoren, wie Termintreue oder weniger Nacharbeiten, schaffen einen unmittelbar greifbaren Nutzen. Andere, wie beispielsweise die Kundenzufriedenheit, sind eher opportunistisch und werden in Form nachhaltig guter Kundenbeziehungen und weiterer Projekte greifbar. Insgesamt zeigen unsere Erfahrungen, dass eine Verdoppelung des Aufwands für RE hin zu 10% des Projektaufwands in den Bereichen Methodik, Prozesse, Training und Werkzeugunterstützung konkret realisierbare Projektnutzen von über 20 % schafft. Das ist ein ROI von mehr als 4, und damit sind nur die direkt messbaren Vorteile berücksichtigt.

1.4 Eine kurze Übersicht des Buchs

Abbildung 1–7 zeigt die Struktur des Buchs. Das Thema RE wird zunächst anhand verschiedener Übersichtskapitel eingeführt. In Kapitel 2 werden die wichtigsten Begriffe sowie die relevanten Standards in einem Zusammenhang beschrieben. Die Kapitel 3 und 4 erläutern das Vorgehen im Requirements Engineering aus verschiedenen Sichten. Zuerst wollen wir die Methodik und Systematik einführen. Dann werden Aufgaben und Rollen und ihr Zusammenhang zum RE charakterisiert. Dabei nehmen wir bewusst eine umfassende Perspektive ein und wollen uns auch mit den angrenzenden Rollen (z.B. des Produktmanagers) auseinandersetzen.

Die Kapitel 5 bis 10 beleuchten die einzelnen Aktivitäten innerhalb des RE systematisch. Kapitel 5 beschreibt, wie Anforderungen ermittelt werden. Wir wollen bewusst nicht vom Sammeln sprechen, denn häufig sind die Anforderungen unbekannt und müssen mühevoll herausgeschält und entwickelt werden. Dieses Kapitel ist ausdrücklich kundenorientiert und soll den Lesern unter Ihnen, die eher die Sicht des Einkäufers oder Kunden einnehmen, dabei helfen, diesen Prozess aus den verschiedenen Perspektiven zu verstehen. Kapitel 6 betrachtet die Spezifikation, also das Beschreiben von Anforderungen. Dabei geht es um die Verbesserung der Anforderungsqualität und um verschiedene formale Arten der Beschreibung, die hinsichtlich ihrer Praktikabilität und Schwierigkeit in der Umsetzung diskutiert werden. Kapitel 7 beleuchtet die Validierung der Anforderungen. Häufig werden die falschen Anforderungen realisiert oder Fehler in der Umsetzung gemacht. Wir zeigen hier Techniken zu Reviews, Prüfungen und konkrete Checklisten, um die Anforderungsqualität zu verbessern. Kapitel 8 beschreibt verschiedene Analyseverfahren, wobei wir ein einheitliches Beispiel einsetzen, um die Unterschiede und Gemeinsamkeiten besser zeigen zu können. Kapitel 9 greift eine oft vernachlässigte Aktivität im Projekt auf, nämlich die Vereinbarung von Anforderungen. Das Kapitel betrachtet »Gesetze« vor dem Hintergrund des Bürgerlichen Gesetzbuchs und seiner Einflüsse auf die Softwareentwicklung. Wir wollen hier kurz die wichtigsten vertraglichen Aspekte skizzieren, vor allem aus der Sicht von Gewährleistungs- und Haftungsfragen. Die Konsequenzen sind hinreichend bekannt und führen dazu, dass verschiedene Interessengruppen eine unterschiedliche Sicht auf das Problem und die Lösung haben und diese nachher mit ziemlicher Sicherheit auch durchfechten wollen. Schließlich beschreibt das Kapitel 10 die Methoden der Anforderungskontrolle und -verfolgung. Während die bisherigen Aktivitäten vorzugsweise vor Start des Projekts durchgeführt und abgeschlossen werden, beschreibt dieses Kapitel all jene Tätigkeiten, die erst im Projekt so richtig interessant werden. Es geht dabei um das Änderungsmanagement, die vertikale und horizontale Verfolgung, Maßzahlen, Komplexitätsbeherrschung und den Test.

