



Christian Aichele  
Marius Schönberger *Hrsg.*

# App4U

Mehrwerte durch Apps im B2B und B2C



Springer Vieweg

---

App4U

---

Christian Aichele • Marius Schönberger  
(Hrsg.)

# App4U

Mehrwerte durch Apps im B2B und B2C



Springer Vieweg

*Herausgeber*  
Christian Aichele

Marius Schönberger  
Fachbereich Betriebswirtschaft  
Fachhochschule Kaiserslautern  
Zweibrücken  
Deutschland

ISBN 978-3-8348-2435-6  
DOI 10.1007/978-3-8348-2436-3

ISBN 978-3-8348-2436-3 (eBook)

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

Springer Vieweg  
© Springer Fachmedien Wiesbaden 2014  
Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags.  
Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier

Springer Vieweg ist eine Marke von Springer DE. Springer DE ist Teil der Fachverlagsgruppe Springer Science+Business Media  
[www.springer-vieweg.de](http://www.springer-vieweg.de)

---

## Vorwort der Herausgeber

**Zielsetzung dieses Buchs** Zielsetzung des vorliegenden Buchs ist es, einen fundierten Überblick über den gegenwärtigen Stand und die zukunftsweisenden Entwicklungen und Technologien aus der überaus hohen Fülle an Themen im Bereich der mobilen Anwendungen und Anwendungsentwicklung zu geben. Das Buch verfolgt diesbezüglich vor allem das Ziel, Grundlagen zu vermitteln, die in eigenen Entwicklungsprojekten für die Sicherstellung einer erfolgreichen Umsetzung angewendet werden können. In diesem Zusammenhang werden notwendige Kenntnisse aus den Bereichen Softwaretechnik, Software Engineering, Projektmanagement sowie Marketing und Vertrieb vermittelt, die für die Entwicklung, Planung und Vermarktung mobiler Anwendungen benötigt werden. Ein weiteres Ziel des Buchs besteht darin, eine Transparenz über die unterschiedlichen Einsatzmöglichkeiten mobiler Applikationen zu schaffen und sowohl Rahmenbedingungen und Problembereiche als auch Handlungsempfehlungen für Unternehmen aufzuzeigen. Hierzu werden dem Leser reale Entwicklungsprojekte vorgestellt, deren Planungs- und Entwicklungsphase aufgezeigt sowie innerhalb des Entwicklungsprozesses identifizierte Probleme und Lösungsansätze dargestellt.

**Was liefert das vorliegende Buch?** Die Kapitel dieses Herausgeberbandes enthalten eine Vielzahl praxisrelevanter Inhalte zu den Themen App-Entwicklung, -Geschäftsmodelle und -Marketing. Das vorliegende Buch liefert weiterhin eine differenzierte Sichtweise über mögliche Einsatzgebiete und Anwendungsbereiche mobiler Anwendungen im B2B- oder B2C-Bereich. Praktiker werden bei der Lektüre dieses Bandes umfangreiche Hilfestellungen und konkrete Informationen zur Umsetzung von Strategien und Vorgehensweisen zur App-Entwicklung erhalten. Aber auch Theoretiker und Wissenschaftler sowie allgemein an der Thematik „App“ Interessierte werden das Buch mit Gewinn lesen können.

**An wen richtet sich „App4U“?** Das vorliegende Buch wendet sich vornehmlich an Unternehmer, IT-Verantwortliche und IT-Praktiker aus IT-anwendenden Unternehmen und IT-Unternehmen, ferner an Lehrende und Studenten der Wirtschaftsinformatik und BWL sowie allgemein all diejenigen Personen in Gesellschaft und Politik, die sich mit der Zukunft des IT-Sektors beschäftigen.

**Aufbau des Buchs** Das Buch strukturiert die Thematik der mobilen Applikationen in drei Hauptteile. Im ersten Teil werden die allgemeinen Grundlagen von Apps, die Strategien zur App-Entwicklung und der App-Markt aus unterschiedlichen Perspektiven heraus betrachtet. Der zweite Teil beschäftigt sich mit Vorgehensmodellen und Technologien zur App-Entwicklung. Gegenstand des dritten Teils sind Praxisbeispiele der App-Entwicklung. Nachfolgend werden die einzelnen Kapitel jeweils entsprechend ihrer inhaltlichen Zuordnung zu den aufgeführten Teilen I. bis III. skizziert.

---

## I. Grundlagen, Strategie und Markt

Christian Aichele und Marius Schönberger führen zunächst in die Thematik der mobilen Applikationen ein. Die Autoren geben eine Übersicht über die Entstehung mobiler Anwendungen und zeigen wesentliche Erfolgsfaktoren in dem B2B- und B2C-Bereich. Abschließend werden der Aufbau und die Schwerpunkte des vorliegenden Buchs aufgezeigt.

Marius Schönberger stellt im zweiten Kapitel „Auf dem Weg zur optimalen mobilen Anwendung: Orientierung und Beweggründe zur Entwicklung mobiler Anwendungen“ zunächst fest, dass sich mit dem Aufbau einer neuen App-Infrastruktur die Prozesse und Arbeitsweise in Unternehmen erheblich verändern. Er legt in seinem Beitrag dar, dass kaum ein Unternehmensbereich von den Veränderungen im Zusammenhang mit der Ausbringung von mobilen Applikationen ausgeschlossen bleibt. Unternehmen entscheiden sich immer mehr, den Einsatz mobiler Anwendungen für kundenspezifischen Mehrwert zu nutzen. Für die Vorbereitung des Entwicklungsvorhabens gibt der Autor einen Orientierungsrahmen vor, der die Besonderheiten der B2B- und B2C-Märkte, das Vorgehen bei der Auswahl und Einführung mobiler Anwendungen sowie anfallende Kosten bei der Entwicklung berücksichtigt.

Christian Aichele beschreibt im Kapitel „Strategien und Geschäftsmodelle für mobile Applikationen: Die Vorgehensweise zur Etablierung mobiler Applikationen im B2B und B2C“ den Paradigmenwechsel in der Informationstechnologie von der klassischen proprietären und monolithischen Informationstechnologie hin zum dynamischen Einsatz mobiler Applikationen auf unterschiedlichsten Endgeräten. Nach einer Darstellung von Strategien für die App-Entwicklung und Vorgehensweisen zur Generierung einer Strategie auf Basis eines Fallbeispiels skizziert der Autor Geschäftsmodelle für mobile Applikationen im B2C- und B2B-Umfeld. Abschließend wird in das Projektmanagement für die App-Entwicklung eingeführt. Der Autor gelangt dabei zu dem Schluss, dass die etablierten Unternehmen, die den App-Einsatz nicht rechtzeitig oder nur unzureichend vollziehen sowie ihr Geschäftsmodell nicht weiterentwickeln, das Risiko eingehen, in Zukunft erhebliche Nachteile in Kauf nehmen zu müssen.

## **II. Vorgehensmodelle und Technologien**

Im vierten Kapitel „Der professionelle Einstieg in die erfolgreiche App-Entwicklung: Fachliche und technologische Grundlagen der App-Entwicklung“ beschreibt Marius Schönberger fachliche und technologische Grundlagen im Zusammenhang mit der operativen Einführung von mobilen Applikationen. Der Autor gibt zunächst eine Einführung in die Thematik der Softwareentwicklung. Daran anknüpfend werden mobile Endgeräte hinsichtlich ihrer Typologisierung, verwendeten Betriebssysteme und spezifischen Eigenschaften betrachtet. In einem weiteren Schritt werden verschiedene Applikationstypen und Entwicklungsstrategien dargestellt. Das Kapitel endet mit einer Vorstellung verschiedener Werkzeuge zur Unterstützung der mobilen Anwendungsentwicklung.

Im Kapitel „Mit Struktur und Methode in die projektindividuelle App-Entwicklung: Praktische Ansätze zur zielorientierten Anwendung von Software Engineering“ stellen Christian Aichele und Marius Schönberger zunächst fest, dass in der aktuellen Diskussion um mobile Anwendungen bisher keine einheitlichen Vorgehensmodelle zur Planung, Implementierung und Vermarktung von Apps bestehen. Aus diesem Grund haben die Autoren ein Vorgehensmodell zur mobilen Anwendungsentwicklung aufgestellt, welches in die Phasen Markt- und Problemanalyse, Planung und Konzeption, Entwicklung und Test sowie Einführung und Veröffentlichung unterteilt ist. Neben den notwendigen Aufgaben- und Tätigkeitsbereichen innerhalb der genannten Phasen werden weiterhin Werkzeuge und Methoden zur Unterstützung des Entwicklungsprozesses vorgestellt.

Klaus Knopper stellt im sechsten Kapitel „Mobile Security: Grundlagen und Beispiele aus Theorie und Praxis zur Systemsicherheit bei der Anwendung mobiler Softwaresysteme“ fest, dass bei der Nutzung von mobilen Applikationen der Datenschutz und die Datensicherheit besondere Aufmerksamkeit erfordern. Neben der Sicherheitsarchitektur mobiler Betriebssysteme und Anwendungen beschreibt der Autor weiterhin verschiedene Vorfälle und Bedrohungsszenarien in Bezug auf die mobile Sicherheit. Anhand der in der Informationssicherheit allgemein bestehenden Teilgebiete Vertraulichkeit, Verfügbarkeit und Integrität gibt der Autor abschließend technische Hilfsmittel, Sicherheitsmaßnahmen und Handlungsempfehlungen.

---

## **III. Praxisbeispiele**

Im Eingangskapitel des dritten, technisch orientierten Buchteils beschäftigt sich Marius Schönberger in seinem Kapitel „Business Case I: Mobile Applikationen zur persönlichen Finanzplanung“ mit der Konzeption und Entwicklung einer mobilen Applikation zur Baufinanzierung. Der Autor beschreibt hierbei den gesamten Ablauf des Software Engineering, beginnend bei der vorliegenden Problemstellung und Motivation zur Entwicklung, über die Planungs- und Konzeptphase bis zur Implementierung und der Evaluierung der mobilen Anwendung.

Christian Radny stellt in seinem Kapitel „Business Case II: Presenter App“ zunächst fest, dass die Kommunikations- und Informationstechnologie eine Schlüsselrolle beim Einsatz von mobilen Applikationen im B2B einnimmt. Dabei sind neue, hochperformante Technologien ein wichtiger Baustein für den Aufbau und die Entwicklung mobiler Softwarearchitekturen. Der Autor beschreibt daher zunächst Besonderheiten bei der Softwareentwicklung für mobile Endgeräte, bevor er nachfolgend auf die Entwicklung und Konzeption einer Presenter-Anwendung für PowerPoint eingeht, die für das mobile Betriebssystem Android umgesetzt wurde. Ziel des Autors ist es, dass der Leser die Softwareentwicklung unter Android verstehen soll und den Aufbau der entwickelten Applikation nachvollziehen kann.

