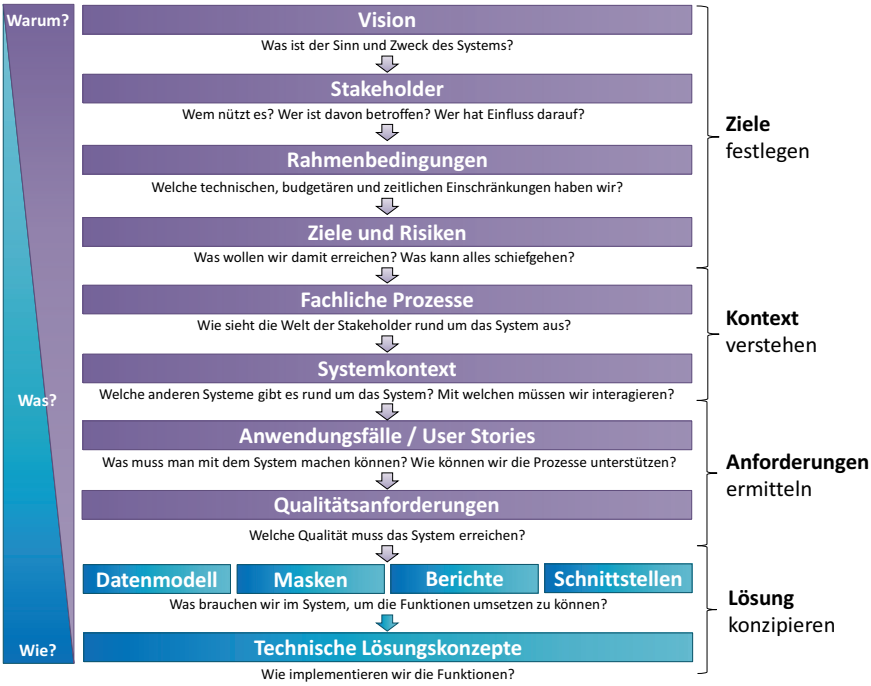
2. Auflage

Markus Unterauer

Workshops im Requirements Engineering

Methoden, Checklisten und Best Practices für die Ermittlung von Anforderungen

dpunkt.verlag


Warum?
Was?
Wie?
Vision
Was ist der Sinn und Zweck des Systems?
Stakeholder
Wem nützt es? Wer ist davon betroffen? Wer hat Einfluss darauf?
Rahmenbedingungen
Welche technischen, budgetären und zeitlichen Einschränkungen haben wir?
Ziele und Risiken
Was wollen wir damit erreichen? Was kann alles schiefgehen?
Fachliche Prozesse
Wie sieht die Welt der Stakeholder rund um das System aus?
Systemkontext
Welche anderen Systeme gibt es rund um das System? Mit welchen müssen wir interagieren?
Anwendungsfälle / User Stories
Was muss man mit dem System machen können? Wie können wir die Prozesse unterstützen?
Qualitätsanforderungen
Welche Qualität muss das System erreichen?
Datenmodell
Masken
Berichte
Schnittstellen
Was brauchen wir im System, um die Funktionen umsetzen zu können?
Technische Lösungskonzepte
Wie implementieren wir die Funktionen?
Ziele festlegen
Kontext verstehen
Anforderungen ermitteln
Lösung konzipieren

Markus Unterauer hat Wirtschaftsinformatik studiert. In seiner Berufspraxis war er in vielen Bereichen der Softwareentwicklung, wie Architektur, Entwurf, Entwicklung, Testen und Testautomatisierung, tätig. Er lernte dabei sowohl klassische als auch agile Methoden intensiv kennen.

Seit 2012 arbeitet Markus Unterauer bei Software Quality Lab als Berater und Trainer. Er ist zertifizierter Requirements Engineer und Scrum Master und hat sich auf die Bereiche Softwareprozesse und Anforderungsmanagement spezialisiert. Markus Unterauer ist als Vortragender in diesen Themenbereichen immer wieder auf Konferenzen tätig.

Markus Unterauer

Workshops im Requirements Engineering

Methoden, Checklisten und Best Practices für die Ermittlung von Anforderungen

2., überarbeitete und erweiterte Auflage

dpunkt.verlag

Markus Unterauer
m.unterauer@gmail.com
markus.unterauer@software-quality-lab.com

Lektorat: Christa Preisendanz
Copy-Editing: Ursula Zimpfer, Herrenberg
Satz: Birgit Bäuerlein
Herstellung: Stefanie Weidner
Umschlaggestaltung: Helmut Kraus, *www.exclam.de*
Druck und Bindung: mediaprint solutions GmbH, 33100 Paderborn

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.

ISBN:
Print 978-3-86490-695-4
PDF 978-3-96088-902-1
ePub 978-3-96088-903-8
mobi 978-3-96088-904-5

2., überarbeitete und erweiterte Auflage 2020

Wieblinger Weg 17
69123 Heidelberg

Hinweis:
Dieses Buch wurde auf PEFC-zertifiziertem Papier aus nachhaltiger Waldwirtschaft gedruckt. Der Umwelt zuliebe verzichten wir zusätzlich auf die Einschweißfolie.