Die vielfältigen Werkzeuge, die im RE zum Einsatz kommen, werden in Kapitel 11 charakterisiert und beschrieben. Zuerst gibt dieses Kapitel eine Über-

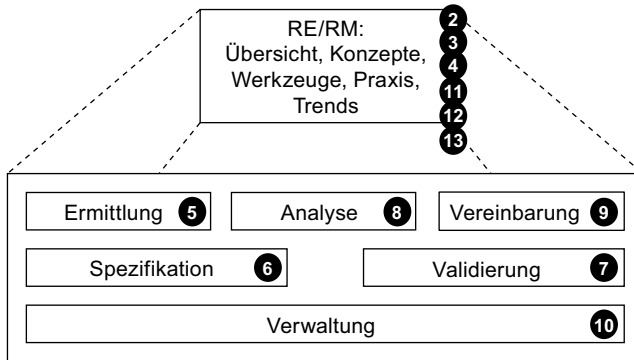


Abb. 1-7 Zuordnung der Buchkapitel zu den Themen des RE/RM (schwarze Kreise sind Kapitelnummern)

sicht, in welchen Situationen welche Werkzeuge eine Unterstützung bieten. Daran schließen sich einige Fallstudien an, die gemeinsam mit Werkzeugherstellern für dieses Buch erarbeitet wurden. Sie dienen nicht der Übersicht über ein spezifisches Werkzeug, sondern zeigen namhafte RE-Werkzeuge in unterschiedlichen Szenarien. Diese verschiedenen Szenarien, die die Erfassung von Anforderungen bis hin zu Verfolgung und Änderungsmanagement abdecken, helfen Ihnen als Leser dabei, sich ein praktisches Urteil davon zu bilden, wie Sie die Werkzeuge einsetzen und welchen Nutzen Sie daraus ziehen können.

Ein Buch aus der Praxis und für die Praxis braucht natürlich auch noch ein dezidiertes »Praxiskapitel«, das all jene Themen aufgreift, die in die bisherige Struktur nicht passten und eher integrierenden Charakter haben. Dies geschieht in Kapitel 12, wo Praxiserfahrungen aufgegriffen werden.

Das abschließende Kapitel 13 fasst den Stand der Technik nochmals zusammen und beleuchtet die wichtigsten Trends des RE in den nächsten Jahren. Hier werden auch die empirischen »Gesetzmäßigkeiten« des RE nochmals an einer Stelle zusammengefasst. Das mag etwas abgehoben klingen in einer Disziplin, die so viele kunden- und projektspezifische Eigenarten hat. Aber genau so, wie auch die Vorgehensweise strukturiert und systematisch beschrieben werden kann, gibt es einige Regelmäßigkeiten, die mit empirischen Studien aus zahlreichen abgeschlossenen Projekten abgeleitet wurden. Dieser Stand der Technik gewinnt an Bedeutung im Projektgeschäft, wenn Sie nachweisen müssen, dass Sie die relevanten Techniken des RE auch wirklich einsetzen. Das kann in Gerichtsverfahren eine Rolle spielen, wenn es zu Schadensersatz- oder Produkthaftungsfragen kommt. Das Kapitel weist aus einer positiven Sichtweise auf, wie sich die ganze Disziplin des Software-Engineering (oder der Softwaretechnik) voraussichtlich weiterentwickelt. Anstatt Angst vor Outsourcing zu schüren, wollen wir an der Stelle lieber betrachten, welche großen Betätigungsfelder alleine das RE eröffnet, wenn die Disziplin nur sauber beherrscht wird.

Abgerundet wird das Buch durch eine Zusammenstellung von Internetressourcen, also den wichtigsten URLs zum Thema RE. Diese URLs können sich

natürlich ändern, aber die beschriebene Auswahl hat sich in den vergangenen Jahren als recht stabil erwiesen.

Alle Begriffe, die im Buch definiert werden oder auf deren Definitionen zurückgegriffen wird, sind im Glossar am Ende des Buchs nochmals zusammengefasst. Eine Zusammenstellung der Literaturquellen sowie ein Index runden das Buch ab.