Im Kapitel „Business Case III: Rating-Tool“ plädiert Dennis Christmann dafür, die App-Technologie nicht isoliert zu betrachten, da diese eine revolutionäre Basistechnologie in der serviceorientierten Architektur darstellt, deren Bedeutung erst durch eine konsequente Integration in die Geschäftsanwendungen von Unternehmen sichtbar wird. Unternehmen stehen jedoch vor der Herausforderung, aus der Vielfalt der am Markt vorhandenen mobilen Anwendungen eine für unternehmensinterne oder -externe Problemstellungen benötigte App auszuwählen. Damit Unternehmen sich am Markt für mobile Applikationen zurechtfinden, greifen viele auf Tests und Rezensionen zurück. Hierzu werden oftmals für die Erzeugung von Rankings Bewertungstools verwendet. Durch den Beitrag wird die Entwicklung eines Rating-Tools zur Bewertung von mobilen Applikationen vorgestellt. Im ersten Schritt wird das Bewertungsmodell entwickelt, auf dessen Basis im Anschluss das Rating Tool in Java entwickelt wird.

Bei allen hier dargestellten Praxisbeispielen handelt es sich um erfolgreiche App-Entwicklungen, die sämtlich zum Zeitpunkt der Drucklegung dieses Buchs Ende 2013 bereits abgeschlossen waren.

Das vorliegende Buch gibt den technischen Entwicklungsstand des Sommers 2013 wieder. Einige Aussagen und Analysen der nachfolgenden Kapitel sind durchaus eng mit diesem Stand verknüpft und folgerichtig vor diesem zeitlichen Bezug zu bewerten. Vielfach sind jedoch die in diesem Buch getätigten Aussagen prinzipieller Natur und infolgedessen auch ohne direkten Zeitbezug. Aussagen zur optimalen Gestaltungsmethodik von App-Projekten, zum Management von App-Entwicklungen, zu Methoden der Entwicklung und Qualitätssicherung und vielem mehr behalten auch nach einer Gesetzesänderung oder geänderten Technologie weiterhin ihre Gültigkeit.

Zum Schluss gilt unser ganz besonderer Dank allen an diesem Buch beteiligten Autoren, ohne deren hohes Engagement beim Verfassen der nachfolgenden Kapitel dieses Buchprojekt nicht hätte realisiert werden können. Darüber hinaus bedanken wir uns bei zahlreichen Führungskräften aus der IT-Industrie, Fachexperten und Praktikern, die uns bei der Erstellung dieses Buchs wiederholt mit Rat und ihrem detaillierten Wissen unterstützt haben. Nicht zuletzt gilt unser Dank auch der professionellen Unterstützung und wohlwollenden Begleitung durch das Lektorat Informatik und Elektrotechnik des Springer Vieweg Verlags.

Wir würden uns freuen, wenn der vorliegende Herausgeberband einen Beitrag zur inhaltlichen Konkretisierung und zum Erfolg von App-Entwicklungen leisten könnte sowie dem Praktiker bei der Umsetzung von Projekten zur App-Erstellung hilfreiche Informationen zur erfolgreichen Realisierung geben kann.

Ketsch, im September 2013  
Homburg, im September 2013

Christian Aichele  
Marius Schönberger

---

# Inhaltsverzeichnis

<b>1</b>	<b>App4U – Die Welt der mobilen Applikationen .....</b>	1
	Christian Aichele und Marius Schönberger	
<b>2</b>	<b>Auf dem Weg zur optimalen mobilen Anwendung .....</b>	13
	Marius Schönberger	
<b>3</b>	<b>Strategien und Geschäftsmodelle für mobile Applikationen .....</b>	35
	Christian Aichele	
<b>4</b>	<b>Der professionelle Einstieg in die erfolgreiche App-Entwicklung .....</b>	87
	Marius Schönberger	
<b>5</b>	<b>Mit Struktur und Methode in die projektindividuelle App-Entwicklung .....</b>	133
	Marius Schönberger und Christian Aichele	
<b>6</b>	<b>Mobile Security .....</b>	217
	Klaus Knopper	
<b>7</b>	<b>Business Case I: Mobile Applikationen zur persönlichen Finanzplanung .....</b>	251
	Marius Schönberger	
<b>8</b>	<b>Business Case II: Presenter App .....</b>	309
	Christian Radny	
<b>9</b>	<b>Business Case III: Rating Tool .....</b>	353
	Dennis Christmann	
	<b>Sachverzeichnis .....</b>	403

---

## Abkürzungsverzeichnis

ADT	Android Developer Tools
AMI	Advanced Metering Infrastructure
AMM	Advanced Metering Management
API	Application Programming Interface
APK	Android Application Package File
App	Mobile Applikation
ARIS	Architektur integrierter Informationssysteme
ARPAnet	Advanced Research Projects Agency Network
B2B	Business-to-Business
B2C	Business-to-Consumer
B2E	Business-to-Employee
BGB	Bürgerliches Gesetzbuch
BMVg	Bundesministerium für Verteidigung
BOM	Business Object Management
BPMN	Business Process Model and Notation
BPR	Business Process Reengineering
BWB	Bundesamt für Wehrtechnik und Beschaffung
BYOD	Bring Your Own Device
CASE	Computer Aided Software Engineering
CERN	Europäische Organisation für Kernforschung
CF	Compact Flash
CRM	Customer Relationship Management
CSS	Cascading Style Sheets
DIN	Deutsches Institut für Normung
DNS	Domain Name System
DVB-H	Digital Video Broadcasting – Handhelds
EC	Electronic Cash
Edge	Enhanced Data Rates for GSM Evolution
EDM	Energy Data Management
EEG	Erneuerbare-Energien-Gesetz

eEPK	Erweiterte Ereignisgesteuerte Prozesskette
EITO	European Information Technology Observatory
E-Mail	Electronic Mail
EnWG	Energiewirtschaftsgesetz
EPK	Ereignisgesteuerte Prozesskette
ERM	Entity-Relationship-Modell
ERP	Enterprise Resource Planning
ESMTP	Extended Simple Mail Transfer Protocol
EStG	Einkommenssteuergesetz
EVA	Eingabe-Verarbeitung-Ausgabe
EVU	Energieversorgungsunternehmen
EXX	European Energy Exchange
GB	Gigabyte
GHz	Gigahertz
GPG	GNU Privacy Guard
GPL	General Public License
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
HT	Hochtarif
HTML	Hypertext-Markup-Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IAO (Fraunhofer)	Institut für Arbeitswirtschaft und Organisation
ID	Identifikationsbezeichnung/Identifikationsnummer
IEEE	Institute of Electrical and Electronics Engineers
IMAP	Internet Message Access Protocol
IMSI	International Mobile Subscriber Identity
iOS	Internetwork Operating System
IrDA	Infrared Data Association
IRR	Internal Rate of Return
IT	Informationstechnik
ITK	Informations- und Telekommunikationstechnologie
JDK	Java Development Kit
JIS	Just-in-Sequence
JIT	Just-in-Time
JVM	Java Virtual Machine
KFZ	Kraftfahrzeug
KPI	Key Performance Measures

KWG	Kreditwesengesetz
KWK	Kraft-Wärme-Kopplung
LAN	Local Area Network
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LNI	Lecture Notes in Informatics
LOC	Lines of Code
LTE	Long Term Evolution
MAC	Media Access Control
MDE	Mobile Datenerfassung
MDM	Meter Data Management
MIT	Massachusetts Institute of Technology
MMC	Multimedia Card
MMS	Multimedia Messaging Service
MSB	Messstellenbetreiber
MSD	Messstellendienstleister
NCSS	Non Commented Source Statements
NDK	Native Development Kit
NPV	Net Present Value
NSA	National Security Agency
NT	Niedertarif
OMG	Object Management Group
OS X	Operating System X
OS	Operation System
PAP	Programmablaufplan
PC	Personal Computer
PDA	Personal Digital Assistant
PGP	Pretty Good Privacy
PHP	Hypertext Preprocessor
POP	Post Office Protocol
QR	Quick Response
QSF	Querschnittsfunktionen
RLM	Registrierende Leistungsmessung
ROI	Return on Invest
RUP	Rational Unified Process
S/MIME	Secure/ Multipurpose Internet Mail Extensions
SCM	Supply Chain Management
SD	Secure Digital
SDK	Software Development Kit
SIM	Subscriber Identity Module
SLP	Standardlastprofil
SMR	Smart Meter Reading

SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SSH	Secure Shell
SSL	Secure Sockets Layer
SVK	Sondervertragskunden
SWOT	Strengths-Weaknesses-Opportunities-Threats
TK	Tarifkunden
TLS	Transport Layer Security
UCAN	Ubiquitous Computing Application Development and Evaluation Process
UID	Unique User ID
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
US	United States (of America)
USB	Universal Serial Bus
UUID	Universally Unique Identifier
VM	Virtual Machine
W3C	World Wide Web Consortium
WI-FI	Wireless Fidelity
WLAN	Wireless Local Area Network
XML	Extensible Markup Language
XP	Extreme Programming

---

## Abbildungsverzeichnis

Abb. 1.1	EVA und Interaktion.....	7
Abb. 1.2	Sequenzierung und Parallelisierung von Geschäftsprozessen .....	9
Abb. 1.3	Aufbau des Buchs „App4U“ .....	11
Abb. 2.1	Beweggründe für die Entwicklung mobiler Anwendungen.....	14
Abb. 2.2	Wesentliche Einsatzgebiete mobiler Anwendungen.....	16
Abb. 2.3	Anwendungsbereiche mobiler Applikationen .....	16
Abb. 2.4	Geschäftsfelder des E-Business.....	18
Abb. 2.5	Mobiltelefonbesitzer in Deutschland .....	19
Abb. 2.6	Darstellung von Datenvolumen im Mobilfunk sowie des Umsatzes mit mobilen Datendiensten in Deutschland .....	20
Abb. 2.7	Nutzungsgrade ausgewählter mobiler Aktivitäten in Europa.....	21
Abb. 2.8	Merkmale der B2B- und B2C-Märkte.....	23
Abb. 2.9	Vorgehensweisen bei der Auswahl oder Entwicklung mobiler Applikationen für den B2B- und den B2C-Markt.....	27
Abb. 2.10	Make-or-Buy-Strategien.....	28
Abb. 2.11	Vor- und Nachteile bei der Eigenentwicklung oder dem Fremdbezug von IT.....	28
Abb. 2.12	Marktanteile mobiler Betriebssysteme in Deutschland und auf dem EU5-Markt.....	30
Abb. 2.13	Durchschnittspreise für die mobile Anwendungsentwicklung in Deutschland .....	32
Abb. 3.1	Vorgehensweise Strategie für mobile Applikationen .....	37
Abb. 3.2	Phasen des Brainstorming .....	38
Abb. 3.3	Ergebnis des Brainstormings „Mobile Applikation Energiewirtschaft“ ....	40
Abb. 3.4	Mindmap .....	42
Abb. 3.5	Mindmap „Mobiler Produktkonfigurator“ .....	44
Abb. 3.6	Project Charter Teil 1 .....	47
Abb. 3.7	Project Charter Teil 2 .....	48
Abb. 3.8	Project Charter e-configurator .....	49
Abb. 3.9	Einzelschritte in der Phase Strategiefindung.....	49

