Einführung 1

Der Weltraum – unendliche Weiten. Wir schreiben das Jahr 2200. Dies sind die Abenteuer des Raumschiffs Enterprise. Persönliches Logbuch des Captains: Der Computer versteht einfach nicht, was ihm gesagt wird. Scotty versucht seit Tagen, die Bedienung des neuen Transporters in den Griff zu bekommen, und unser Tricorder liefert immer dieselbe unverständliche Fehlermeldung …

1.1 Wir alle sind Benutzer

Ist Ihnen auch schon aufgefallen, dass im Fernsehen die Leute immer mühelos mit der Technik klarkommen? Wir hingegen stolpern bei Anwendungsprogrammen, tippen falsche PIN-Codes ein, verlaufen uns in Flughäfen und verzweifeln an neuen Geräten. Im täglichen Kontakt mit technischen Systemen haben wir uns alle schon eine Vorstellung davon gemacht, was Usability bedeutet. Lassen Sie uns diese Einführung deshalb mit einigen Usability-Klassikern aus dem Alltag der Gegenwart beginnen. Sicher sind auch Ihnen schon solche oder ähnliche Situationen mit gut oder schlecht benutzbaren technischen Systemen in Erinnerung geblieben:

- Der Fahrkartenautomat, der immer gut funktionierte, bis zu dem Zeitpunkt, als Sie die Fahrkarte (mit Quittung) für den nächsten Tag lösen wollten.

M. Richter, M.D. Flückiger, *Usability Engineering kompakt*, IT kompakt,
DOI 10.1007/978-3-642-34832-7_1, © Springer-Verlag Berlin Heidelberg 2013

- Der neue Harddisk-Videorekorder, der auf Knopfdruck das Fußballspiel aufzeichnet, wenn der Pizza-Kurier klingelt. Oder war es eine Tastenkombination? Und wo war noch mal die Anleitung?
- Die Leichtigkeit, mit der Sie Musik aus dem Internet herunterladen, in Musiklisten ordnen und auf Ihrem Smartphone überall hören können.
- Die Telefonrechnung, nachdem Sie mit dem neuen automatischen Buchungssystem endlich Ihre Kinotickets für die Abendvorstellung reserviert hatten.

Interaktive Produkte begleiten uns in unserem Alltag. Wahrscheinlich gehören auch Sie zu jenen Menschen, die sich längst damit abgefunden haben, dass viele Systeme schlichtweg kaum zu benutzen, andere dagegen hervorragend sind. Ist das Zufall? Welche Faktoren bestimmen, ob wir mit einem Produkt sehr einfach, nur schwer oder gar nicht zum Ziel kommen? Welche Möglichkeiten bieten sich, diese Faktoren bereits in der Entwicklung systematisch in den Griff zu bekommen? Mit diesen Fragestellungen beschäftigt sich Usability Engineering.

1.2 Der Benutzer ist nicht wie ich

Bestimmt haben Sie schon einmal einen wichtigen Text geschrieben und den Entwurf jemand anderem zum Lesen gegeben. Sicher haben Sie die Erfahrung gemacht, wie wertvoll die Hinweise dieser anderen Person waren. Sie selbst hatten sich über längere Zeit intensiv mit dem Thema befasst und waren deshalb nicht mehr in der Lage, sich in die Sicht eines außenstehenden Lesers zu versetzen. Sie hätten den Text auch einfach alleine schreiben können, er wäre allerdings nicht so gut geworden wie nach Einarbeitung des Feedbacks.

Die Entwicklung von Software oder interaktiven Produkten ist (in aller Regel) komplexer als das Verfassen eines Textes. Die Projektbeteiligten sind vom Blickwinkel der späteren Anwender in zweierlei Hinsicht weit entfernt:

- Sie sind Spezialisten, die sich über längere Zeit mit der eingesetzten Technologie befasst haben und die Sichtweise eines unbedarften Benutzers nicht mehr ohne Weiteres einnehmen können.

- Sie sind bezüglich des Anwendungsgebietes, in dem die entwickelte Lösung zum Einsatz kommt, oft Laien. Hier ist der Benutzer der Experte. Die Entwickler werden sich nicht umfänglich mit dem Fachgebiet, den Konzepten und Begriffen und schon gar nicht mit den konkreten Abläufen in der alltäglichen Anwendung vertraut machen können.

In beiden Punkten ist die Perspektive der Benutzer notwendig, damit eine brauchbare Lösung entstehen kann. Usability Engineering befasst sich im Wesentlichen damit, wie die Benutzersicht systematisch in die Entwicklung einbezogen werden kann.

Hintergrund: Perspektivenübernahme
Als *Perspektivenübernahme* wird in der Psychologie die Fähigkeit bezeichnet, eine bestimmte Gegebenheit aus der Sicht eines anderen zu verstehen. Die Fähigkeit der Perspektivenübernahme entwickelt sich im Kindesalter und wird im Verlauf des Lebens individuell unterschiedlich stark ausgeprägt. Dabei spielt es nicht nur eine Rolle, ob man sich in die Lage eines anderen versetzen kann. Entscheidend ist auch, den Bedarf für eine Perspektivenübernahme zu erkennen, die Lage aus der Sicht des anderen zu analysieren und die daraus resultierenden Erkenntnisse anzuwenden.