Jedes der Kapitel besitzt am seinem Ende mehrere Unterkapitel mit jeweils gleichem Titel, die sich wie ein roter Faden durch das gesamte Buch ziehen. Zunächst liefert ein Unterkapitel jeweils angepasste Checklisten zum jeweiligen Thema. Ein weiteres Unterkapitel fasst kurz und prägnant alle wichtigen Praxistipps zusammen. Diese Zusammenstellung spricht die Sprache des Praktikers und wurde so ausgewählt, dass Sie als Leser und Benutzer einzelne Elemente herausgreifen können und direkt in Ihrem Projektgeschäft und Ihrer Tagesarbeit umsetzen können. Ein jeweils letztes Unterkapitel stellt einige Fragen an die Praxis. Damit können Sie das gerade Gelesene in Ihrem eigenen Kontext reflektieren und leichter umsetzen. Es handelt sich also nicht um die Art von Verständnisfragen, wie Sie es aus Lehrbüchern kennen, sondern eher um Fragen, die Ihren eigenen Horizont öffnen, um das gerade konsumierte Wissen zu verdauen und genau das abzuleiten, was Ihre eigenen Projekte konkret benötigen.

1.5 Einführung in das durchgängige Beispiel

Ein Buch für die Praxis braucht ein praktisches Beispiel. Dieses Beispiel soll möglichst alle Facetten eines realen Produkts abbilden, also beispielsweise IT, eingebettete Systeme, nichtfunktionale Anforderungen verschiedener Art und Wartungsaufgaben. Gewählt haben wir dazu eine funktionsfähige Personenaufzugsanlage, die am Institut für Automatisierungs- und Softwaretechnik der Universität Stuttgart steht. Aufgrund der vielfältigen Anwendungsmöglichkeiten verbindet das Beispiel sowohl Aspekte der IT-Welt als auch eingebetteter Systeme. Die Anlage besteht aus zwei Aufzugsschächten mit jeweils vier Stockwerken und erlaubt eine unabhängige Personenbeförderung in beiden Schächten. Abbildung 1–8 zeigt die Aufzugsanlage.

Zur Übersicht werden die zentralen Funktionen kurz beschrieben. Fahrgäste drücken in einem der vier Stockwerke einen Rufknopf mit Richtungsvorwahl. Eine der beiden Kabinen fährt das Stockwerk an und öffnet die Türe. Innerhalb der Kabine wird das Zielstockwerk vorgewählt, das dann angefahren wird. Die aktuelle Stockwerknnummer wird in der Kabine dargestellt. Bei gleichzeitiger Benutzung des Aufzugs durch verschiedene Fahrgäste werden die beiden Kabinen in ihrer Fahrt so optimiert, dass nur kurze Wartezeiten entstehen. Aus Sicherheitsgründen sind verschiedene Schutzfunktionen eingebaut, wie beispielsweise Rauchmelder in den Kabinen, eine Notstromunterstützung für einen etwaigen Stromausfall sowie Türverriegelungen, um ein unbeabsichtigtes Öffnen während der Fahrt zu vermeiden.