Abb. 3.10 Design Thinking .....	50
Abb. 3.11 Marktpotenzial e-configuretor .....	55
Abb. 3.12 Vertriebsziele e-configuretor .....	56
Abb. 3.13 Grundannahmen Business Case e-configuretor.....	57
Abb. 3.14 Varianten Business Case e-configuretor .....	59
Abb. 3.15 Kennzahlensystem ROI.....	61
Abb. 3.16 Business Case e-configuretor „Accelerator“ .....	61
Abb. 3.17 Business Case e-configuretor „Web Service“.....	62
Abb. 3.18 Business Case e-configuretor „Softwarelizenz“ .....	63
Abb. 3.19 Chancen-Risiken-Analyse.....	63
Abb. 3.20 SWOT-Analyse.....	64
Abb. 3.21 SWOT-Analyse e-configuretor.....	65
Abb. 3.22 Produkt- und Serviceportfolio .....	66
Abb. 3.23 Partner .....	66
Abb. 3.24 Aktivitäten und Ergebnisse .....	67
Abb. 3.25 Entscheidungsempfehlung.....	68
Abb. 3.26 Phasen der Strategiedefinition .....	68
Abb. 3.27 First Draft Klassendiagramm e-configuretor.....	71
Abb. 3.28 Preiskomponenten Stromkunden (TK ohne LM).....	72
Abb. 3.29 Einzelschritte in der Phase Strategieumsetzung.....	74
Abb. 3.30 Projektmanagement.....	83
Abb. 3.31 Projektorganisation.....	84
Abb. 4.1 Zusammenhang zwischen Software, -produkt und -system .....	90
Abb. 4.2 Klassifizierung von Software.....	92
Abb. 4.3 Arten von Mobilität .....	95
Abb. 4.4 Durlacher Umfeldanalyse von mobilen Endgeräten.....	98
Abb. 4.5 Übersicht aktueller mobiler Betriebssysteme.....	99
Abb. 4.6 Blockbild eines mobilen Endgerätes .....	101
Abb. 4.7 Akteure und Komponenten eines mobilen Systems.....	104
Abb. 4.8 Nativer, webbasiert und hybrider Ansatz zur mobilen Applikationsentwicklung.....	107
Abb. 4.9 Zusammenhänge zwischen Zeichen, Daten und Information.....	111
Abb. 4.10 Ausgewählte Datentypen zur Programmierung.....	112
Abb. 4.11 Aufbau einer Entwicklungsumgebung .....	122
Abb. 4.12 Grafische Benutzeroberfläche von Eclipse.....	123
Abb. 4.13 Designansicht des App Inventors.....	124
Abb. 4.14 Designansicht von Xcode .....	125
Abb. 4.15 Designansicht von Visual Studio 2010 .....	126
Abb. 5.1 Vorgehensmodell zur mobilen Anwendungsentwicklung .....	135
Abb. 5.2 Aktivitäten und Ergebnisse des Vorgehensmodells zur App-Entwicklung	136
Abb. 5.3 Ordnungsschema von Vorgehensmodellen .....	140
Abb. 5.4 Wasserfallmodell zur Softwareentwicklung.....	142

Abb. 5.5	Spiralmodell zur Softwareentwicklung.....	143
Abb. 5.6	V-Modell zur Softwareentwicklung.....	144
Abb. 5.7	RUP-Modell zur Softwareentwicklung.....	145
Abb. 5.8	Extreme Programming zur Softwareentwicklung.....	148
Abb. 5.9	UCAN-Modell zur mobilen Anwendungsentwicklung.....	149
Abb. 5.10	Bewertung der Vorgehensmodelle hinsichtlich ihrer Grundprinzipien....	151
Abb. 5.11	Bewertung der Vorgehensmodelle hinsichtlich ihrer Phasenunterstützung.....	151
Abb. 5.12	Bewertung der Vorgehensmodelle hinsichtlich ihrer Werkzeugunterstützung.....	152
Abb. 5.13	Bewertung der Vorgehensmodelle hinsichtlich ihrer Komplexität.....	152
Abb. 5.14	Idealtypischer Produktlebenszyklus einer mobilen Anwendung .....	154
Abb. 5.15	Elemente in der Marktphase.....	156
Abb. 5.16	Kunden- und marketplacebezogene Entwicklungsprojekte.....	157
Abb. 5.17	Merkmale zur Problemfindung.....	158
Abb. 5.18	Problem- vs. Lösungsbeschreibung im Entwicklungsprozess .....	159
Abb. 5.19	Prozess von der auslösenden Idee bis zur finalen Lösungsstrategie .....	161
Abb. 5.20	Brainstormingbeispiel zum Thema „Mobile Applikationen“ .....	162
Abb. 5.21	Mindmapbeispiel zum Thema „Mobile Applikationen“.....	165
Abb. 5.22	Beispiel für eine Nutzwertanalyse .....	166
Abb. 5.23	Vorgehensweise in der Planungs- und Konzeptionsphase.....	168
Abb. 5.24	Unterschiedliche Anforderungstypen .....	170
Abb. 5.25	Einbettung der Projektplanung in den Projektablauf .....	171
Abb. 5.26	Ausgewählte Aspekte des Lastenheftes .....	173
Abb. 5.27	Beispiel für den Prozess eines Softwareentwurfs .....	175
Abb. 5.28	Ausgewählte Symbole für Programmablaufpläne nach DIN 66001 .....	178
Abb. 5.29	PAP zur Berechnung des arithmetischen Mittels .....	179
Abb. 5.30	Ausgewählte Symbole für Struktogramme nach DIN 66261.....	179
Abb. 5.31	Struktogramm zur Berechnung des arithmetischen Mittels .....	180
Abb. 5.32	Arten von Beziehungstypen im ERM .....	181
Abb. 5.33	Beispiel eines ER-Modells.....	182
Abb. 5.34	Beispiel einer EPK zum Prozessablauf „Kundenauftrag bestätigen“ .....	184
Abb. 5.35	Beispiel eines BPMN-Modells.....	185
Abb. 5.36	Die Xcode-Entwicklungsumgebung mit Sicht auf das Projekt und zugehörige Ressourcen.....	189
Abb. 5.37	Die Eclipse-Entwicklungsumgebung mit Sicht auf das Projekt und zugehörige Ressourcen.....	190
Abb. 5.38	Designansicht des App Inventors.....	191
Abb. 5.39	Designansicht von Visual Studio 2010 .....	192
Abb. 5.40	Modell des Vorgehens beim Softwaretest.....	194
Abb. 5.41	PhoneGab-Framework .....	196
Abb. 5.42	Titanium-Mobile-Framework.....	197

Abb. 5.43	iOS-Simulator .....	198
Abb. 5.44	Beispiel für das Debugging einer Android-Anwendung.....	199
Abb. 5.45	Step-by-Step-Einführung.....	202
Abb. 5.46	Big-Bang-Einführung.....	203
Abb. 5.47	Gestaltung und Komponenten des Einführungsprozesses von mobilen Anwendungen in Unternehmen .....	204
Abb. 5.48	Geschäftsmodell für den Vertrieb mobiler Anwendungen über einen Onlinestore .....	205
Abb. 6.1	Android software stack.....	223
Abb. 6.2	Angeforderte Rechte einer Android-App vor der Installation.....	226
Abb. 6.3	Zusätzliche Aktivierung von Schutzfunktionen bereits installierter Apps unter Android 4.3 .....	227
Abb. 6.4	Ausgabe von Zertifikatinformationen und Fingerabdruck in Firefox (Desktopversion 24).....	237
Abb. 6.5	Ausgabe von Zertifikatinformationen und Fingerabdruck in Firefox (Android-Version 25).....	238
Abb. 6.6	Aus- und Eingaben (rot) während der Erzeugung des SSL-Schlüsselpaars	239
Abb. 6.7	DJIGZO SSL-Mail mit Verschlüsselung und Signatur unter Android.....	240
Abb. 6.8	Empfang einer verschlüsselten und signierten E-Mail mit DJIGZO .....	240
Abb. 6.9	Einstellung der Signaturprüfung in Android.....	246
Abb. 6.10	Ausschnitt aus „Normen zur IT-Sicherheit“ .....	247
Abb. 7.1	Aktuelle Herausforderungen im Markt für private Baufinanzierung .....	255
Abb. 7.2	Bewertung der Vorgehensmodelle hinsichtlich ihrer Grundprinzipien....	258
Abb. 7.3	Bewertung der Vorgehensmodelle hinsichtlich ihrer Phasenunterstützung.....	258
Abb. 7.4	Bewertung der Vorgehensmodelle hinsichtlich ihrer Werkzeugunterstützung.....	259
Abb. 7.5	Bewertung der Vorgehensmodelle hinsichtlich ihrer Komplexität.....	259
Abb. 7.6	Netzdiagramm zur Analyse mobiler Betriebssysteme.....	260
Abb. 7.7	Gegenüberstellung technologischer Ausprägungen mobiler Betriebssysteme .....	261
Abb. 7.8	Analyse und Bewertung Android-basierter Entwicklungsumgebung.....	263
Abb. 7.9	Projektplan zur Applikationsentwicklung.....	264
Abb. 7.10	Aktivitäten der Entwurfs- und Implementierungsphase .....	267
Abb. 7.11	Unterschiedliche Anforderungstypen .....	268
Abb. 7.12	Musskriterien an die Baufinanzierungs-Applikation.....	270
Abb. 7.13	Wunschkriterien an die Baufinanzierungs-Applikation.....	271
Abb. 7.14	Ausschlusskriterien an die Baufinanzierungs-Applikation.....	271
Abb. 7.15	Vereinfachte Darstellung des Programmablaufs .....	273
Abb. 7.16	Aufbau und Struktur des Design-Editors der App-Inventor- Entwicklungsumgebung.....	275
Abb. 7.17	Aufbau und Struktur des Blocks-Editors der App-Inventor- Entwicklungsumgebung.....	276