Schreiben Sie uns:
Falls Sie Anregungen, Wünsche und Kommentare haben, lassen Sie es uns wissen: *hallo@dpunkt.de*.

5 4 3 2 1 0

Vorwort zur 2. Auflage

Requirements Engineering ist toll! Seit fast 20 Jahren schreibe ich nun Spezifikationen, seit acht Jahren helfe ich meinen Kunden als Berater und Trainer, in ihren Projekten die richtigen Anforderungen herauszufinden und so an die Entwicklungsteams weiterzugeben, dass diese daraus ein tolles Produkt entwickeln können, mit dem sie ihre Kunden nicht nur zufriedenstellen, sondern begeistern.

Die erfolgreichsten Projekte waren dabei immer diejenigen, bei denen nicht einfach Anforderungen in einer endlosen Reihe mühsamer Meetings besprochen wurden, sondern bei denen die Anforderungsermittlung als kreativer Prozess gestaltet war und Anwender, Kunden und das Entwicklungsteam aktiv an den Anforderungen gearbeitet und diese mit unterschiedlichsten Methoden in Workshops regelrecht erforscht haben.

Aus dieser Erfahrung heraus entstand 2014 die erste Auflage dieses Buches. Ich beschreibe darin kurz und kompakt ein durchgängiges Konzept, wie man in Workshops Anforderungen gemeinsam erarbeitet, verfeinert und für die Weitergabe ans Team aufbereitet.

Mittlerweile sind einige Jahre vergangen. Agile Methoden haben sich in der Softwareentwicklung längst durchgesetzt und strahlen in die Hardwareentwicklung aus. Immer häufiger arbeite ich mit Product Ownern und agilen Teams daran, die Anforderungen ihrer Kunden besser zu verstehen und schneller in Produkte umzusetzen, die oft schon mit kleinen Features echten Mehrwert für die Anwender schaffen. Ich finde das immer noch grandios! Gerade für agile Teams gibt es mittlerweile viele sehr gute Methoden für das Requirements Engineering: User Stories werden in Impact Maps aus Zielen abgeleitet. In einer Story Map strukturiert das Team die vielen Anforderungen und stellt diese zu Releases zusammen, die Schritt für Schritt Wert schaffen. Einfache Priorisierungsmethoden erlauben es dem Team, den Kunden immer im Blick zu behalten. Der Schlüssel dafür ist das ständige gemeinsame Arbeiten an den Anforderungen und das Ringen um die beste Lösung.

All diese neuen Methoden und Erfahrungen sind in die zweite Auflage dieses Buches eingeflossen. Es enthält dazu ein großes neues Kapitel zu agilem Requirements Engineering mit User Stories. In diesem sind viele Workshop-Methoden für

Anforderungserhebung, Refinement, Priorisierung und Schätzung in agilen Teams beschrieben. Neben diesem neuen Schwerpunkt für Workshops in agilen Projekten habe ich auch die anderen Kapitel mit der Erfahrung der letzten Jahre überarbeitet. Viele Checklisten wurden erweitert, Grafiken angepasst, sodass sie noch deutlicher demonstrieren, wie man das Gezeigte in Workshops einsetzen kann, und zahlreiche kleine Tipps und Tricks aus der Erfahrung in der praktischen Anwendung in meinen Projekten wurden eingearbeitet.

Ich freue mich, wenn dieses Buch Sie als Projektleiter, Product Owner, Produktmanager oder Teammitglied unterstützt und Sie darin weiterbringt, noch besser zu verstehen, was für Ihre Anwender ein wertvolles Produkt ausmacht.

Für Feedback bin ich jederzeit sehr dankbar, vielleicht haben Sie ja Lust, Ihre Erfahrungen zu teilen oder zu diskutieren ...

am besten per Mail an *m.unterauer@gmail.com*.

Auf jeden Fall wünsche ich Ihnen viel Erfolg beim Ausprobieren und Anwenden!

Markus Unterauer
Linz, im August 2019

Vorwort zur 1. Auflage

In jedem Softwareprojekt müssen wir Anforderungen systematisch ermitteln, dokumentieren, prüfen und abstimmen sowie verwalten. Diese Aufgaben werden unter dem Begriff »Requirements Engineering« zusammengefasst [Pohl & Rupp, 2015]. Dieses Buch beschäftigt sich mit dem ersten Schritt, dem Ermitteln von Anforderungen. Als Requirements Engineer bin ich am Ende dann erfolgreich, wenn die gebaute Software es den Anwendern ermöglicht, ihre Ziele zu erreichen und ihre Probleme zu lösen. Dies kann nur gelingen, wenn die Basis dafür stimmt, also die richtigen Anforderungen vollständig ermittelt und verstanden wurden.

Warum ein weiteres Buch zum Ermitteln von Anforderungen?