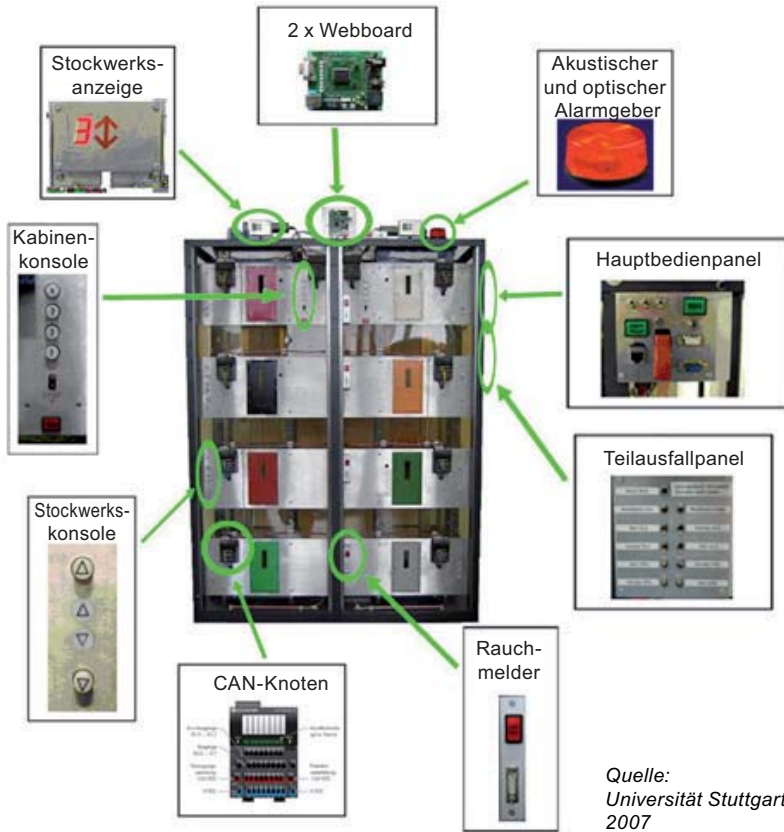
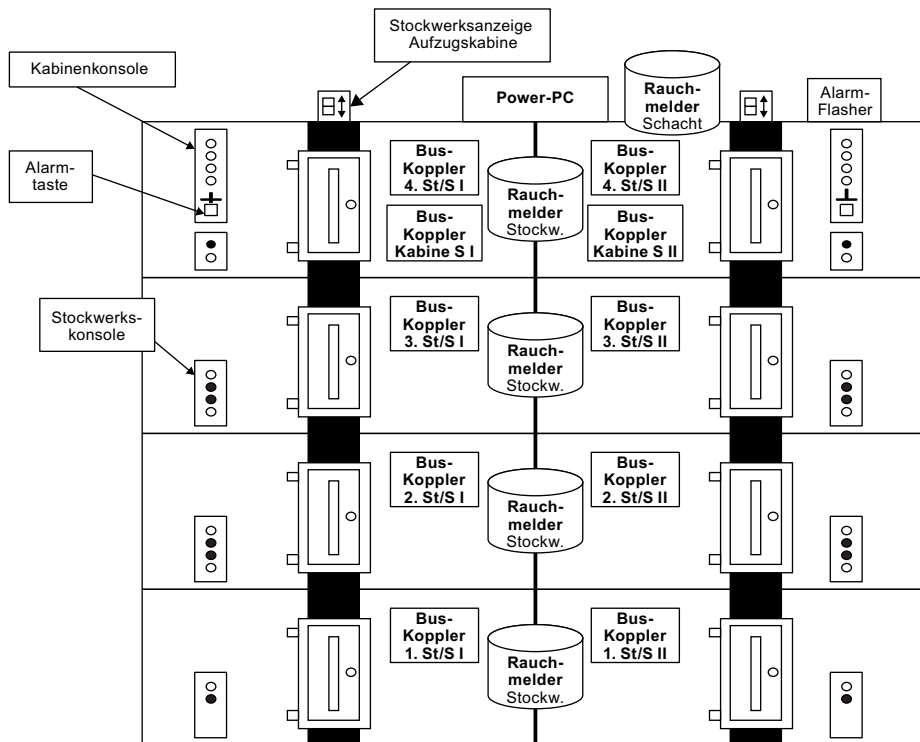


Abb. 1-8 Personenaufzug als betriebsfähiges Modell aufgebaut

Abbildung 1-9 zeigt die Systemarchitektur des Personenaufzugs. Der dargestellte Embedded-PowerPC übernimmt sämtliche Steuerungsaufgaben des Modellprozesses. Als Betriebssystem kommt auf dem Steuerungsrechner Linux zum Einsatz. Bis auf wenige Ausnahmen werden alle Sensor- und Aktordaten über den zentralen CAN-Bus mit dem Aufzugsmodell ausgetauscht. Die Kabinen selbst sind als aktive CAN-Knoten realisiert. Die Positionserkennung der Kabinen im Schacht erfolgt mittels Mikroschaltern. Um eine reale Anlage nachbilden zu können, wurden viele Bedienelemente eines realen Aufzugs (wie zum Beispiel Wahltasten für das Zielstockwerk, Stoppschalter usw.) in das Aufzugsmodell integriert. Die Stromversorgung erfolgte über ein in das Modell integriertes Netzteil. Eine Machine-to-Anywhere (M2A)-Schnittstelle erlaubt die Fernwartung. Soll eine Fernsteuerung bzw. Ferndiagnose des Modellprozesses durchgeführt werden, so muss ein Fernwartungsclient vom PowerPC aus über Ethernet erreichbar sein. Die Fernsteuerung bzw. Ferndiagnose erfolgt dann über ein Benutzerschnittstellenprogramm, das auf dem zu verwendenden Endgerät installiert wird. Neben dem CAN-Bus existieren noch zwei weitere Signalleitungen zwischen dem Pro-