1.3 Was ist Usability Engineering?

Alles sollte so einfach wie möglich gemacht werden, aber nicht einfacher.
(Albert Einstein)

Der Begriff Usability

Noch vor zwei Jahrzehnten war *Usability* als Begriff, zumindest im deutschen Sprachraum, weitgehend unbekannt. Eine eher kleine Fachgemeinde von „Software-Ergonomen" befasste sich mit der „Gebrauchstauglichkeit" bei der Verwendung von interaktiven Systemen. Für unvoreingenommene Ohren müssen diese Bezeichnungen klingen, als wären sie dem Labor einer Forschungseinrichtung entwichen. Auch der etwas geläufigere Ausdruck „Benutzerfreundlichkeit" ist nicht ganz befriedigend, da der Grad der Freundlichkeit eines Systems gegenüber dem Benutzer ein eher diffuses Konzept darstellt. Die „Benutzbarkeit" eines

Systems ist eindeutiger und scheint uns im Sinne der Allgemeinverständlichkeit die treffendste Übersetzung. Usability hat sich inzwischen aber auch im deutschen Sprachraum als Begriff in der alltäglichen Konversation, in den Medien oder auf Produktbeschreibungen etabliert. Immer häufiger ist der Begriff *User Experience* anzutreffen, der auf das Gesamterlebnis des Benutzers mit einem Produkt abzielt (siehe Abschn. 8.2). Wir verwenden die genannten Bezeichnungen in diesem Buch weitgehend gleichbedeutend.

Usability – mehr als die Qualität der Benutzeroberfläche

Es gibt viele Definitionen, was Usability ist, und wir wollen an dieser Stelle auch keine weiteren formal korrekten und allgemeingültigen Konzepte postulieren. Für die Einbettung dieses Buches ist eine Begriffsbestimmung indessen notwendig.

Usability wird manchmal im engeren Sinne als Gütekriterium für die Gestaltung einer Benutzeroberfläche verstanden. Qualitätskriterien sind etwa die Anordnung von Bedienelementen, die Anzahl notwendiger Klicks oder die Verständlichkeit der angezeigten Bezeichnungen und Dialoge.

Hinter dem Begriff verbirgt sich jedoch mehr. Die Benutzbarkeit eines Systems muss im Kontext seiner Verwendung beurteilt werden. Software-Anwendungen oder Produkte weisen eine hohe Usability auf, wenn sie von den vorgesehenen Benutzern einfach erlernt und effizient verwendet werden können und diese damit ihre beabsichtigten Ziele und Aufgaben zufriedenstellend ausführen können.

Ein gutes Beispiel, um den Unterschied zwischen engerem und weiterem Verständnis von Usability zu verdeutlichen, ist der große Erfolg von Kurznachrichten (SMS) mit dem Aufkommen von Mobiltelefonen. Niemand wird bestreiten, dass die rein numerischen Tastaturen dieser Geräte eigentlich nicht für die Erfassung von Text vorgesehen waren. Viele Benutzer empfanden die Erstellung von Nachrichten damit sogar als ziemlich umständlich. Die Benutzerschnittstelle im engeren Sinne war für sie nicht optimal. Betrachtet man die gesamte Anwendung, bot sie hingegen genau das, was der Benutzer eigentlich wollte: Kurze Nachrichten konnten auf einfache und effiziente Weise übermittelt werden.

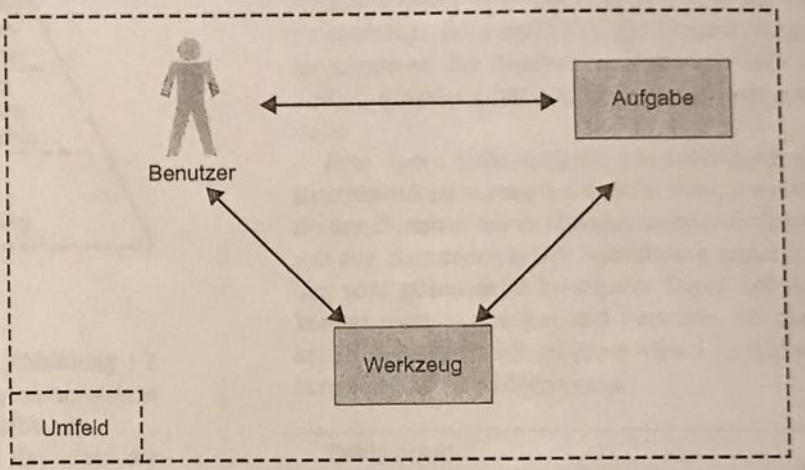

Abb. 1.1 Usability steht dafür, wie gut Benutzer ein Werkzeug in ihrem Umfeld zur Bewältigung ihrer Aufgaben einsetzen können

Das Ziel des Benutzers wurde vom System gut unterstützt. Mit anderen Worten, die Anwendung insgesamt wies eine gute Usability auf. Dieses Beispiel verdeutlicht, dass die Betrachtung der Benutzeroberfläche alleine zu kurz greifen würde. Abbildung 1.1 zeigt die vier prinzipiellen Komponenten eines Mensch-Maschine-Systems.