Es gibt bereits viele Bücher, die das Thema Requirements Engineering sehr umfassend behandeln und gut strukturieren. In diesem Buch möchte ich wichtige Fragen aus der praktischen Arbeit in Workshops beantworten:

- Wie gestalte ich die Workshops zur Ermittlung der Anforderungen?
- Wie ermittle ich in meinem Projekt die Anforderungen wirklich?
- Wie moderiere ich solche Meetings und Workshops?
- Welche Fragen stelle ich? Worauf muss ich inhaltlich achten?
- Womit fange ich an? Was mache ich in den ersten Workshops? Was dann?

Diese Fragen beschäftigen mich und viele meiner Kunden und für diese Fragen möchte ich Antworten bieten. Ich möchte Ihnen konkrete Methoden und Vorgehensweisen an die Hand geben, wie Sie Workshops zur schrittweisen Ermittlung von Anforderungen effektiv gestalten können. Dabei geht es über eine theoretische Betrachtung allgemeiner Methoden hinaus und tief hinein in die Mühen der täglichen Arbeit als Product Owner, Projektleiter, Business Analyst oder Requirements Engineer.

Die hier vorgestellten Methoden habe ich in meinen Projekten erfolgreich eingesetzt und kann guten Gewissens sagen, dass es auf diese Weise funktioniert. Natürlich ist das nur ein möglicher Pfad durch die Anforderungsermittlung. Er basiert auf meinen Erfahrungen in meinen Projekten. Ich bin sicher, dass jedes

Ihrer Projekte anders ist. So sind Projekte nun einmal. Manche der hier dargestellten Methoden und Tipps werden Sie daher so nicht anwenden können. Vielleicht greifen Sie nur einen Teil davon heraus, passen ihn an Ihr Projekt an und sind dadurch erfolgreicher. Perfekt! Genau das ist unsere Aufgabe als Requirements Engineer, nämlich alles zu tun, was nötig ist, um es den Entwicklern zu ermöglichen, tolle Software zu bauen, die den Anwendern wirklich hilft. Dabei ist es notwendig, flexibel zu reagieren, vom Plan und der vorgefertigten Methode abzuweichen und Dinge zusätzlich zu tun oder wegzulassen. Haben Sie den Mut dazu. Sehen Sie die hier vorgestellten Methoden und Vorgehensweisen als Werkzeugkasten, aus dem Sie sich nach Bedarf bedienen. Mein Ziel ist erreicht, wenn Ihnen das Buch in Ihrer täglichen Arbeit hilft, bessere Software effizient und effektiv zu spezifizieren und zu bauen.

Weil ein bloßes Sammelsurium von Methoden nur bedingt verständlich ist, sind die Kapitel in diesem Buch entlang einer durchgängigen Vorgehensweise angeordnet. In dieser werden die Anforderungen an ein Softwareprodukt ausgehend von einer groben Vision über konkrete fachliche Ziele bis hin zu den dafür benötigten Systemfunktionen systematisch erarbeitet.

Dieses Vorgehen und die ausgewählten Methoden haben sich vor allem in Projekten zur Erstellung von individuellen Softwarelösungen für interne oder externe Kunden sehr gut bewährt. Bei der Erstellung von Standardsoftware müssen Sie die eine oder andere Methode oder Checkliste anpassen. Gerade die Fragen »Wer sind meine Kunden? Wen kann ich bezüglich der Anforderungen fragen?« sind hier etwas anders anzugehen als bei anderen Projekten. Die grundsätzliche Vorgehensweise und die Idee, Workshops zur Anforderungsermittlung einzusetzen, helfen aber auch in solchen Produktentwicklungsvorhaben.

Ein Wort des Dankes

Im Frühjahr 2013 habe ich an der Johannes Kepler Universität in Linz ein Seminar zum Thema »Basis-Moderations- und Präsentationstechniken« bei Andrea Zellinger besucht. Dieses Seminar hat die Art, wie ich an Meetings und Workshops herangehe, grundlegend verändert. Andrea hat mir beigebracht, wie man Workshops zielorientiert aufbaut und die Teilnehmer strukturiert und professionell zum geplanten Ergebnis führt. Ohne dieses Seminar und die extrem motivierende Leitung von Andrea wäre dieses Buch wohl nie entstanden. Vielen Dank dafür!

Danke auch an meine Kolleginnen und Kollegen bei Software Quality Lab, die mich immer wieder von meinem gewohnten Trott abbringen und mir neue Wege zeigen, wie man Softwareentwicklung noch ein wenig besser machen kann. Ihr seid die Besten!

Einen besonderen Dank möchte ich an meinen guten Freund Roman Schacherl richten, der mich an vielen Abenden am Billardtisch zwar spieltechnisch völlig demoralisiert hat, mir dafür aber in spannenden Diskussionen rund um Softwareprojekte geholfen hat, auf den Punkt zu bringen, was unsere Aufgabe als Requirements Engineer ist. Danke Roman! … und das nächste Mal gewinne ich. Danke auch an Karin Huber und Rainer Stropek für das Review und euer tolles Feedback!