Quelle: Universität Stuttgart, 2007

Abb. 1-9 Systemarchitektur des Personenaufzugs

zessmodell und dem PowerPC, die die Signalstatus-Stromversorgung und Notstromversorgung aktivieren. Diese sind aus Gründen der Betriebssicherheit nicht in den CAN-Bus integriert.

Das Beispiel wird an verschiedenen Stellen aufgegriffen, um Aspekte des RE zu konkretisieren, beispielsweise für die Beschreibung oder Modellierung einer funktionalen Anforderung oder einer Sicherheitsanforderung.

1.6 Wie Sie von diesem Buch profitieren

Dies ist ein Buch für Einsteiger und Profis. Nach der Lektüre

- haben Sie einen Überblick über Theorie und Praxis des Requirements Engineering;
- haben Sie einen ersten Einblick, wie moderne Werkzeuge und Notationen Sie beim Requirements Engineering praktisch unterstützen können;
- können Sie die wichtigen Elemente des Requirements Engineering in Ihren Projektalltag übertragen und dort produktiv einsetzen.

Soweit Sie als Softwareentwickler in einem Unternehmen arbeiten, gibt Ihnen dieses Buch einen guten Überblick zu den relevanten Fragestellungen und Lösungen

des Requirements Engineering. Praktische Tipps am Ende jedes Kapitels helfen Ihnen dabei, essenzielle Vorgehensweisen schnell und »leicht verdaulich« zu extrahieren. Die Techniken sind praxiserprobt und leicht umsetzbar. Requirements Engineering klingt zwar nach Formalismus und wird häufig in einen Topf geworfen mit Aufwandschätzung, Modellierung und Projektmanagement. Das Buch versucht, die Themen sauber zu trennen und trotzdem einen Querbezug zu schaffen.

Falls Sie Software in Eigenregie entwickeln, liefert das Buch die Basis für ein funktionierendes Requirements Engineering. Es spricht die wichtigsten Grundlagen an, die selbst in sehr kleinen Projekten eine große – oft überlebensnotwendige – Rolle spielen. Da es sehr klar zwischen Prozessen und Werkzeugen trennt, werden Sie lernen, wie Sie mit einfachsten Mitteln eine solide Basis Ihrer Anforderungen halten und verwalten. Die angesprochenen Prinzipien sind skalierbar und nicht nur für große Projekte relevant. Beispielsweise braucht auch ein Kleinstprojekt ein funktionierendes Änderungsmanagement, um zu verfolgen, welche Anforderungen im Moment akzeptiert sind und welche sich noch in der Pipeline befinden. Wenn Sie einem Kunden einen Termin auf der Basis von gegenseitig vereinbarten Anforderungen versprochen haben, werden sie es häufig erleben, dass kurze Zeit später die ersten Änderungen kommen. Dies ist normal, muss aber konsequent abgefangen werden. Wenn Sie einmal damit beginnen, solche Änderungen auf Zuruf zu akzeptieren, weil sie ja »sehr klein« sind, haben Sie eine Tür aufgemacht, die Sie nie mehr bei diesem Kunden schließen können. Auf welcher Basis würden Sie argumentieren, dass manche Änderungen »sehr klein« und andere »nicht mehr so klein« sind?