Eine Definition von Usability in diesem weiteren Sinne wurde in einer ISO-Norm festgelegt. Diese Definition wird oft zitiert, und Sie sollten sie deshalb kennen. Die ISO-Norm 9241-11 definiert *Gebrauchstauglichkeit* als „das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen." [DIN EN ISO 9241-11, 1998].

Aus dieser Definition lässt sich ableiten, dass die verbreitete Ansicht, Usability sei ausschließlich eine Eigenschaft eines Produktes, falsch ist. Um das an einem sehr einfachen Beispiel zu verdeutlichen: Die Usability eines Hammers zum Einschlagen von Nägeln kann gut sein. Doch sie wird ziemlich schlecht ausfallen, wenn Ihre Aufgabe darin besteht, Schrauben einzudrehen. Entsprechend muss das zu erstellen-

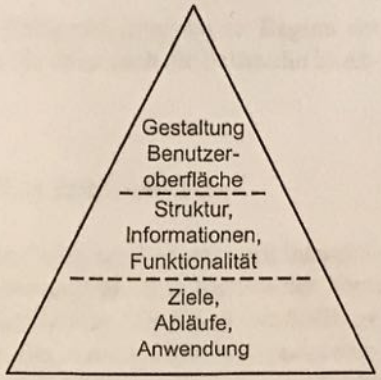

Abb. 1.2 Für die Optimierung der Benutzbarkeit müssen nebst der Benutzerschnittstelle weitere Ebenen wie zum Beispiel Ziele und Funktionalität des Produkts einbezogen werden

de Produkt in die Welt der Benutzer eingepasst werden. Abbildung 1.2 veranschaulicht, dass bei der Betrachtung der Usability verschiedene Ebenen berücksichtigt werden müssen, die aufeinander aufbauen.

Für die Erstellung einer benutzbaren Lösung heißt das, dass die Optimierung der Benutzeroberfläche allein nicht ausreicht. Folgende Aufgaben sind essenziell (mehr dazu finden Sie im Abschn. 2.2):

- Analyse der Benutzer, ihrer Aufgaben und des Anwendungskontexts,
- Festlegen des Funktionsumfangs und der benötigten Informationen,
- Erarbeitung der optimalen Abläufe und Prozesse.

In diesem Zusammenhang werden wir in diesem Buch öfter auf *Prozesse* oder *Abläufe aus Benutzersicht* zu sprechen kommen, um den zeitlichen Aspekt von Interaktionen zu verdeutlichen. Stellen Sie sich die Benutzung eines Systems vereinfacht als Schritte in einer bestimmten, wiederkehrenden Abfolge vor, die zum Ziel führen. Wahrscheinlich gibt es nicht nur eine, sondern mehrere solche Abfolgen. Aus Benutzersicht können die gewünschten Abläufe mehr oder weniger gut mit dem System ausgeführt werden. Usability umfasst die Eignung eines Systems, die Abläufe und Prozesse aus Benutzersicht zu unterstützen.

Usability Engineering: Reduktion auf das Wesentliche

Usability Engineering umfasst Mittel und Techniken, um bei der Entwicklung neuer Software oder Produkte die angestrebte Benutzbarkeit zu erreichen. Dies beinhaltet die Fragestellung, wer die genaue Benutzergruppe ist, die Analyse der Arbeitsabläufe sowie die Festlegung der idealen Funktionalität und Konzeption einer passenden Benutzerschnittstelle.

Eine wesentliche Aufgabe von Usability Engineering ist es, unnötige Komplexität zu vermeiden und die Funktionalität eines Produktes auf ein für den Benutzer ideales Minimum zu reduzieren. Das technische System soll den Anwender in der Ausführung seiner Ziele optimal unterstützen und wird genau dafür konzipiert. Diese Reduktion auf das Wesentliche kommt nicht von selbst und Personen mit den entsprechenden Fähigkeiten müssen im Projektteam sein. Der Aufwand zahlt sich allerdings bereits in der Realisierung aus.

Denkanstoß

Stellen Sie sich einen Toaster vor, der gleichzeitig Spiegeleier braten kann. Welche Zielgruppen erreichen Sie mit diesem Produkt:

a.) Toast-Liebhaber
b.) Spiegeleier-Liebhaber
c.) Beide Zielgruppen: Sowohl Toast- als auch Spiegeleier-Liebhaber
d.) Die Schnittmenge: Jene Leute, die bevorzugt Toast zusammen mit Spiegelei zum Frühstück genießen.