Und danke natürlich an das Team vom dpunkt.verlag, allen voran Christa Preisendanz. Das Feedback und die strenge, aber immer objektive und konstruktive Kritik aller Reviewer haben das Buch deutlich besser gemacht.

Inhalt

1 Einleitung

1.1 Meine Vision für die Anforderungsermittlung

> *Anforderungsermittlung bedeutet, herauszufinden, was unsere Kunden wirklich brauchen. Dafür müssen wir unsere Kunden verstehen und begreifen, was sie tun. Wir müssen ihre Bedürfnisse kennen und herausfinden, wie ihnen Software helfen kann, was diese können muss und wie die wesentlichen Funktionen laufen sollen.*

Ein tiefes Verständnis der Probleme, Ziele und Bedürfnisse unserer Kunden ist das Fundament für alle Anforderungen. Basierend darauf können wir gemeinsam mit Anwendern und Entwicklern herausfinden, was die Anforderungen an die neu zu erstellende oder zu ändernde Software sind. Dieses Verständnis für den Kunden und seine Anforderungen müssen wir dann zu denjenigen transportieren, die die Software bauen. Diese müssen es so umsetzen, dass das System am Ende unseren Kunden hilft, ihre Ziele zu erreichen und Probleme zu lösen. Das Verständnis für den Kunden, seine Ziele und Probleme ermöglicht mir als Requirements Engineer, die richtigen Fragen zu stellen, und dem Entwickler, die richtigen Entscheidungen bei der Umsetzung zu treffen.

Meiner Meinung nach ist es nicht immer zwingend notwendig, alles bis ins kleinste Detail vorab auszuspezifizieren. Wenn die Entwickler Erfahrung in der Fachdomäne und mit der eingesetzten Technologie haben, kann ich Detailentscheidungen getrost ihnen überlassen. Haben die Entwickler diese Erfahrung nicht oder ist ein neuer Umsetzungspartner mit im Boot, muss wesentlich genauer erhoben und spezifiziert werden. Es ist meine Aufgabe als Requirements Engineer, zu erkennen, welchen Grad an Spezifikation ein Projekt erfordert, und mein Vorgehen danach anzupassen. Ziel ist nicht die perfekteste Spezifikation oder die Verwendung einer bestimmten Methode, sondern Software, die unsere Kunden weiterbringt!

1.2 Von anderen lernen

Bevor wir uns ins Ermitteln von Anforderungen für Softwaresysteme stürzen, möchte ich Ihnen eine kleine Geschichte zum Thema Umgang mit Kunden und deren Anforderungen in einem völlig anderen Umfeld erzählen:

Erinnern Sie sich noch an Ihren letzten Autokauf? Bei mir war das vor gut 15 Jahren. Ich betrat den Schauraum des Autohändlers meines Vertrauens. Sofort sprang Herr G., der Topverkäufer vor Ort, auf, kam auf mich zu und begrüßte mich herzlich. Er fragte mich freundlich, wie er mir helfen könne. »Ich brauche ein neues Auto, mein altes ist vorige Woche zusammengefallen«, brachte ich es gleich auf den Punkt. »Oje! Na dann schauen wir mal. Wo wohnen Sie denn, Herr Unterauer?«, fragte mich Herr G. Es folgte ein nettes Gespräch, in dem er sich nach meiner Wohnsituation erkundigte, meinem Job, ob ich Kinder hätte, wohin ich in Urlaub fahre und was ich gerne in meiner Freizeit mache. Kaum mal ein Wort über ein Auto. Dafür hatte Herr G. am Ende ein sehr gutes Bild davon, wie mein Alltag so aussieht und wofür ich denn ein Auto überhaupt brauche.

Erst jetzt fragte er mich, was ich mir vorgestellt hätte. Ha! Ich hatte mir schon eine genaue Liste gemacht: Nicht allzu groß sollte es sein, nicht zu teuer, mit Klimaanlage, CD-Player, geringem Verbrauch und langer Lebensdauer. Ein Sportlenkrad und Nebelscheinwerfer wollte ich auch unbedingt haben. Den Allradantrieb, der eigentlich auf meiner Liste gestanden hatte, hatte ich gedanklich während des Gesprächs schon gestrichen. Herr G. hörte sich das alles an, machte sich Notizen und zog dann einige Prospekte hervor, die er mir in die Hand drückte. »Ich möchte Ihnen mal etwas zeigen, Herr Unterauer«, sagte er und führte mich zu einem schicken Mittelklassewagen. Er beschrieb mir die Ausstattung, die Vorzüge, die ewige Lebensdauer und garantierte Pannenfreiheit des Wagens. Das gefiel mir alles sehr gut! Als ich dann allerdings auf das Preisschild schielte, begann meine linke Gesichtshälfte unkontrolliert zu zucken. Der Wagen war zwar toll und hatte alles, was ich zuvor aufgezählt hatte, war aber aufgrund der Größe und Ausstattung viel zu teuer. Herr G. reagierte sofort und bugsierte mich sachte zu einem sportlich angehauchten Kleinwagen. Was soll ich sagen: Der war es! Herr G. plauderte beim Auto noch ein wenig mit mir, hatte aber natürlich längst gemerkt, dass wir den perfekten Wagen für mich gefunden hatten.