Besser ist es, von vornherein einen Prozess zu vereinbaren, der besagt, dass alle Anforderungen in einer Liste gepflegt werden. Falls es zu Änderungen kommt, werden diese analysiert und dann vereinbart. Änderungen können Auswirkungen auf Termine und Kosten haben. Falls Sie zu einem Festpreis arbeiten, wird es Ihnen manchmal sogar helfen, eine späte Änderung noch aufzunehmen, da Sie damit Termine und den Preis nochmals beeinflussen können. Es gibt keine Änderung ohne vorherige Abschätzung und formalisierte Vereinbarung. Danach finden sich die Anforderungen auf Ihrer Liste wieder. Das geht mit wenig Aufwand, und die Kunden werden Ihre Professionalität schätzen.

Das Buch empfiehlt wirksame Prinzipien und Vorgehensweisen, die sich an der Praxis orientieren. Es wird kaum auf Gegenliebe stoßen, wenn Sie von Ihren Kunden verlangten, dass sich die Anforderungen nicht mehr ändern dürfen. Anforderungen sind naturgemäß Änderungen unterworfen. Also müssen beide Seiten damit umgehen können. **Wichtig ist, dass man die Änderungen konsistent bearbeitet und erst dann Zusagen macht, wenn man sie auch einhalten kann.**

Falls Sie Projekte leiten, bietet sich das vorliegende Buch sowohl als Einführung in das Thema als auch als weiterführende Literatur an, die spezielle Bereiche vertieft. Sie finden eine Menge wertvoller Tipps und lernen, wie Requirements Engineering konkret implementiert wird und wie Risiken und typische Schwierigkeiten im Requirements Engineering gehandhabt werden können. Dieses Buch

bietet eine Menge an konkreten Tipps aus dem Projektalltag, die der Autor in seiner beruflichen Praxis gesammelt hat. Fallstudien und Praxiserfahrungen am Ende des Buchs (Kapitel 12) greifen die wichtigsten Themen nochmals aus der Sichtweise konkreter betrieblicher Fragestellungen auf. Sie helfen bei der »Übersetzung« des Themas Requirements Engineering und der zugehörigen Lösungen in Ihren eigenen betrieblichen Alltag.

Sind Sie in der Systementwicklung tätig, ohne überhaupt Software zu betrachten? Wir empfehlen das Buch auch für andere Branchen, wo RE gebraucht wird, aber keine spezifische Methodik zur Verfügung steht. Die hier vorgestellten Methoden und Prozesse sind größtenteils nicht Software- oder IT-spezifisch, sondern universell in der Systemtechnik einsetzbar.

Als Requirements-Ingenieur und als Produktmanager zeigt Ihnen das Buch, wie Sie erfolgreich eine Brücke bauen zwischen den sehr verschiedenen Sichtweisen heterogener Interessengruppen innerhalb und außerhalb des Projekts. Zielkonflikte tragen zu besseren Lösungen bei und sollten nicht sofort zu Konfrontationen führen. Wir unterstützen Sie mit Methoden und Tipps, um solche verschiedene Ziele und Perspektiven herauszuarbeiten, zu verstehen, daraus die richtige Balance der Anforderungen zu ermitteln und damit Win-win-Ergebnisse zu erreichen.

Soweit Sie als Qualitätsverantwortlicher oder im Bereich der Prozessverbesserung arbeiten, hilft Ihnen das Buch, die verschiedenen Modelle (z.B. ISO 9001, CMMI, SPICE) praktisch einzusetzen. Der Autor hat selbst viele Jahre damit verbracht, unterschiedliche Unternehmen und Produktlinien im Anforderungs- und Produktmanagement zu verbessern. Wir wollen uns hier vor allem auch mit der Messbarkeit und Verfolgung von Anforderungen im Lebenszyklus befassen – ein Thema, das in den meisten Büchern zum Requirements Engineering zu kurz kommt.

Neulinge im Thema Requirements Engineering und Studierende der Informatik oder der Softwaretechnik (Software-Engineering) werden von den ersten Kapiteln stark profitieren, denn dort betten wir das Thema in einen größeren Bezug zu anderen Prozessen der Softwaretechnik ein. Fragen an die Praxis am Ende eines jeden Kapitels helfen dabei, sich selbst zu prüfen und zu erkennen, ob die relevanten Themen auch im Kontext verstanden wurden. Diese Fragen beziehen sich daher nicht ausschließlich auf das Kapitel, in dem sie auftreten. Oftmals bauen sie auch auf vorangegangenen Themen auf.