Bonusfrage: Wie wird idealerweise die Dauer der Toast-Zeit auf die Dauer der Spiegelei-Bratzeit abgestimmt?

Der obige Denkanstoß soll einen weiteren Aspekt verdeutlichen: Produkte werden häufig mit vielen Features ausgestattet, um möglichst viele Käufer anzusprechen. Mehr Funktionsvielfalt muss aber in der Regel durch eine höhere Komplexität in der Bedienung erkauft werden. Diese zusätzliche Komplexität darf den Nutzen aus Benutzersicht nicht überschreiten, andernfalls akzeptieren die Benutzer das Produkt nicht oder weichen auf Konkurrenzprodukte aus. Die Zielgruppe der potenziellen Benutzer wird damit nicht größer, sondern kleiner. Konsequentes

Usability Engineering zeigt solche Zielkonflikte schon zu Beginn der Produktentwicklung auf. Vergleichen Sie dazu auch die Fallstudie in Abschn. 6.3.

Anwendungsgebiete für Usability Engineering

Gute Usability spielt überall dort eine Rolle, wo Benutzer mit interaktiven technischen Systemen zu tun haben und damit in irgendeiner Form eine Benutzerschnittstelle zum Einsatz kommt. Dies umfasst Software am Arbeitsplatz ebenso wie Produkte, die in der Freizeit verwendet werden. Der Fokus liegt auf Systemen mit grafischer Benutzeroberfläche (GUI), doch auch Sprachdialoge oder physische Bedienelemente können bezüglich Usability optimiert werden.

Die Entwicklung der Benutzerschnittstelle wird ein zunehmend wichtiger Bestandteil im Software Engineering. Das in diesem Buch beschriebene Vorgehen fokussiert auf die Entwicklung von Software-Anwendungen sowie Produkten mit softwarebasierten Benutzerschnittstellen. Aufgrund der vielen Freiheitsgrade und der Komplexität, die moderne Software-Entwicklungen beinhalten, sehen wir hier die größte Wirkung für benutzerorientierte Vorgehensweisen. Die vorgestellten Methoden und Prinzipien lassen sich jedoch ohne Weiteres auch für die Optimierung von Hardware-Produkten, Dienstleistungsangeboten oder Arbeitsprozessen im weiteren Sinne anwenden. Mehr dazu finden Sie im Abschn. 8.2 zu „User Experience".

Symptome für ungenügende Usability

Denkanstoß

Entwickelt Ihr Unternehmen selbst Software oder Produkte? Wie stellen Sie heute sicher, dass Ihre Produkte gut benutzbar sind? Sind Sie zufrieden mit der erreichten Usability?

Der obige Denkanstoß klingt banal. Doch woher wissen Sie, wie gut die Benutzbarkeit Ihrer Produkte ist? Haben Sie entsprechende Rückmeldungen oder haben Sie diese gemessen? Es ist nicht immer ganz einfach,

die Symptome schlechter Usability auf die Ursache zurückzuführen. Entsprechend verstärkt man unter Umständen den Marketingaufwand, statt Usability-Maßnahmen zu ergreifen. Die folgende Liste zeigt einige Symptome, die von ungenügender Usability herrühren können:

- Die Mitarbeiter arbeiten mit den Systemen nicht so schnell wie erhofft.
- Die Einarbeitung und die Schulung von Benutzern nimmt viel Zeit in Anspruch.
- Die Qualität der geleisteten Arbeit sinkt merklich.
- Die Hotline ist überlastet.
- Mitarbeiter minimieren die Tätigkeit am System. Arbeitsschritte werden auf andere Art und Weise gelöst.
- Prozessvorgaben werden umgangen und Sicherheitsmaßnahmen ignoriert.
- Es gibt immer wieder Fälle, in welchen „Benutzerfehler" die Ursache von Schäden (Unfälle, Datenverluste, kommerzielle Schäden) sind.

1.4 Ein Blick in die Vergangenheit

Um Usability Engineering als Fachdisziplin einzuordnen, lohnt sich ein Blick zurück. Ohne Anspruch auf einen vollständigen geschichtlichen Abriss möchten wir hier einige Meilensteine und Personen aufführen, die maßgeblich zur Entstehung und Verbreitung des Gebietes beigetragen haben:

- Im 15. Jahrhundert stellt Leonardo da Vinci die Kenntnis des Menschen in den Mittelpunkt für die Entwicklung neuer Technologien. Sein Gedankengut sollte Wissenschaft und Technik nachhaltig beeinflussen.
- In den 40er-Jahren des letzten Jahrhunderts investiert vor allem das amerikanische Militär in die Optimierung der Mensch-Maschine-Schnittstelle komplexer Systeme. **Human Factors**, das Fachgebiet zur Erforschung menschlicher Einflussgrößen bei der Anwendung von Technologien, entsteht.
- 1957 erscheint die erste Ausgabe der Fachzeitschrift *Ergonomics*, welche die internationale Verbreitung der **Ergonomie** als Wissen-

schaft zur Erforschung der Beziehungen zwischen dem Menschen und seiner Arbeit auslöst.