Abb. 7.18	Prototypen zur Simulation der Benutzeroberfläche der mobilen Applikation.....	277
Abb. 7.19	Umsetzung der Oberflächengestaltung in App Inventor .....	278
Abb. 7.20	Umsetzung der Funktionen im Blocks-Editor.....	279
Abb. 7.21	Funktionsbaum zur mobilen Applikationsentwicklung.....	281
Abb. 7.22	Deklaration einer Variablen .....	281
Abb. 7.23	Skalierung der Steuerelemente.....	282
Abb. 7.24	Aufruf einer Benutzeroberfläche.....	282
Abb. 7.25	Berechnung der gesamten Objektkosten.....	283
Abb. 7.26	Berechnung des Eigenkapitals in Prozent der Objektkosten.....	283
Abb. 7.27	Berechnung einer Finanzierungsrate .....	284
Abb. 7.28	Komplexität vor der Verbesserung der Laufzeit.....	285
Abb. 7.29	Ergebnisse der Halstead-Metriken vor der Verbesserung der Laufzeit .....	285
Abb. 7.30	Eingabe der Einkommenssituation vor und nach der Verbesserung der Laufzeit.....	286
Abb. 7.31	Komplexität nach der Verbesserung der Laufzeit.....	286
Abb. 7.32	Gegenüberstellung der Halstead-Metriken vor und nach der Verbesserung .....	287
Abb. 7.33	Ablauf des Fehlertests.....	289
Abb. 7.34	Funktion „Einkommen eingeben“ der BauFinanz-Applikation.....	291
Abb. 7.35	Funktion „Ausgaben eingeben“ der BauFinanz-Applikation.....	291
Abb. 7.36	Funktion „Eigenkapital eingeben“ der BauFinanz-Applikation.....	292
Abb. 7.37	Funktion „Finanzierung eingeben“ der BauFinanz-Applikation .....	293
Abb. 7.38	Funktion „Nebenkosten eingeben“ der BauFinanz-Applikation.....	294
Abb. 7.39	Funktion „Objektkosten eingeben“ der BauFinanz-Applikation .....	295
Abb. 7.40	Funktion „Finanzierungsbedarf ausgeben“ der BauFinanz-Applikation...	296
Abb. 7.41	Funktion „Freier Betrag ausgeben“ der BauFinanz-Applikation.....	297
Abb. 7.42	Funktion „Mehrbelastung ausgeben“ der BauFinanz-Applikation.....	297
Abb. 7.43	Fehleranalyse mobile Applikation – mangelnde Lesbarkeit der Benutzereingaben.....	298
Abb. 7.44	Fehleranalyse mobile Applikation – Fehler bei der Dateneingabe.....	299
Abb. 7.45	Fehleranalyse mobile Applikation – Fehler bei der Eingabe von Fließkommazahlen .....	299
Abb. 7.46	Fehleranalyse mobile Applikation – fehlende Nutzungsbedingungen.....	300
Abb. 7.47	Eingesetzte Mobiltelefone beim Integrationstest.....	302
Abb. 7.48	Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der vierten Frage .....	303
Abb. 7.49	Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der sechsten Frage .....	304
Abb. 7.50	Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der zehnten Frage.....	305
Abb. 8.1	Smartphone Marktanteile .....	313

Abb. 8.2	Marktanteil der Android-Versionen.....	316
Abb. 8.3	Vergleich JVM und Dalvik-VM.....	318
Abb. 8.4	Lebenszyklus einer Android Activity.....	320
Abb. 8.5	Beispielhafter Aufbau von View Groups und Views.....	322
Abb. 8.6	SDK-Manager starten.....	324
Abb. 8.7	Android-SDK-Manger.....	325
Abb. 8.8	Starten des Virtual-Device-Managers.....	326
Abb. 8.9	Android Virtual-Device-Manager.....	326
Abb. 8.10	Virtuelles Android-Gerät.....	327
Abb. 8.11	Entwicklungsoberfläche des App Inventors.....	327
Abb. 8.12	Eignung von Bluetooth und Wireless-LAN .....	332
Abb. 8.13	Sequenzdiagramm zur Steuerung der Präsentation.....	336
Abb. 8.14	Auszug aus einer PowerPoint-XML-Datei .....	340
Abb. 8.15	Sequenzdiagramm – Laserpointer.....	342
Abb. 8.16	Bildschirmaufnahme Activity 1: Verbindungsaubau.....	343
Abb. 8.17	Bildschirmaufnahme Activity 2: Präsentationsauswahl.....	344
Abb. 8.18	Bildschirmaufnahme Activity 3: Steuerung .....	345
Abb. 8.19	Bildschirmaufnahme der Benachrichtigungsleiste.....	345
Abb. 8.20	Bildschirmaufnahme Activity 4: Folienauswahl.....	346
Abb. 9.1	Nutzungsdauer von mobilen Applikationen.....	355
Abb. 9.2	Das Prototypingmodell.....	356
Abb. 9.3	Schichtenmodell einer DB-Java-Anwendung.....	357
Abb. 9.4	JDBC-2-Schichten-Architektur .....	358
Abb. 9.5	App-Kategorien von Apple und Google .....	360
Abb. 9.6	Zusammenfassung einiger App-Kategorien.....	360
Abb. 9.7	Die App-Kategorien des Ratingmodells.....	361
Abb. 9.8	App-Infos bei Google Play und im Apple-Store.....	362
Abb. 9.9	Allgemeine Informationen des Ratingmodells.....	363
Abb. 9.10	Altersfreigaben im Ratingmodell .....	363
Abb. 9.11	Allgemeine Informationen zum „WhatsApp Messenger für Android“ ....	364
Abb. 9.12	Gemeinsame Kriterien für Apps.....	365
Abb. 9.13	Sonstige App-Kategorien.....	370
Abb. 9.14	Kriterien für die restlichen App-Kategorien .....	371
Abb. 9.15	Legende zum Bewertungssystem .....	371
Abb. 9.16	Aufrufen von phpMyAdmin .....	374
Abb. 9.17	Anlegen der App-Datenbank.....	374
Abb. 9.18	Anlegen einer Tabelle.....	374
Abb. 9.19	Anlegen der Spalte „Name“ .....	374
Abb. 9.20	Struktur einer Tabelle .....	375
Abb. 9.21	Die App-Datenbank.....	375
Abb. 9.22	GUI des Rating-Tool-Prototyps .....	377
Abb. 9.23	Panel der Kategorie Spiele.....	377

Abb. 9.24 Die Methode „getNameField“ .....	378
Abb. 9.25 Anzeige der Tabellen .....	378
Abb. 9.26 Wechseln zwischen den Fenstern .....	379
Abb. 9.27 Die Klasse MySQLConnection .....	379
Abb. 9.28 Aufbau der URL .....	379
Abb. 9.29 Herstellen der Verbindung .....	380
Abb. 9.30 Senden der SQL-Abfragen .....	380
Abb. 9.31 Schließen der Verbindung .....	380
Abb. 9.32 Die Klasse „PanelSwitch“ .....	381
Abb. 9.33 Auszug aus der „changePanel“-Methode .....	382
Abb. 9.34 Die Klasse „createQuerys“ .....	382
Abb. 9.35 Initialisieren der Parameter .....	382
Abb. 9.36 Generieren des ersten SQL-Befehls .....	383
Abb. 9.37 Generieren des zweiten SQL-Befehls .....	383
Abb. 9.38 Die Klasse „calculateRating“ .....	383
Abb. 9.39 Addition der Bewertungen .....	384
Abb. 9.40 Berechnen des Ratings .....	384
Abb. 9.41 Die Klasse „DBView“ .....	385
Abb. 9.42 Die Drop-down-Menüs .....	385
Abb. 9.43 Die Methode „showTableContent“ .....	386
Abb. 9.44 Das Table Model .....	386
Abb. 9.45 Beispiel zur SQL-Abfrage .....	386
Abb. 9.46 Die Tabelleninhalte dem „JTable“ übergeben .....	387
Abb. 9.47 Table Model an „JTable“ übergeben .....	387
Abb. 9.48 Auslesen des Tabelleninhaltes .....	388
Abb. 9.49 Die Klasse „Start“ .....	388
Abb. 9.50 Der Aufruf des Konstruktors .....	388
Abb. 9.51 Der Log-in-Dialog .....	388
Abb. 9.52 Die Klasse „LoginDialog“ .....	389
Abb. 9.53 Menüpunkt „Datenbank“ .....	389
Abb. 9.54 Anpassen des Verbindungsbaus .....	390
Abb. 9.55 Die Erfolgsmeldung .....	390
Abb. 9.56 Die Fehlermeldung .....	391
Abb. 9.57 Das Klassendiagramm des Rating Tools .....	392
Abb. 9.58 Allgemeine Informationen zur App .....	392
Abb. 9.59 Allgemeine Bewertungskategorien von „WhatsApp“ .....	393
Abb. 9.60 Spezielle Bewertungskategorien von „WhatsApp“ .....	394
Abb. 9.61 Eintragen der allgemeinen Informationen .....	395
Abb. 9.62 Abgeben der Bewertungen .....	395
Abb. 9.63 Meldung über erfolgreiche Speicherung .....	396
Abb. 9.64 Ausschnitt der Anzeige der „WhatsApp“-Daten .....	396
Abb. 9.65 Das Ratingmodell für mobile Applikationen .....	397
Abb. 9.66 Informationsaustausch zwischen den Komponenten .....	398

---

# App4U – Die Welt der mobilen Applikationen

1

Christian Aichele und Marius Schönberger

*Mobile Applikationen als Beschleuniger (Accelerator) für neue Geschäftsmodelle*

---

## Zusammenfassung

Mobile Applikationen (Apps) sind insbesondere dafür bekannt, dass sie Konsumenten Unterhaltung und partiellen Mehrwert bieten. In Unternehmen werden Apps bisher vorrangig für die mobile Kommunikation sowie im Marketing und Vertrieb genutzt. Die ständige Weiterentwicklung sowie der technologische Fortschritt machen jedoch zukünftig einen Ausbau der Anwendungsbereiche in Unternehmen möglich. Mobile Applikationen unterscheiden sich von Desktopanwendungen hinsichtlich der Art der Endgeräte, wie z. B. Smartphones oder Tablet-PCs, welche die Applikationen schneller und ortsunabhängig verfügbar machen. Für einen andauernden Erfolg der smarten Programme wird die Generierung neuartiger Geschäftsmodelle und innovativer Geschäftsprozesse, die einen echten Added Value für die Unternehmen bieten, entscheidend sein.

---

C. Aichele (✉)

Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: christian.aichele@fh-kl.de

M. Schönberger

Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: marius.schoenberger@fh-kl.de

## 1.1 Apps, eine Erfolgsgeschichte im B2C

Kleine smarte Programme für die grafischen Oberflächen von Personal Computern (PCs) gibt es seit einigen Jahren. Diese sogenannten Gadgets oder Widgets stellen Informationen aus Newstickern, Börsenwerte, Wetterdaten oder Uhrzeiten zur Verfügung und laufen auf Basis sogenannter Engines (Widget-Engines). Sie stellen somit Subprogramme dar und haben eine den Apps ähnliche Erscheinungsform, sind aber nicht in sich gekapselt. Diese Gadgets erfreuen sich hoher Beliebtheit und stellen die evolutionären Vorgänger der mobilen Anwendungen dar.