Obwohl ich mich kaum losreißen konnte, setzten wir uns als Letztes vor den Computer von Herrn G. Jetzt ging die technische Detailarbeit los. Welches Getriebe, welche Motorisierung, welches Radio, welche Sitzbezüge, Felgen, Lackfarbe und ungefähr noch 20.000 weitere Fragen hatte Herr G. auf seiner Checkliste. Da musste ich jetzt durch. Eine halbe Stunde später verließ ich verschwitzt, aber glücklich das Autohaus. In meiner Tasche hatte ich den Kaufvertrag für mein nagelneues Auto.

Wir können für die Ermittlung von Anforderungen in Softwareprojekten eine Menge von diesem Beispiel aus dem »richtigen Leben« lernen:

1. **Mit Zielen und Nutzen beginnen**
   Starten Sie wie Herr G. mit der Frage nach den Zielen und dem »Warum brauchen Sie das?«. Lernen Sie die Welt der Stakeholder kennen. Nur wenn wir das Ziel kennen, können wir die optimale Lösung finden. Unsere Stakeholder kommen mit einem bunten Mix aus Zielen, Nutzen, Problemen und konkreten technischen Lösungsideen zu uns. Es ist unsere Aufgabe, die Stakeholder etwas zu bremsen und herauszuarbeiten, was der Zweck des Systems ist, welche Ziele damit erreicht und welche Probleme gelöst werden sollen.
2. **Anforderungen schrittweise verfeinern**
   Wenn Sie wissen, was die Ziele sind, erarbeiten Sie die Anforderungen vom Groben zum Feinen. Die zentrale Frage ist hier: »Was soll das System können?« Beginnen Sie mit groben Anforderungsblöcken, präzisieren Sie sie für die Planung und Budgetierung so weit, dass Sie sie schätzen können, und spezifizieren Sie sie für die Umsetzung dann detailliert aus.
3. **Ständig dazulernen**
   Als Requirements Engineer lernen wir zu Beginn die Sprache der Stakeholder und was diese wollen. Unsere Stakeholder lernen, was sie wirklich brauchen und was mit der gegebenen Technologie alles möglich ist. Alle gemeinsam lernen wir, wie wir am besten zusammenarbeiten können, damit wir die Ziele erreichen. Da dieses Lernen im ganzen Projektverlauf weitergeht, ist es völlig natürlich, dass sich die Anforderungen ändern. Unsere Stakeholder dachten vielleicht zu Beginn des Projektes, sie würden für eine bestimmte Aufgabe einen gedruckten Bericht benötigen, schlussendlich stellt sich aber heraus, dass eine reine Bildschirmanzeige die bessere Lösung ist. Rechnen Sie also fix mit Änderungen!

PS: Ich habe meinen Wagen mittlerweile schon 15 Jahre und er bringt mich immer noch ohne Pannen oder größere Reparaturen überall hin. Danke Herr G.! Wenn uns das in unseren Softwareprojekten ebenso gelingt, sind wir schon einen großen Schritt weiter.

1.3 Anforderungen schrittweise ermitteln

In meiner Praxis hat sich folgendes schrittweise Vorgehen zum Ermitteln von Anforderungen bewährt:

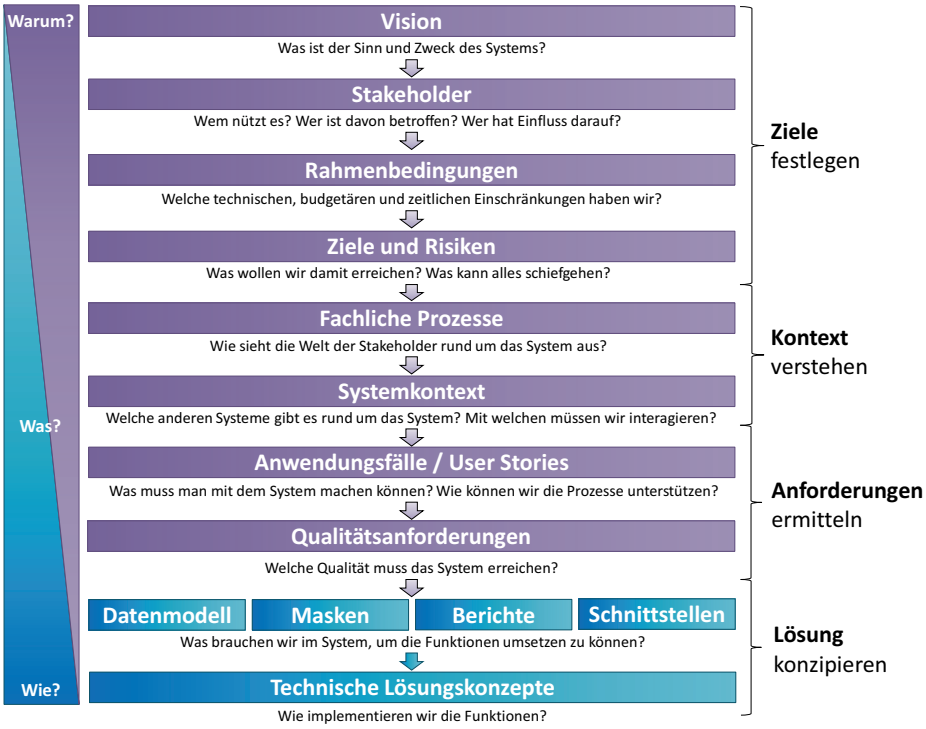


Abb. 1–1 *Anforderungen schrittweise ermitteln und verfeinern*

Beginnen Sie damit, herauszufinden, was der Auftraggeber mit dem System erreichen will. Lernen Sie seine Welt kennen und verstehen. Je besser Ihnen dieser erste Schritt gelingt, desto bessere Fragen werden Sie stellen können und desto besser wird das neue System den Anwendern helfen, ihre Probleme zu lösen und Ziele zu erreichen. Erst danach beschreiben Sie, was das System können und wie es aussehen soll. Das Ergebnis ist ein System, das genau diejenigen Funktionen und Eigenschaften bietet, die die Anwender tatsächlich brauchen. Nicht mehr und nicht weniger. Es wird weder etwas gebaut, das dann nicht verwendet wird. Das wäre Verschwendung. Noch wird etwas weggelassen, was benötigt würde. Das würde zu mangelhaftem Nutzen führen. Das System erfüllt exakt die tatsächlichen Bedürfnisse der Anwender bei geringstmöglichen Entwicklungskosten.

Die linke Seite in Abbildung 1–1 zeigt die Fragen, die uns im Laufe der Zeit leiten. Beginnend mit dem »Warum?« über das »Was?« kommen wir erst am Ende zum technischen »Wie?« Diese Trennung verläuft nicht immer klar und eindeutig. Wenn Sie sich in einer frühen Phase in einer technischen Detaildiskussion

verzetteln, ist das kein Drama. Notieren Sie die wichtigsten Punkte für später und kehren Sie dann wieder zur ursprünglichen Fragestellung zurück.

Der Mittelteil von Abbildung 1–1 beschreibt die einzelnen Themen, die Sie Schritt für Schritt bearbeiten. Beginnen Sie mit dem Systemzweck, den wesentlichen Geschäftszielen und den großen Funktionsbereichen. Diese werden oft gemeinsam mit den wichtigsten Rahmenbedingungen vom Projektsponsor vorgegeben. Aus dem Zweck und den Funktionsbereichen können Sie ableiten, wer die Stakeholder sind. Mit den Stakeholdern können Sie erarbeiten, welche konkreten Ziele erreicht und welche Probleme an der aktuellen Situation gelöst werden sollen. Beschreiben Sie auf grober Ebene die aktuelle Systemlandschaft und die fachlichen Prozesse, in die das neue System eingebettet sein wird.

Gemeinsam mit den Stakeholdern entwerfen Sie, was das System können muss und wie die fachlichen Prozesse der Anwender unterstützt werden können. Diese als Anwendungsfälle spezifizierten Funktionen bilden das zentrale Element der funktionalen Spezifikation. Zu den Anwendungsfällen ergänzen Sie die notwendigen Daten, Masken, Berichte und Schnittstellen (siehe Abb. 1–2).

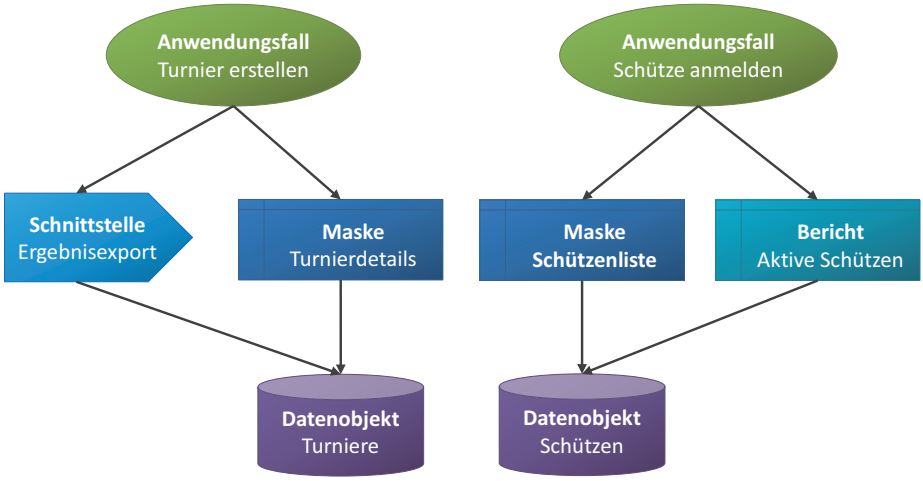


Abb. 1–2 *Systemfunktionen als Anwendungsfälle spezifizieren und dafür Daten, Masken, Berichte und Schnittstellen beschreiben*

All diese Informationen zum »Warum?« und »Was?« bilden gemeinsam eine solide Grundlage für den Entwurf des technischen Lösungskonzeptes.