- 1970 gründet Brian Shackel in England das Forschungsinstitut HUSAT (*Human Sciences and Advanced Technology*). Die Erforschung der Kommunikation zwischen Mensch und Computer (**Human Computer Interaction** oder kurz *HCI*) wird zur anerkannten wissenschaftlichen Disziplin.
- Mitte der 1980er-Jahre erlebt die systematische Untersuchung der Mensch-Computer-Interaktion durch die Disziplin der **Software-Ergonomie** mit der zunehmenden Verbreitung von Rechnern in der Arbeitswelt einen Aufschwung. Insbesondere im deutschen Sprachraum erscheint eine Fülle von Veröffentlichungen, die das Gebiet prägen.
- 1988 veröffentlicht Donald Norman sein heute als Klassiker geltendes Werk *The Psychology of Everyday Things* [Norman 88]. Das Buch verdeutlicht auf eindrucksvolle Weise die Relevanz psychologischer Faktoren bei der Entwicklung von technischen Systemen.
- 1993 erscheint das Buch *Usability Engineering* von Jakob Nielsen [Nielsen 93]. Es beschreibt die Anwendung von Usability-Methoden in einem systematischen Prozess und gilt als Wegbereiter für benutzerorientierte Vorgehensweisen.
- Im Internet-Boom Ende der 1990er-Jahre setzen viele Unternehmen auf das World Wide Web. Die Nachfrage für die Erstellung benutzerfreundlicher Websites und Anwendungen steigt explosionsartig. *Web Usability* wird zum Schlagwort.
- Im neuen Jahrtausend führt die zunehmende Digitalisierung von Inhalten wie Musik, Foto, Video, die für jedermann erschwinglichen Geräte und Breitband-Verbindungen zu einer weiteren Veränderung. Der Computer wird zum Arbeits-, Kommunikations- und Unterhaltungsmittel und damit zum Alltagsgegenstand. Web 2.0 und **Social Media** Plattformen führen zu mehr und schnellerem Austausch. Neue Technologien mit revolutionären Konzepten für die Mensch-Computer-Interaktion werden alltagstauglich und durchdringen den Markt. Usability wird zum neuen Differenzierungsfaktor für Unternehmen und ihre Produkte.
- 2007 präsentiert Apple das *iPhone*. Das Gerät verfügt über bahnbrechende, intuitive Interaktionsmöglichkeiten und setzt einen neuen

Standard für die Benutzbarkeit mobiler Produkte und Anwendungen. Die damit verbundene Möglichkeit der Installation kleinerer Anwendungen (**Apps**) auf mobilen Geräten wird breiten Bevölkerungskreisen geläufig. Durch die Einfachheit der Benutzung läutet das iPhone eine neue Ära der Informationstechnologie ein. Fortan verändern Smartphones, Tablet PCs und andere mobile Begleiter unseren Alltag. Die **Mobile User Experience** dieser Produkte und Anwendungen wird zu einem zunehmend wichtigen Thema (siehe auch Abschn. 8.6).

Usability Engineering im Entwicklungsprozess

2

Ein Vorgehen für Usability Engineering zu beschreiben, das auf all die verschiedenen Situationen passt, die Sie antreffen werden, ist unserer Ansicht nach nicht möglich. Sie werden auch kein „Kochbuch" für die Anwendung von Usability-Methoden zu lesen bekommen. Die folgenden Kapitel sollen vielmehr eine Hilfestellung geben, wie Usability Engineering in der Praxis der Software- und Produktentwicklung betrieben werden kann, wie die wichtigsten Usability-Methoden grundsätzlich ablaufen, wie sie eingeplant werden können und worauf Sie achten sollten.

Dieses Kapitel gibt eine Übersicht über den Zusammenhang der verschiedenen Usability-Methoden und deren Integration in gängige Software-Entwicklungsprozesse. Im nächsten Kapitel werden die einzelnen Methoden näher beschrieben.

2.1 Software Engineering: Die vergessenen Benutzer?

Software-Entwicklungsprozesse

Gemäß einer viel zitierten Studie der Standish Group sind *unvollständige Anforderungen* und *mangelnde Einbeziehung der Benutzer* die zwei führenden Gründe, warum Software-Projekte scheitern [Standish Group 1994–2010]. Aktuelle Vorgehensmodelle nehmen für sich in Anspruch, hier Abhilfe zu schaffen. Tätigkeiten im Bereich Anforderungsanalyse

M. Richter, M.D. Flückiger, *Usability Engineering kompakt*, IT kompakt,
DOI 10.1007/978-3-642-34832-7_2, © Springer-Verlag Berlin Heidelberg 2013

Tab. 2.1 Die sechs primären Software-Engineering-Disziplinen im Unified Process (Quelle: IBM Rational Unified Process®)

Geschäftsprozessmodellierung (Business Modeling)
Anforderungsanalyse (Requirements)
Analyse und Entwurf (Analysis & Design)
Implementierung (Implementation)
Test (Test)
Auslieferung (Deployment)

und -management haben in den meisten neueren Prozessbeschreibungen einen hohen Stellenwert.