Beiden Gruppen, den Einsteigern und den Praktikern, gerecht zu werden, gelingt durch eine klare Struktur, die in jedem Kapitel wichtige Themen zusammenfasst. So wissen Einsteiger, was gemeint ist, und können sich orientieren, während die Profis sofort in die Tiefe gehen und finden können, was ihre derzeitige Situation gerade verlangt.

1.7 Ein Blick über den Tellerrand

Dieses Buch fokussiert auf die systematische und zielorientierte Umsetzung von RE in der Praxis. Daher können wir nicht alle Themen detaillieren, wie dies ein Grundlagenwerk tun kann. Wir empfehlen daher – als Blick über den Tellerrand des Tagesgeschäfts hinaus – einige weitere Bücher, die solche Grundlagen liefern. Eines können wir versprechen: Jedes der im folgenden genannten Bücher gibt Ihnen weitere Impulse und Ideen.

Der Rahmen, in den RE eingebettet ist, nämlich die Softwaretechnik, deren Vorgehensweisen und Managementprinzipien, ist im Buch von Balzert beschrieben [Balzert2008]. Das Buch stellt die relevanten Managementtechniken und Vorgehensmodelle vor und beschreibt, wie verschiedene Modelle in der Praxis eingesetzt werden. Verschiedene Schwerpunktthemen, wie strategisches Management in der IT und globale Softwareentwicklung, verdeutlichen die heutige Ausrichtung der Softwaretechnik.

Die Grundlagen des RE sind umfassend im umfangreichen Werk von Klaus Pohl beschrieben [Pohl2006]. Viele Themen, beispielsweise Notationen und Methoden, die wir hier aus Platzgründen und aufgrund der umsetzungsorientierten Ausrichtung dieses Buchs nicht vertiefen können, werden dort auf eine saubere Basis gestellt. Wir empfehlen sein Buch als Ergänzung zum Nachschlagen von Prinzipien, die Sie grundlegend einordnen und verstehen wollen.

Eine gute Ergänzung zur Erreichung von Win-win-Ergebnissen im RE ist das hervorragende Buch von Al Davis [Davis2005]. Analog unserer Philosophie beschreibt er, wie man RE so umsetzt, dass eine Balance zwischen »hinreichend guten« Ergebnissen erreicht wird, ohne zu viel Overhead zu erzeugen.

Wie gute Anforderungen beschrieben werden, zeigt uns Ian Alexander in [Alexander2002]. Chris Rupp verrät sprachliche Besonderheiten und erklärt, auf was Sie in Spezifikationstexten zu achten haben [Rupp2006]. Beide Bücher adressieren die Spezifikationsphase und liefern viele Beispiele zur Darstellung von funktionalen und nichtfunktionalen Anforderungen.

Eine Vertiefung der Notationen UML und SysML ist in [Weilkiens2006] zu finden. Falls Sie sich für die geschichtliche Entwicklung von Analyse- und Spezifikationssprachen interessieren, empfehlen wir zusätzlich [Sommerville1998]. Dieses Buch geht sehr viel stärker auf Notationen und Modellierungstechniken ein, als wir dies hier aus Platzgründen tun können.

An verschiedenen Stellen des Buchs unterstreichen wir, dass RE als Disziplin so spannend und in der praktischen Software- und Systementwicklung so unheimlich wichtig ist, weil man mit Menschen arbeitet und gemeinsam zielorientiert die Grundlagen für immer wieder neue Produkte und Geschäftserfolge schafft. Dazu braucht es sehr viel mehr als die Beherrschung von Notationen und Werkzeugen. Es geht um die Fähigkeit, mit anderen Menschen gemeinsam erfolgreich zu sein. Dazu gehören »Soft Skills«, die aus der Sicht der Softwareentwicklung im

Buch von Vigenschow und Schneider [Vigenschow2007] dargestellt sind. Wir empfehlen dieses Buch, um den Blick über den Tellerrand etwas zu verbreitern und um sich selbst weiterzuentwickeln.