- ▶ Mit dem Begriff **Gadget** werden kleine technische Spielereien oder Geräte bezeichnet, die sich durch keine oder wenig Funktionalität und einem originellen Design auszeichnen. Als Gadget wurden anfänglich auch kleine Programme für grafische Benutzeroberflächen bezeichnet, die dem Benutzer Informationen bieten. Mittlerweile hat sich dafür der Begriff Widget durchgesetzt.
- ▶ **Widgets** sind Programmkomponenten grafischer Benutzeroberflächen. Widget ist ein Kunstwort aus Windows und Gadget. Widgets werden in einem Fenster auf einer grafischen Benutzeroberfläche dargestellt und reagieren auf Input oder Benutzerinteraktionen. Im Gegensatz zu Apps verwenden Widgets die vom grafischen Benutzersystem angebotenen Dienste und Fenster.
- ▶ **Widget-Engines** sind Softwarekomponenten von Betriebssystemen mit grafischen Benutzeroberflächen und stellen die Funktionalität zur Einbindung und Nutzung von Widgets bereit.

Apps haben mittlerweile eine immense Verbreitung gefunden. Die kleinen smarten Alles- und Nichtskönner bieten eine unglaubliche Vielfalt sinniger, aber auch unsinniger Anwendungen. Dies scheint aber die Anwender nicht abzuschrecken. Im Gegenteil: Aufgrund der geringen Preise werden Apps ausprobiert, ggf. wenige Tage angewendet und dann wieder gelöscht. Nur wenige Apps schaffen es in eine permanente Nutzung. Die Leichtigkeit der Installation, die allgegenwärtige Verfügbarkeit, die breite Anwendbarkeit und die permanente Angebotserweiterung wecken die Neugierde und den Spieltrieb der Anwender. Beschwerden über unsinnige Anwendungen mit ggf. betrügerischem Hintergrund beschränken sich in der Regel auf negative Rezensionen und haben nur selten Folgen für den Entwickler der Applikation.

Aber was war und ist verantwortlich für den Erfolg der mobilen Applikationen, wodurch differenzieren sich die Apps von bisheriger Anwendungssoftware?

- **Der Erfolg der Smartphones, insbesondere des Trendsetters iPhone:**

Erst die Einführung des iPhones mit seinem Touchscreen, dem in den Hintergrund gerückten Betriebssystem und der über die normalen Telefonfeatures hinausgehenden Funktionalität sorgte auf der Anbieter- und Nachfragerseite für einen enormen Schub in der Nutzung der mobilen Applikationen.

- **Die neuen Devices, Smartphones, Tablets, Ultrabooks, TVs u. a.:**

Neben den Smartphones hat insbesondere die Verbreitung der Tablet-PCs für einen Zuwachs des App-Angebots gesorgt. Angefangen mit dem Trendsetter iPad auf Basis des Betriebssystems Apple iOS folgten unzählige Tablet-PCs auf Android-Basis. Im Vergleich zu stationären PCs sind Tablet-PCs günstiger und einfacher in der Handhabung, erfordern keinen Netzanschluss, sind mobil und halten die wichtigsten Funktionalitäten und Kommunikationsmöglichkeiten permanent bereit. Mittlerweile finden sich mobile Applikationen auch auf den sogenannten Ultrabooks, die Features der Tablet-PCs und stationären PCs integrieren, sowie auch auf nicht aus der PC-Welt stammenden elektronischen Geräten wie TVs, Smart Home Steuerungen, Haushaltsgeräten, Steuerungsgeräten der Energieerzeugung und vielen weiteren. Dadurch hat Software in Form von Apps zahlreiche neue Anwendungen und auch nicht softwareaffine Benutzergruppen erreicht.

- **Die intuitive Benutzerführung:**

Die Touchscreens erlauben eine intuitive Benutzung durch Berührung mit den Fingern. Die reduzierte Funktionalität ermöglicht einen kognitiv einfachen Learning-by-Doing-Ansatz. Selbst nicht IT-affine Benutzergruppen können relativ schnell mit den simplen mobilen Applikationen umgehen. Dadurch, dass die meisten Applikationen auch für Apps mit kleinen Bildschirmen optimiert sind, ist die Erfassung der wenigen Funktionspunkte schnell möglich.

- **Die einfache Funktionalität:**

Die meisten mobilen Anwendungen verfügen im Vergleich zu PC-gestützten Softwareanwendungen über einen relativ reduzierten Funktionsumfang. Apps verfügen typischerweise über einen dedizierten Anwendungsbereich mit entsprechend limitierten Funktionalitäten. Dadurch wird eine größere Anwendergruppe angesprochen, die Hürde sich mit einer unbekannten Software auseinanderzusetzen ist klein.

- **Die fehlende Komplexität der Nutzung, keine Maus, keine Tastatur, kein Stift:**

Einzig benötigte Benutzerschnittstelle ist der Touchscreen. Über eingeblendete Tastaturen können komplexere Texte eingegeben werden. Das Handling einer Maus oder die Arbeit mit einem Stift ist nicht notwendig. Durch die Nähe zu der Handhabung analoger Medien wie Zeitschriften oder Büchern wird die Nutzung der Apps für alle möglich.

- **Die Integration analoger Handhabung:**

Das Blättern in Zeitschriften oder Büchern wird in vielen Apps simuliert. Die tabletartige Oberfläche kann durch Schieben oder Wischen den Zugriff auf viele weitere mobile Applikationen ermöglichen. Voicefunktionalitäten ermöglichen den Aufruf von Funktionen durch Spracheingabe. Das Hin- und Herschieben der Apps durch Ge-

drückthalten des Buttons erinnert an das Bewegen von Bausteinen in einem Baukasten. Diese Digitalisierung analoger Automatismen ermöglicht die schnelle Erlernbarkeit der mobilen Betriebssysteme und auch der mobilen Applikationen. Zukünftige Benutzergenerationen benötigen diese Analogien zu dem analogen Handling sehr wahrscheinlich nicht mehr. Sie sind von frühester Jugend an mit den Touchscreenfunktionalitäten vertraut. Aber auch Senioren und nicht IT-affinen Nutzern wird durch diese Nähe und die immensen, Neugierde und Interesse schaffenden Möglichkeiten der mobilen Applikationen die digitale Welt schnell vertraut.

- **Die Gewinnung neuer Benutzergruppen:**

Wie schon in den obigen Beispielen aufgezeigt, haben die mobilen Applikationen ihr Potenzial, durch einfaches Handling, intuitive App-Funktionalitäten und innovative Ansätze neue Benutzergruppen zu gewinnen, genutzt. Insbesondere bisher nicht IT-affine Personengruppen, wie z.B. Senioren u.a., wurden durch touchscreenbasierte Devices an die Informationstechnologie herangeführt. Mit der schnellen Verbreitung neuer Applikationen und neuer Devices, wie z.B. die Touchscreensteuerungen von TV-Geräten, KFZ-Multimedia und Navigationssystemen, Steuerungen von Haushaltsgeräten und weitere, wird sich dieser Trend noch enorm beschleunigen.

- **Der Added Value:**

Mobile Applikationen bieten in vielerlei Hinsicht Mehrwert. Durch die permanente Onlineverbindung können jederzeit beliebige Anfragen an Onlineenzyklopädien, Onlinewörterbücher, Navigationsdienste und viele andere gerichtet werden. Navigationssysteme helfen Wege zu finden, Einkaufs-Apps unterstützen bei der Suche nach Schnäppchen oder den passenden Shops und halten Einkaufslisten bereit, die ggf. mit dem Partner oder der Familie online abgestimmt wurden, Stadtführer zeigen markante und sehenswerte Lokationen, Restaurantführer helfen bei der Auswahl nahe gelegener Möglichkeiten zum Essengehen, Hotelführer zeigen adäquate Übernachtungsmöglichkeiten, Terminplaner erinnern an Verabredungen, Veranstaltungsempfehlungen können abgerufen werden oder werden per Pushmeldung vorgeschlagen und vieles mehr. Durch die interaktive und jederzeit mögliche Kommunikation bieten die Apps ihren besonderen Mehrwert an.

- **Die Allgegenwart:**

Apps haben den Einzug in die analoge Welt geschafft. Analoge Medien verweisen per QR-Code auf weiterführende Informationen durch mobile Applikationen. Apps finden in elektronischen Geräten Anwendung, die von Computern weit entfernt scheinen. Über die meisten modernen TV-Geräte können Apps aufgerufen werden. Steuerungen von Haushaltsgeräten verwenden Apps. Zeitungen verweisen auf Apps für politische Wahlen. Institutionen und Organisationen stellen sich per Apps vor. Die mobilen Applikationen haben durch die permanente Verfügbarkeit der mobilen Devices und durch die Integration der analogen in die digitalen Medien eine De-facto-Allgegenwart erreicht.

- **Das riesige Angebot:**

Die Entwicklung von Apps ist relativ einfach. Innerhalb weniger Tage können selbst komplexe Funktionalitäten integriert werden. Der Vertrieb wird durch etablierte App-Stores, digitale Marktplätze für den Verkauf der Apps, übernommen. Weltweit entwickeln Unternehmen, Einzelunternehmer und Privatpersonen die Applikationen für eine Vielzahl von Zwecken. Anders als herkömmliche Softwareprodukte werden Apps zum Teil auch nur sehr kurzweilig eingesetzt. Aufgrund des riesigen Angebots wird zur Auswahl konkurrierender Apps auch der Kauf und anschließende Test präferiert. Selbst suboptimale Apps haben einen Markt. Das befähigt die App-Entwickler schneller und kreativer Marktlücken und neue Markttrends zu decken bzw. neue Ideen zu kopieren. Ein verifiziertes und validiertes Testen der Apps findet nicht statt. Einzig die E-Community entscheidet über Erfolg oder Nichterfolg der Applikationen. Gute Bewertungen in den App-Stores führen zum Absatz und zu weiteren guten Bewertungen.

- **Die leichte und schnelle Entwicklung:**

Versierte Entwickler können ohne Probleme mehrere Apps pro Woche erstellen. Aufgrund der vorgegebenen Betriebssysteme und deren vorhandenen und abrufbaren Oberflächen, Menüführung und Integration der Sensoren und Aktoren der Devices kann der Fokus der Entwicklung auf den Kern der Programme (die Algorithmen) gerichtet werden. Vorhandene Entwicklungsumgebungen bieten umfangreiche Frameworks mit einsetzbaren Klassenbibliotheken an. CASE-Umgebungen (Computer Aided Software Engineering) haben mit dem App-Zeitalter eine Renaissance erlebt. Und gerade weil der Funktionsumfang der mobilen Applikationen limitiert ist, scheint der CASE-Ansatz hier um einiges erfolgversprechender als der Versuch, in den 90ern große monolithische Softwaresysteme mit CASE zu beherrschen.