1.4 Anforderungsermittlung als iterativer Prozess

Diese Vorgehensweise scheint auf den ersten Blick dem klassischen »Big Design Upfront«-Ansatz zu folgen, bei dem erst das gesamte System vollständig und bis ins Detail spezifiziert werden muss, bevor mit der Umsetzung begonnen werden darf. So können Sie es zwar machen, ich empfehle es aber nicht.

Besser ist es, sich an agilen Prinzipien zu orientieren[1]. Das bedeutet, Sie beginnen mit dem Systemzweck, den Zielen und den Stakeholdern. Die darauffolgenden Schritte werden stark überlappend durchgeführt, indem Sie für alle Ebenen immer erst eine Liste mit Überschriften sammeln und deren Inhalte und Details im Laufe der Zeit ausarbeiten, sobald die Umsetzung bevorsteht.

Beispielsweise erstellen Sie mit den Stakeholdern zuerst eine Liste der zu unterstützenden Geschäftsprozesse. Diese priorisieren Sie nach Nutzen. Die wichtigsten 1–5 Prozesse, die schon im ersten Release unterstützt werden müssen, sehen Sie sich nun genauer an. Nur für diese Top-5-Prozesse leiten Sie Anwendungsfälle und genaue Anforderungen ab. Diese werden dann im ersten Release umgesetzt. Während der Umsetzung beginnen Sie mit der Spezifikation des nächsten Release. Das heißt, Sie wählen aus der Liste der Geschäftsprozesse die nächsten aus, spezifizieren sie genauer, leiten Anwendungsfälle ab usw.

Das Grundprinzip dieser Vorgehensweise ist: Wir spezifizieren so spät wie sinnvoll möglich. James Shore spricht hier vom »latest responsible moment« [Shore, 2007]. Es wird lediglich das genau ausspezifiziert, was in der nächsten Zeit umgesetzt wird. Alles andere wird nur grob abgesteckt [Bergsmann, 2018] (siehe Abb. 1–3).

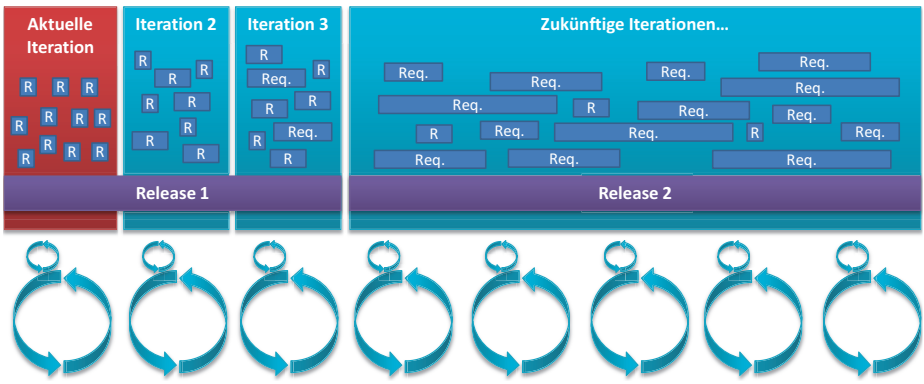


Abb. 1–3 *Je näher die Umsetzung, desto genauer wird spezifiziert (Grafik angelehnt an [Ambler & Lines, 2012] und [Bergsmann, 2018]).*

1. Eine gute Einführung in agile Methoden finden Sie z.B. in [Beck et al., 2001] und [Schwaber, 2004].

Auf diese Weise vermeidet man, dass Dinge spezifiziert werden, die am Ende doch nicht gebraucht oder völlig anders umgesetzt werden. Einer meiner Kunden erzählte beispielsweise, er hatte in einem seiner Projekte Hunderte Stunden in ein 210-seitiges Spezifikationsdokument investiert. Umgesetzt wurde lediglich ein Drittel, den Rest hatten die Entwickler nicht einmal mehr gelesen. Solche »write only«-Spezifikationen wollen wir vermeiden.

Darüber hinaus verteilt sich der Aufwand für die Anforderungsermittlung besser auf das gesamte Projekt (siehe Abb. 1–4, [Bergsmann, 2018]).