Der *Unified Process* (**RUP**), ein etablierter Software-Entwicklungsprozess, definiert beispielsweise **Requirements** als eine von sechs primären Disziplinen im Software Engineering (siehe Tab. 2.1).

Die Disziplin *Requirements* umfasst jene Aktivitäten, bei denen es um das Erfassen, Dokumentieren und Verwalten von Anforderungen der verschiedenen Interessengruppen geht (siehe unter „Hintergrund: Requirements Engineering"). Als eine der wichtigsten Interessengruppen sollten die Benutzer dabei eine zentrale Rolle spielen!

Agile Vorgehensweisen wie beispielsweise Scrum versprechen noch mehr Kommunikation und engere Feedbackzyklen zwischen allen an der Software-Entwicklung Beteiligten (siehe auch „Hintergrund: Agile Software-Entwicklung").

Für Usability-Engineering-Aktivitäten sind in modernen Software-Entwicklungsprozessen somit einige wesentliche Rahmenbedingungen erfüllt:

- Die Bedürfnisse der Benutzer werden im Rahmen der Anforderungsanalyse erhoben.
- Die Anforderungen aus Benutzersicht werden modelliert und in die Entwicklung getragen.
- Eine iterative Vorgehensweise erlaubt die ständige Überprüfung und Verfeinerung von Ergebnissen mit Kunden und Benutzern.
- Einige Best Practices aus dem Usability Engineering, wie User Interface Storyboards zur Darstellung von Dialogabläufen, Personas zur Charakterisierung der Benutzer und die Erstellung von GUI-Prototypen, werden in vielen Vorgehensbeschreibungen explizit erwähnt.

Es wäre somit zu erwarten, dass in Software-Projekten generell und in der Entwicklung von Benutzerschnittstellen im Speziellen eine systematische Einbeziehung der Benutzer und ihrer Anforderungen stattfinden müsste.

Ist das Problem damit gelöst? Die Praxis spricht eine andere Sprache. In vielen Projekten wird das User Interface vom Entwickler alleine, ohne fremde Hilfe erstellt. Die späteren Benutzer werden selten einbezogen.

Ein Grund dafür mag in Folgendem liegen: Obwohl die Berücksichtigung der Benutzeranforderungen in aktuellen Software-Engineering-Vorgehensmodellen als wichtig erachtet wird, fehlen für das tatsächliche Einbeziehen von „Endbenutzern" entsprechende Methoden und Techniken weitgehend. Damit fehlen auch Erfahrungswerte für die Projektplanung. Das Resultat ist, dass viele Projekte benutzerorientierte Vorgehensweisen als optional oder als zusätzlichen Aufwand betrachten und ohne entsprechende Aktivitäten vorgehen.

Hintergrund: Agile Software-Entwicklung

In agilen Vorgehensweisen wird die Entwicklung einer Software in kurze – im Allgemeinen zwei bis vier-wöchige – Abschnitte aufgebrochen. Diese Zyklen werden Iterationen genannt. Das Ergebnis jeder solchen Iteration ist ein realisiertes, getestetes und mit Kunden geprüftes Teilprodukt. Dieses kann – soweit es die zu diesem Zeitpunkt umgesetzte Funktionalität erlaubt – potenziell verwendet werden.

Eine Iteration soll dabei insbesondere ein Lernprozess sein. Dazu legt das Team zusammen mit Kunden fest, was genau entwickelt werden soll, realisiert und prüft die vereinbarte Software, reflektiert anschließend die Ergebnisse und das Vorgehen und leitet Verbesserungsmaßnahmen ein. So lernt ein Projektteam über das zu entwickelnde Produkt, die eingesetzten Technologien und die Zusammenarbeit.

Agile Teams streben aus diesem Grund danach, die Spezifikation so zeitnah wie möglich mit der eigentlichen Entwicklung vorzunehmen und zwar im direkten Gespräch mit Kunden anstatt über Dokumente. So entsteht die Spezifikation aus dem entwickelten Teilprodukt, dem besseren Problemverständnis und der optimierten Zusammenarbeit. Ein Projektteam beurteilt immer wieder neu, wie mit den verbleibenden Ressourcen das bestmögliche Produkt realisiert werden kann.

Agile Prinzipien wurden im *Manifest für agile Software-Entwicklung* [Beck et al. 01] festgehalten:

- Die Kommunikation zwischen allen Beteiligten – inklusive der späteren Benutzer – wird über die Definition von Prozessen und Werkzeugen gestellt.

- Funktionierende Software wird als wichtiger erachtet als umfassende Dokumentation.
- Die Zusammenarbeit mit dem Kunden spielt eine wichtigere Rolle als Vertragsverhandlungen.
- Die Reaktion auf Veränderungen steht über der Befolgung eines Plans.