- **Der globale Markt:**

Die mobilen Apps werden über die App-Stores weltweit angeboten. Selbst monolinguale Apps werden weltweit von den Usern angenommen. Woran liegt das? Die einfachen Oberflächen, die reduzierte Funktionalität und die Visualisierung der Funktionen ermöglichen dem Anwender das Erkennen des Prinzips auch ohne die jeweiligen Sprachkenntnisse. Zahlreiche Apps werden auch nur englischsprachig angeboten, eine Sprache, die von dem überwiegenden Teil der App-Konsumenten verstanden und auch gesprochen wird. Die oftmals von den monolithischen Softwaresystemen geforderte und aufgrund der Komplexität auch erforderliche Multilingualität ist für Apps verzichtbar. Bilinguale Versionen erreichen einen Großteil des Marktes.

- **Das Trial-and-Error-Prinzip:**

Im Gegensatz zu PC-gestützten Softwareprodukten werden Apps ohne große Evaluation gekauft, heruntergeladen, ausprobiert, ggf. genutzt und dann wieder gelöscht oder einfach nicht mehr verwendet. Aufgrund der geringen Preise, der schnellen Verfügbarkeit und der Bewertungen der E-Community wird die Entscheidung für eine App schnell gefällt. Der Ärger über eine nicht nutzbare App bzw. über unnütze Apps hält sich in Grenzen. Die Vielfalt macht es. So lange der Großteil der Applikationen sinnvolle Features enthält, wird der Reinfall bei einigen wenigen Apps akzeptiert.

- **Die geringen Preise:**

Viele Apps werden kostenlos bzw. als Open-Source-Produkte angeboten. Die App-Entwickler verdienen dann ggf. an der inkorporierten Werbung. Zahlreiche Apps werden auf einem niedrigen Preislevel als Vollversion oder als Version mit limitiertem Funktionsumfang bzw. limitierter zeitlicher Nutzbarkeit angeboten. Nur wenige Apps bewegen sich in dem mittleren oder hohen Preissegment. Diese kostenlosen Versionen bzw. geringen Preise haben auch zu einem geänderten Nutzungsverhalten geführt. Software in Form von Apps wird konsumiert und nach dem Konsum verworfen. Nur einige wenige Applikationen finden ihren Weg in eine permanente Nutzung.

- **Das kostenlose Angebot:**

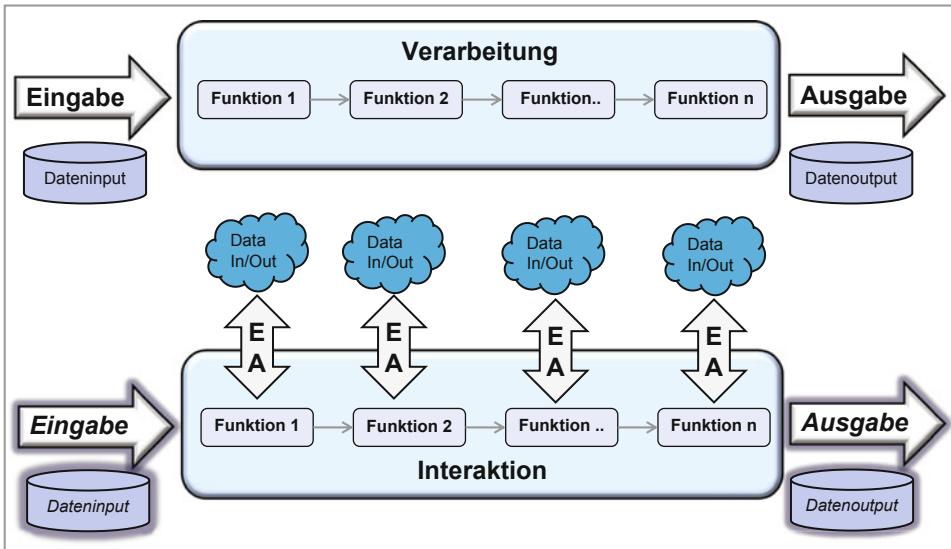
Die ersten Formen kostenloser PC-Software gab es als sogenannte Freeware und Shareware. Später kamen die Open-Source-Produkte. Viele Apps können als Vollversion bzw. als Version mit einem limitierten Funktionsumfang kostenlos verwendet werden. Diese Applikationen enthalten zum großen Teil Werbung und werden durch diese Werbung auch finanziert. In diesem Sinne sind die Apps Freeware. Ein mengenmäßig großer Anteil der Apps kann auch zu geringen Preisen bezogen werden, die für die potenziellen Nutzer nahezu kostenlos erscheinen (Centbeträge).

- **Freeware** ist eine vom Entwickler kostenlos bereitgestellte Software. Eine Offenlegung des Codes zur weiteren Entwicklung ist meistens nicht beinhaltet.
- **Shareware** ist eine vom Entwickler für einen limitierten Zeitraum kostenlos zur Verfügung gestellte Software. Nach Ablauf des Zeitraums ist eine weitere Nutzung nur durch eine (ggf. kostenpflichtige) Lizenzierung möglich.
- **Open Source** ist eine (kostenlose) lizenzierte Software, deren Quellcode offen gelegt ist, verändert und erweitert werden darf und (kostenlos) unverändert oder verändert weitergegeben werden darf.

- **Die Interaktivität:**

Die monolithischen Softwaresysteme folgen dem EVA-Prinzip (Eingabe-Verarbeitung-Ausgabe, siehe Abb. 1.1). Die mobilen Applikationen basieren zum großen Teil auf Kommunikation und Interaktion. Die Apps sind mit der E-Community verbunden. Auf Dateneingaben bzw. Aktionen erhält der Anwender weitere Dateneingaben bzw. Reaktionen, die dann zu weiteren eigenen Aktionen führen. Diese Interaktion kann von beliebiger Dauer sein. Ein initialer Dateninput (Eingabe) und ein finaler Datenoutput (Ausgabe) sind nur optional. Diese Interaktion macht auch den Reiz der mobilen Applikationen aus. Sie ist nie gleich und auch nicht vorhersehbar. Jeder Neustart der Applikation führt zu anderen Interaktionen und Abläufen. Viele Anbieter von Spieleapplikationen verwenden diese Interaktion auch als ihr Geschäftsmodell. Push Messages bieten dem User (Spieler) Zusatzfeatures an, die das Spiel vereinfachen, interessanter machen, einem Vorteile verschaffen, aber auch kosten. Dadurch erwirtschaften auch vordergründig kostenlose Apps Erlöse.

**Das EVA-Prinzip** beschreibt die grundsätzliche Funktionsweise von Software. Auf Basis von über Benutzerschnittstellen eingegebenen Daten finden funktionale Verarbeitungsschritte statt. Die Verarbeitung besteht ggf. aus mehreren sequenziellen Funktionen. Abgeschlossen wird der Prozess durch die Ausgabe der verarbeiteten Daten.



**Abb. 1.1** EVA und Interaktion

- **Die Marktplätze:**

Die Marktplätze für mobile Applikationen sind zum Teil offen (Betriebssystem Android) und zum Teil proprietär (Betriebssysteme Apple iOS, Windows Phone, Symbian). Die bekanntesten und am meisten frequentierten Marktplätze für die einzelnen Betriebssysteme sind:

- Android Marktplätze (Amazon App-Store für Android, Google Play Store, Samsung App-Store),
- Apple iOS Marktplätze (iTunes App Store, Mac App Store),
- Windows Phone Apps (Windows Phone Store),
- Symbian (Nokia App-Store),
- BlackBerry OS (BlackBerry World) und
- GetJar (Drittanbietermarktplatz mit Apps für Android, Mobile Windows, Symbian und BlackBerry).

Aufgrund der reduzierten Unterstützung des Smartphonebetriebssystems Symbian, des Kaufs von Nokia Phone durch Microsoft und der Nokia-Strategie, insbesondere Smartphones mit dem Microsoft-Betriebssystem Windows Phone anzubieten, wird der Nokia-Store kurz- bis mittelfristig keine größere Rolle mehr spielen. Android-Apps können auch über beliebige Stores oder Webserver bezogen werden und werden ggf. über den jeweiligen Anbieter zertifiziert oder auch nicht. Apple-iOS-Apps, Windows-Phone-Apps, Symbian-Apps und BlackBerry-Apps müssen durch den jeweiligen Anbieter des proprietären Stores zertifiziert werden. GetJar selbst ist eine mobile Android-Applikation, die über den Google Play Store heruntergeladen werden kann und mobile Apps für unterschiedliche Plattformen abrufbar macht. Die meisten Apps werden durch Drittanbieter entwickelt, die Entwicklungsumgebung wird von

dem Betriebssystemanbieter in der Regel kostenlos zur Verfügung gestellt. Die Marktplätze bieten einen globalen Zugang und eine sofortige Installationsmöglichkeit der gewünschten Apps.

- **Die Verschmelzung von Soft- und Hardware:**

Erst die mobilen Applikationen machen die mobilen Devices interessant. Smartphones, Tablet-PCs und Ultrabooks werden mit attraktiven Applikationen zum Kauf angeboten. Das Apple iPhone oder das Apple iPad verdanken ihren Erfolg den Apps. Die Apps wurden durch die attraktiven Devices zu präferierten Softwareprodukten. Nur durch die Symbiose der Soft- mit der Hardware werden attraktive und begehrswerte Produkte geschaffen. Der Erfolg wird durch diese Integration und durch permanente Innovationen auf der Hard- und Softwareseite auch in der Zukunft gesichert sein.

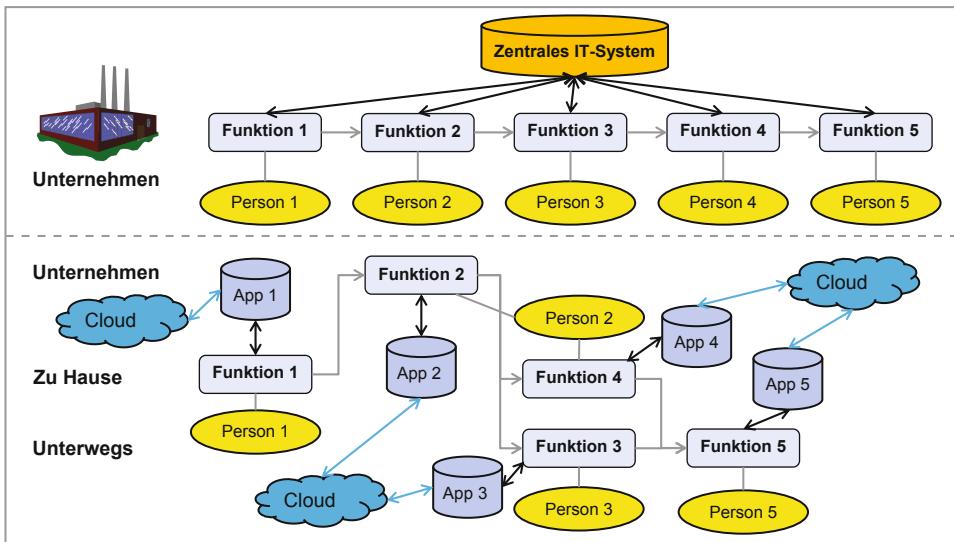
- **Mobile Applikationen** (Kurzform App) sind Softwareanwendungen, die in Form von gekapselten Programmen auf mobilen Endgeräten lauffähig sind.