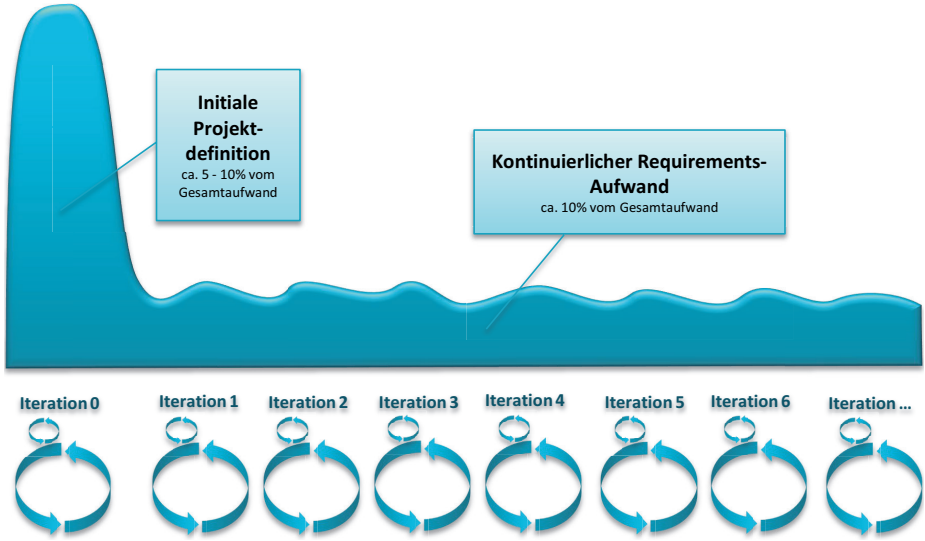


Abb. 1–4 *Aufwand für Anforderungsermittlung verteilt sich über das ganze Projekt.*

Ein weiterer wichtiger Aspekt, den wir bei der Ermittlung von Anforderungen berücksichtigen müssen, ist, dass wir und unsere Stakeholder zu Beginn gar nicht alle Anforderungen kennen. Wir können sie also auch mit noch so viel Aufwand nicht ermitteln. Unser Vorgehen muss uns also ermöglichen, ständig dazuzulernen und das bisher Erreichte zu hinterfragen und zu korrigieren. Wir müssen es dafür unseren Stakeholder so einfach wie möglich machen, Feedback zu geben und möglichst eng mitzuarbeiten [Grau & Lauenroth, o.J.]. Je besser eine Methode das Feedback und die Zusammenarbeit mit unseren Stakeholdern unterstützt, desto besser hilft sie uns in unseren Projekten.

1.5 Workshops und andere Techniken

In diesem Buch geht es um den Einsatz von Workshops zur Ermittlung von Anforderungen. Neben Workshops gibt es natürlich noch zahlreiche andere Methoden und Techniken zum Ermitteln von Anforderungen, die Sie ergänzend oder alternativ zu Workshops einsetzen können. Die folgenden stellen eine kleine Auswahl dar [IREB C.-A. , 2012; Ebert, 2019]:

- **Diskussion und Besprechung**
  Die einfachste Form, ein Thema zu behandeln, ist eine simple Besprechung. Alle Teilnehmer sitzen an einem Tisch und jeder soll sich zum Thema äußern. Ein Moderator achtet darauf, dass sich alle einbringen, dass die Diskussion beim Thema bleibt, und protokolliert die Ergebnisse.
- **Interview und Fragebogen**
  Ein Interview ist eine mehr oder weniger strukturierte mündliche Befragung. Interviews können eingesetzt werden, wenn ein Thema mit einzelnen Personen im Detail beleuchtet oder die Meinungen verschiedener Personen unabhängig voneinander eingeholt und verglichen werden sollen. Ein Beispiel für den Einsatz von Interviews als Ergänzung zu Workshops ist die Detailausarbeitung von Anwendungsfällen, wie in Abschnitt 11.4 beschrieben.

  Ein Fragebogen ist eine schriftliche Befragung. Der große Vorteil dabei ist, dass er statistisch auswertbare und vergleichbare Ergebnisse von einer großen Menge an Personen möglich macht. Setzen Sie ihn immer dann ein, wenn Sie rasch viele Personen, z. B. potenzielle Anwender oder Kunden, zu einem klar definierten Thema befragen wollen.
- **Feldbeobachtung und »In die Lehre gehen«**[2]
  Bei der Feldbeobachtung sehen Sie sich als Requirements Engineer einen Ablauf in freier Wildbahn an. Daraus leiten Sie dann Anforderungen ab. Ein Beispiel für den Einsatz einer Feldbeobachtung als Ergänzung zu Workshops finden Sie in Abschnitt 9.3.

  Das »In die Lehre gehen« geht eine Stufe weiter als die Feldbeobachtung. Hier sehen Sie nicht nur zu, sondern führen den Prozess selbst aus. Natürlich lernt man so am meisten über die Anforderungen an das neue System.
- **FMEA (Failure Mode and Effects Analysis)**
  Bei einer FMEA bewerten Sie für alle Komponenten eines Systems, welche Fehler auftreten können, welche Auswirkungen diese Fehler haben und wie oft sie auftreten werden. Dafür leitet man dann Gegenmaßnahmen ab. FMEA sind im sicherheitskritischen Bereich üblich und durch Normen gefordert [Spillner, A., Roßner, T., Winter, M. & Linz, T., 2014].

2. Geläufiger ist für diese Methode die englische Bezeichnung »Apprenticing«.