Agile Vorgehensweisen verfolgen eine Philosophie, die sich stark an den menschlichen Denkprozessen und der Zusammenarbeit im Team orientiert und versprechen damit eine höhere Produktivität bei der Softwareentwicklung. Mehr zu dieser Thematik finden Sie im Buch „Software entwickeln mit Verstand" [Dirbach et al. 11].

Benutzerorientierte Modelle

Auf der anderen Seite existieren seit gut fünfzehn Jahren benutzerorientierte Vorgehensmodelle, welche die Anwendung von Usability-Methoden im Entwicklungsprozess beschreiben. **User Centered Design** (UCD) beschäftigt sich damit, wie Benutzer systematisch in die Entwicklung von Systemen und Produkten einbezogen werden können. An dieser Stelle sei auf die hervorragenden, durchgängig benutzerorientierten Vorgehensbeschreibungen von namhaften Vertretern des Fachgebiets verwiesen:

- *The Usability Engineering Lifecycle* von Deborah Mayhew [Mayhew 99] beschreibt den Zusammenhang von Usability-Methoden über den gesamten Lebenszyklus bei der Entwicklung einer neuen Lösung. Mayhew illustriert, wie aus Benutzeranforderungen konkrete Usability-Ziele abgeleitet und in User Interface Entwürfe umgesetzt werden, die wiederum mit Benutzern geprüft werden.
- *Contextual Design* von Hugh Beyer und Karen Holtzblatt [Beyer et al. 98] ist ein benutzerzentrierter Design-Prozess. Er vertieft die Analyse der Benutzer und deren Umfeld und zeigt auf, wie die gewonnenen Informationen in die Entwicklung einfließen. Aus diesem Vorgehen stammt die Methode „Contextual Inquiry" (vergleiche Abschn. 3.1).
- *Goal Directed Design* von Alan Cooper [Cooper et al. 10] beschreibt eine Vorgehensweise, um Benutzeranforderungen zu modellieren und in ein passendes Interaktionsdesign umzusetzen. Cooper's Ansatz orientiert sich bei der Gestaltung neuer Lösungen an den überliegenden Zielen der Benutzer. Um die relevanten Eigenschaften und

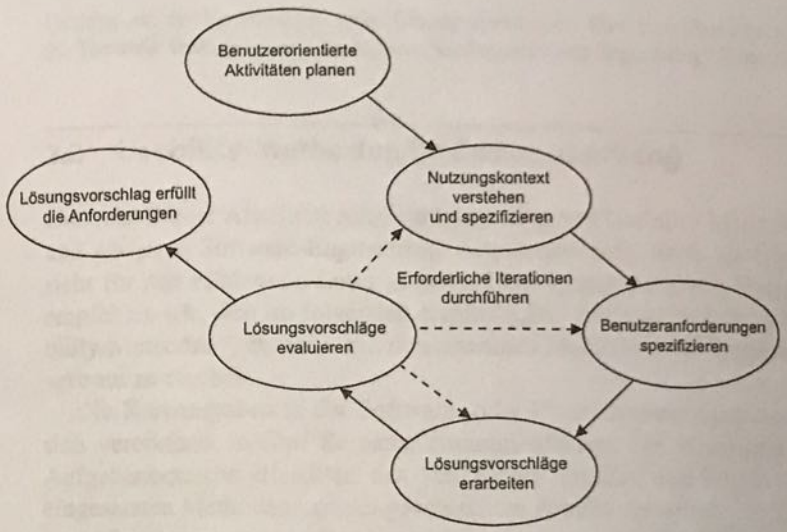

Abb. 2.1 Die ISO-Norm 9241-210 definiert die Schritte, die für eine benutzerorientierte Gestaltung von interaktiven Systemen eingehalten werden müssen. Ein neuer Lösungsvorschlag erfüllt die Anforderungen erst dann, wenn er erfolgreich mit Benutzern evaluiert wurde

Bedürfnisse der Benutzer darzustellen, hat Cooper die Verwendung prototypischer Benutzer *(Personas)* eingeführt (siehe Abschn. 3.2).

Ein Prozessmodell für die *benutzerorientierte Gestaltung interaktiver Systeme* wurde sogar in einer ISO-Norm verankert [DIN EN ISO 9241-210]. In Abb. 2.1 sind die wichtigsten Prozessschritte dargestellt. Eine genaue Beschreibung dieser Schritte ist in ISO 9241-210 enthalten.

Wer nach diesen Ansätzen vorgeht, wird bessere Software und Produkte produzieren – kein Zweifel!

Eines ist indessen nicht von der Hand zu weisen: Eine Integration in gängige Software-Entwicklungsprozesse fehlt weitgehend. Es entsteht unweigerlich das Gefühl, dass diese Ansätze als aufwändige, eigenständige Disziplinen gehandelt werden: „Vergesst eure alten Vorgehensweisen und nehmt unseren Prozess". Diese Abgrenzungshaltung mag dazu beitragen, dass Usability Engineering von Auftraggebern und

Projektbeteiligten oft als zeit- und kostenintensiver Zusatz zum bestehenden Vorgehen betrachtet wird.