### 1.1.1 Erfolg durch Apps im B2B?

Die Systeme im Bereich Business sind immer noch geprägt durch proprietäre und monolithische Systeme. Auch der weitestgehend erfolgte Wechsel auf serviceorientierte Architekturen hat nur in den Randbereichen der unterstützten Geschäftsprozesse für eine größere Variabilität der Systeme gesorgt. Der Einsatz von Cloud Services wird mit großer Vorsicht angegangen. Die Furcht vor Datendiebstahl und Industriespionage ist riesig. Die meisten Unternehmen sehen hier ganz klar den Vorteil eigener und abgeschotteter Systeme. Bring Your Own Device (BYOD) und der Einsatz von Apps außerhalb von Vertrieb und Marketing ist zumindest im europäischen Raum und insbesondere in Deutschland kein Hypothema.

Aber was bisher der serviceorientierten Architektur nicht gelungen ist, könnte durch den Einsatz von mobilen Applikationen zu einer Revolution führen. Der arbeitsplatz- und zeitunabhängige Einsatz von IT-unterstützten Unternehmensfunktionen sorgt nicht nur für eine schnellere Durchführung der Geschäftsprozesse, sondern kann auch zu ganz neuen Arbeitsmodellen führen. Die redundant vorhandene Möglichkeit, die Funktionen auf unternehmensinternen Devices und eigenen Geräten jederzeit und an jedem Ort durchzuführen, erlaubt eine nicht für möglich gehaltene Variabilität im Einsatz der individuellen Arbeitszeit. Selbst Meetings und Workshops können zum großen Teil unabhängig von der physischen Anwesenheit über mobile Applikationen erfolgen. Nur noch erstmalige Treffen oder solche, die von besonderer Wichtigkeit sind (z. B. Erstmeetings mit potenziellen Kunden, Investoren u. a.), erfordern das persönliche Erscheinen. Der mobile Mitarbeiter benötigt weniger Ressourcen (Raum, Energie, Infrastruktur) und ist letzten Endes durch die Selbstgestaltung der Arbeitszeiten und Arbeitsintensitäten auch zufriedener mit dem beruflichen Umfeld.

Dies ist aber nur ein Schlaglicht auf den Einsatz von Apps im B2B. Auch die Organisationsabläufe bzw. die Geschäftsprozesse müssen nicht einem vorstrukturierten, sequenziellen Ablauf folgen, sondern können je nach Bedürfnissen und Gegebenheiten neu sequenziert und parallelisiert werden.



**Abb. 1.2** Sequenzierung und Parallelisierung von Geschäftsprozessen

In der Abb. 1.2 ist im oberen Teil der übliche Geschäftsprozess auf Basis einer zentralen Informationstechnologie dargestellt. Die Einzelfunktionen eines Geschäftsprozesses werden in einer strukturierten und nicht abänderbaren Sequenz durchgeführt. Für die Anwendung der Funktionen ist die Anwesenheit der einzelnen Mitarbeiter/Personen in der Regel die Voraussetzung. Im unteren Teil ist der IT-unterstützte Unternehmensprozess in einzelne mobile Applikationen aufgesplittet. Person 1 startet den Geschäftsprozess durch Anwendung der Funktion 1 des Prozesses zu Hause. Die zu generierenden bzw. zu übergebenden Daten werden an eine cloudbasierte Datenhaltung übergeben. Person 2 befindet sich im Unternehmen (eine Anwesenheit am eigenen Arbeitsplatz ist nicht notwendig) und führt die Funktion 2 durch. Person 2 erkennt, dass Sie auch befähigt und berechtigt ist, die Funktion 4 durchzuführen. Diese wird dann zu Hause erledigt. Person 3 ist unterwegs und führt parallel die Funktion 3 durch. Abschließend wird durch Person 5, auch unterwegs, die Funktion 5 erfüllt. Außer im Unternehmen kommen nur eigene Devices (Smartphones, Notebooks, Tablet-PCs, TV) zur Ausführung der Funktionen zum Zuge.

Das große Problem ist das rechtzeitige Erkennen der Potenziale, bevor nur wieder eine Me-too-Attitüde möglich ist. Die Strategien und gewählten Geschäftsmodelle für die App-Anwendung im Bereich B2B bringen die entscheidende Weichenstellung für das jeweilige Unternehmen (siehe Kap. 3). Aber nur wer bereit ist, seine Organisation und Geschäftsprozesse im Hinblick auf den Einsatz mobiler Applikationen zu analysieren, wird zu den Innovatoren gehören können.<sup>1</sup>

<sup>1</sup> In dem Prozess der Annahme einer Innovation können die Unternehmen in folgende Gruppen klassifiziert werden (vgl. Aichele und Doleski 2013, S. 25): Innovatoren (die ersten 5 bis 10 %, die eine Innovation annehmen); Early Adopters (die nächsten 10 bis 15 %); Frühe Mehrheit (weitere 30 %); Späte Mehrheit (weitere 30 %); Laggards (Nachzügler) (die verbleibenden 20 %).

### 1.1.2 App4U – Ein Leitfaden für die App-Entwicklung

Die Nutzung von mobilen Endgeräten bezog sich bisher auf den Austausch von Informationen zwischen Menschen. Durch den technologischen Fortschritt, den Ausbau mobiler Breitbandnetze und die Weiterentwicklung internetbasierter Anwendungen, haben mobile Endgeräte heutzutage einen neuen Stellenwert eingenommen. Die Verwendung mobiler Endgeräte und Anwendungen bezieht sich heute nicht mehr auf die reine Kommunikation, sondern auf die Interaktion zwischen zwei oder mehreren Personen. Internetbasierte Dienste, wie z. B. das Instant Messaging, bieten neben der herkömmlichen textbasierten Kommunikation eine Bandbreite an interaktiven Möglichkeiten an, mit den Gesprächspartnern in Kontakt zu treten. Mittlerweile existiert eine große Bandbreite an mobilen Anwendungen, um solche internetbasierten Dienste für die Allgemeinheit anzubieten. Dies ist nur ein Grund dafür, warum softwareentwickelnde Unternehmen vor der großen Herausforderung stehen, immer wieder neue und innovative Anwendungen zu entwickeln, die sich von der breiten Masse abheben und Alleinstellungsmerkmale aufweisen. Weiterhin bestehen technische Herausforderungen, wie z. B.<sup>2</sup>

- **Integration der Geschäftsabläufe und die vorhandene IT-Landschaft:**

Die Anbindung von mobilen Anwendungen an die Unternehmens-IT ist unerlässlich und kann aufgrund der heutigen Vielfalt der Anwendungssysteme und Schnittstellen zu diesen Systemen zu Problemen bei der Integration in die Systemlandschaft führen. Des Weiteren kann bei der Einführung mobiler Anwendungen nicht immer auf standardisierte Verfahren zurückgegriffen werden, da einerseits keine einheitlichen Vorgehensweisen existieren und andererseits unterschiedliche und nicht auf mobile Anwendungen abgestimmte Unternehmensprozesse die Festlegung eines einheitlichen Einführungsprozesses verhindern.

- **Die Heterogenität der mobilen Betriebssysteme:**

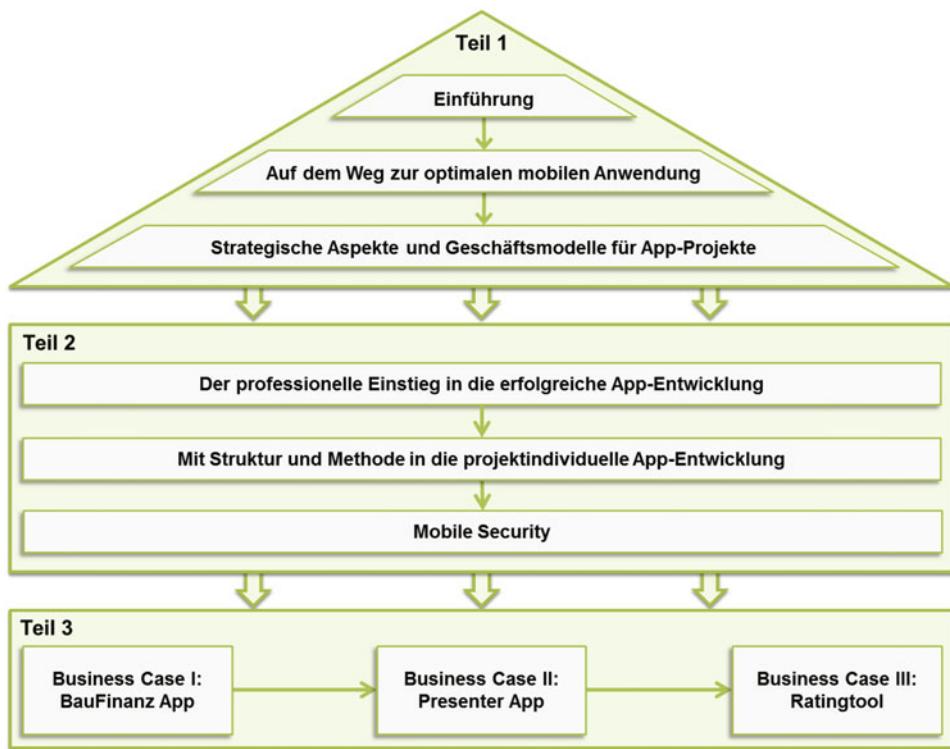
Am Markt für mobile Anwendungen besteht aktuell eine Vielzahl an unterschiedlich ausgeprägten Betriebssystemen, wie z. B. Android, iOS oder Windows Phone 8, welche aufgrund verschiedener Funktionsmöglichkeiten und Entwicklungsansätze die Herstellung und den Vertrieb von mobilen Applikationen sowie das Management der zugehörigen mobilen Endgeräte erschweren.

- **Die Skalierbarkeit mobiler Anwendungen:**

Während individuell entwickelte mobile Anwendungen für eine geringe Anzahl Nutzer relativ leicht zu vermarkten und zu warten sind, stellt sich für mobile Anwendungen, die auf eine große Anzahl Nutzer ausgerichtet sind, die Frage nach der Skalierbarkeit der Applikation, also, wie eine Vielzahl Benutzer zeitgleich mit der Applikation ausgestattet und wie ggf. Wartungsarbeiten durchgeführt werden können.

---

<sup>2</sup> Vgl. Vercales und Linhoff-Popien (2012, S. 10 f.).