Wenn Sie nun aber ein Projekt nach einem bestehenden Vorgehensmodell oder einem gegebenen Entwicklungsprozess abwickeln müssen, weil Ihr Unternehmen dies vorschreibt oder weil Sie darin ganz einfach viele nützliche Hilfsmittel finden, wie können Sie benutzerorientierte Methoden integrieren? Sollen Sie nun Geschäftsprozessmodellierung oder Benutzerbeobachtungen in Auftrag geben, um die notwendigen Arbeitsschritte zu analysieren? Verwenden Sie besser Szenarien, User Stories oder Use Cases für Ihre Spezifikation? Wann testen Sie Ihre neue Lösung am besten mit Benutzern – und mit wie vielen? Die folgenden Kapitel sollen Sie bei solchen und ähnlichen Fragen unterstützen.

Das Beste aus beiden statt zwei Welten

Die Erfahrung aus einer Vielzahl von Projekten hat uns gezeigt, dass sich Usability Engineering nahtlos in gängige Software und Produkt-Entwicklungsprozesse integrieren lässt und sich dadurch viele Vorteile bieten. So hat sich beispielsweise die Anwendung von Usability-Methoden im *Requirements Engineering* in der Praxis als äußerst erfolgreich erwiesen. Erstens werden die Anforderungen der Benutzer systematisch in die Analyse einbezogen und zweitens kann sichergestellt werden, dass diese Anforderungen auch ihren Weg in die Realisierung finden. Das Ergebnis ist eine Lösung, die sich an ihrem tatsächlichen Verwendungszweck ausrichtet.

Eine solche integrierte Sicht bedingt, dass Vorgehensbeschreibungen nicht einfach unreflektiert angewendet werden, sondern dass die zugrunde liegenden Prinzipien beider Welten verstanden und berücksichtigt werden.

Hintergrund: Requirements Engineering

Requirements Engineering befasst sich damit, die Bedürfnisse der Benutzer, Auftraggeber und weiterer Interessengruppen (*Stakeholder*) hinreichend aufzuarbeiten, zu verwalten und zu kommunizieren, damit das Projektteam eine passende Lösung erstellen kann. Wesentlich für den Projekterfolg ist eine von allen Interessengruppen vereinbarte und getragene Zielvorstellung. Aufbauend darauf kann der Analyst die Rahmenbedingungen und Qualitätsanforderungen erarbeiten sowie den funktionalen

Umfang und das Verhalten der neuen Lösung modellieren. Eine gute Übersicht über die Thematik bietet das Buch „Systematisches Requirements Engineering" [Ebert 10].

2.2 Usability-Methoden im Zusammenhang

Hinweis: Dieser Abschnitt zeigt die Integration von Usability-Methoden und gängigen Software-Engineering-Vorgehensweisen. Er ist als Übersicht für den erfahrenen Leser gedacht. Dem Einsteiger in die Materie empfehlen wir, sich im folgenden Kapitel („Die 7 ± 2 wichtigsten Usability-Methoden") zunächst mit den einzelnen Methoden und Konzepten vertraut zu machen.

Die Kernaufgaben in der Software- oder Produktentwicklung lassen sich vereinfacht in fünf Bereiche zusammenfassen. Die Einteilung in Aufgabenbereiche erleichtert das Verständnis von Ziel und Zweck der eingesetzten Methoden und der gewünschten Arbeitsergebnisse. Sie sind unabhängig vom eingesetzten Entwicklungsprozess. Abbildung 2.2 zeigt diese Aufgabenbereiche in der Übersicht.

Jedem Aufgabenbereich kann ein Ziel im Sinne der Benutzerorientierung zugeordnet werden:

- *Analyse:* Benutzer und Kontext verstehen,
- *Modellieren:* Entwurf und Optimierung einer passenden Lösung,
- *Spezifikation:* Die Lösung in die Entwicklung tragen,
- *Realisierung:* (Unterstützung bei der) Implementierung der Lösung,
- *Evaluation:* Resultate mit Benutzern überprüfen.

Diese Aufgabenbereiche sind nicht im Sinne eines zeitlichen Ablaufs zu verstehen. So kann es beispielsweise zielführend sein, zunächst eine Evaluation eines bestehenden Systems vorzunehmen, bevor die Modellierung einer neuen Lösung angegangen wird. Oder es kann ein erstes Modell erstellt werden, das der Analyse dient. In der Praxis werden diese Aufgaben oft auch wiederholend (iterativ) oder sogar parallel durchgeführt. In Kap. 4 werden diese zeitlichen Zusammenhänge näher beschrieben.

Im Folgenden wird für jeden Aufgabenbereich aufgezeigt, welche Ansätze und Methoden einerseits aus den Software-Engineering-Vorgehensmodellen und andererseits aus benutzerorientierten Prozessmodel-