**Abb. 1.3** Aufbau des Buchs „App4U“

- **Die Sicherheit mobiler Anwendungen:**

Die Gefahren von mobilen Anwendungen sind vielfältig. Beim täglichen Anwenden mobiler Applikationen werden ständig mehrere Parteien mit ins Spiel gebracht, die Zugriff auf verschiedene Informationen des Benutzers oder des Endgerätes benötigen. Eine Sicherung privater Daten sowie eine Überprüfung der am verwendeten Internetdienst beteiligten Personen ist damit zu empfehlen. Insbesondere beim BYOD-Ansatz ist eine Trennung privater und geschäftlicher Daten kaum zu ermöglichen.

Das vorliegende Buch soll die oben genannten Herausforderungen und die in den ersten beiden Kapiteln beschriebenen Erfolgsfaktoren stärker beleuchten und einen Leitfaden für die erfolgreiche mobile Anwendungsentwicklung bereitstellen. Hierzu wurden unterschiedliche Schwerpunkte gesetzt, wie z. B. (vgl. Abb. 1.3):

- Beweggründe für die Entwicklung mobiler Anwendungen und Nutzungsgrade ausgewählter mobiler Aktivitäten,
- Vorstellung aktueller Studien und Kennzahlen zur Nutzung von Smartphones und Tablet-PCs,

- Darstellung aktueller Marktanteile mobiler Betriebssysteme,
- Darstellung des idealtypischen Produktlebenszyklus mobiler Anwendungen,
- Motivationen zur App-Entwicklung,
- Geschäftsmodelle für den B2B- und B2C-Bereich,
- Sicherheitsmethoden für mobile Applikationen und mobile Geschäftsprozesse,
- Darstellung realer Entwicklungsprojekte.

Das vorliegende Buch gibt den technischen Entwicklungsstand des Sommers 2013 wieder. Einige Aussagen und Analysen der nachfolgenden Kapitel sind durchaus eng mit diesem Stand verknüpft und folgerichtig vor diesem zeitlichen Bezug zu bewerten. Vielfach sind jedoch die in diesem Buch getätigten Aussagen prinzipieller Natur und infolgedessen auch ohne direkten Zeitbezug. Aussagen zur optimalen Gestaltungsmethodik von App-Projekten, zum Management von App-Entwicklungen, zu Methoden der Entwicklung und Qualitätssicherung und vieles mehr behalten auch nach einer Gesetzesänderung oder geänderter Technologie weiterhin ihre Gültigkeit.

---

## Literatur

- Aichele, C., Doleski, O.: Einführung in den Smart Meter Markt. In: Aichele, C., Doleski, O. (Hrsg.) Smart Meter Rollout, S. 3–42. Springer, Berlin (2013)
- Vercales, S., Linhoff-Popien, C.: Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 3–19. Springer, Berlin (2012)

---

# Auf dem Weg zur optimalen mobilen Anwendung

2

Marius Schönberger

*Orientierung und Beweggründe zur Entwicklung mobiler Anwendungen*

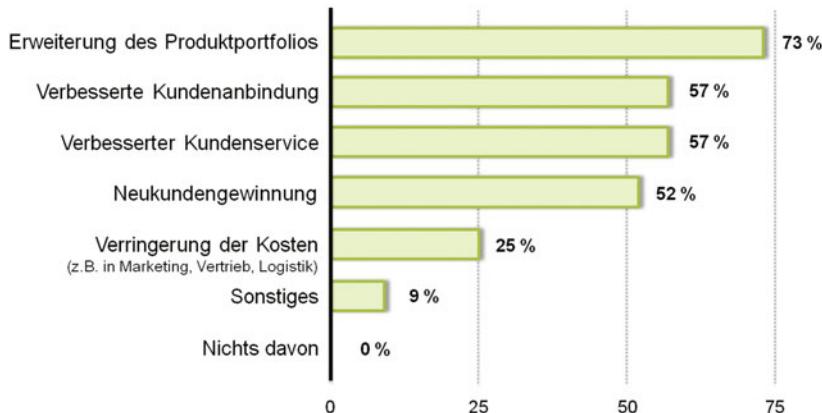
---

## Zusammenfassung

Unzählige Bereiche des fortschrittlichen und modernen Lebens und Arbeitens werden durch den erfolgreichen Einsatz von Informations- und Kommunikationssystemen unterstützt. Heutzutage prägen Begriffe wie „Smartphone“ und „Tablet-PC“ den Lebensalltag vieler Menschen auf der Welt. Vor wenigen Jahren war an diese fast allgegenwärtige Präsenz mobiler Informations- und Telekommunikationstechnologien (ITK) sowie an das Zusammenwachsen von Internet- und Privatanwendungen nicht zu denken. Durch den Ausbau des Funknetzes und dem damit verbesserten Zugang zu Breitbandtechnologien werden mobile Anwendungen und Endgeräte immer mehr in den Mittelpunkt des privaten und unternehmerischen Alltags rücken. Jedoch machen die großen Unterschiede zwischen den Betriebssystemen, Endgeräten und Vermarktungsmöglichkeiten den Weg zur eigenen mobilen Applikation nicht gerade leicht. Der vorliegende Beitrag liefert wichtige Hilfestellungen zu grundlegenden Handlungsscheidungen, die vor der eigentlichen Entwicklung mobiler Anwendungen getroffen werden müssen. Hierfür werden zunächst allgemeine Beweggründe und Anwendungsbereiche mobiler Applikationen genannt und eine Übersicht über aktuelle Studien und Kennzahlen zur Nutzung von Smartphones und Tablet-PCs gegeben. Die Aufführung gegenwärtiger Marktanteile mobiler Betriebssysteme sowie die Nennung diverser Nutzungsgrade ausgewählter mobiler Aktivitäten sollen weitere Beweggründe für die Entwicklung mobiler Applikationen liefern.

---

M. Schönberger (✉)  
Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: marius.schoenberger@fh-kl.de



**Abb. 2.1** Beweggründe für die Entwicklung mobiler Anwendungen. (In Anlehnung an Faßnacht und Ziegler 2011, S. 5)

## 2.1 Beweggründe und Anwendungsbereiche mobiler Applikationen

Die Entwicklung mobiler Applikationen hat sich gegenwärtig zu einem eigenständigen Zweig der Softwareentwicklung etabliert. Mobile Anwendungen tragen bereits heute in vielen großen als auch mittelständischen Unternehmen zur Wertsteigerung bei. Diese Anwendungen sind bislang dahin gehend optimiert, einzelnen Nutzern einen individuellen Mehrwert zu bieten, bspw. in Form von Reiseführern, durch Zugriff auf soziale Netzwerke oder anhand von Spielen zum kurzweiligen Zeitvertreib. Für einige Unternehmen reicht dies jedoch nicht mehr aus, sodass sie mobile Applikationen bereits als Teil des internen und externen Produktpportfolios betrachten. Hierzu erfolgt die Entwicklung mobiler Produkte, die Schaltung mobiler Marketingkampagnen oder die Optimierung innerbetrieblicher Prozesse durch den Einsatz mobiler Dienste.<sup>1</sup> Die Erweiterung des eigenen Portfolios ist nur ein Beweggrund für die Entwicklung und Nutzung mobiler Anwendungen. Dies ergab eine Onlineumfrage des Marktforschers Bitkom, die von Januar bis Februar 2011 unter insgesamt 518 Personen aus Unternehmen der ITK-Branche durchgeführt wurde.<sup>2</sup> Weitere Beweggründe für die Entwicklung mobiler Anwendungen und Dienster werden in Abb. 2.1 aufgezeigt.

Zu den sonstigen Gründen für die Entwicklung mobiler Applikationen gaben die befragten Unternehmen ein modernes und fortschrittliches Firmenimage, effizientere und

<sup>1</sup> Vgl. Zeidler et al. 2012, S. 61.

<sup>2</sup> Vgl. Faßnacht und Ziegler 2011, S. 5.

schnellere Kommunikation sowie ein verbessertes Wissensmanagement an.<sup>3</sup> Weiterhin gaben knapp zwei Drittel der befragten Personen an, in einem produzierenden Unternehmen zu arbeiten, welches mobile Anwendungen entwickelt oder plant, diese zu entwickeln. Weitere 80 % der Befragten geben an, dass ihr Unternehmen mobile Applikationen einsetzt oder plant, diese einzusetzen.<sup>4</sup> Diese Zahlen lassen jedoch keine direkte Schlussfolgerung auf die Nutzung und Anwendungen mobiler Applikationen im gesamten ITK-Markt zu.

Die Entwicklung sowie das Management für mobile Applikationen stellen zusätzliche Anforderungen an Nutzbarkeit, Sinnhaftigkeit sowie Umsetzbarkeit. Eine geeignete Portierung betrieblicher Anwendungen auf mobile Endgeräte erfordert Überlegungen, welche die geeignete mobile Darstellung und Aufbereitung von Daten und Funktionalitäten auf verschiedene Geräteklassen betreffen. Unternehmen müssen sich aus diesen Gründen folgende Auswahl an Kernfragen bezüglich der Entwicklung mobiler Anwendungen stellen:<sup>5</sup>

- Welche Funktionalitäten im Unternehmen sollen mobilisiert werden?
- Welche Prozesse eignen sich für eine Mobilisierung?
- In welchem Umfang müssen Prozesse für eine Mobilisierung verändert werden?
- Welche mobilen Anwendungen bieten einen eindeutigen Mehrwert?

Je nach Einsatzgebiet können mobile Applikationen in Anwendungen zur Unterstützung der Wertschöpfung und Durchführung von Transaktionen sowie in Anwendungen für Endkunden unterschieden werden. Mobile Anwendungen zur Unterstützung der Wertschöpfung zielen hauptsächlich auf effizientere und effektivere Prozesse ab. Das Ziel bei der Durchführung von Transaktionen mittels mobiler Applikationen besteht in der Koordination und Durchführung des Leistungsaustauschs eines Sachgutes oder einer Dienstleistung. Mobile Anwendungen, die als Produkt an Endkunden verkauft werden, sind als ökonomische Güter anzusehen.<sup>6</sup> Abbildung 2.2 zeigt den Zusammenhang zwischen den drei wesentlichen Einsatzgebieten für mobile Anwendungen.

Über mögliche Anwendungsbereiche innerhalb den in Abb. 2.2 aufgezeigten Einsatzgebieten liefert die bereits genannte Bitkom-Umfrage ebenfalls nähere Hinweise. Nach Angaben der Teilnehmer eignen sich demnach mobile Anwendungen insbesondere für Informationsdienste, Social Media, Location Based Services und Spiele.<sup>7</sup> Abbildung 2.3 stellt eine Auswahl der Umfrageergebnisse grafisch dar.

<sup>3</sup> Vgl. Faßnacht und Ziegler 2011, S. 6.

<sup>4</sup> Vgl. Faßnacht und Ziegler 2011, S. 3.

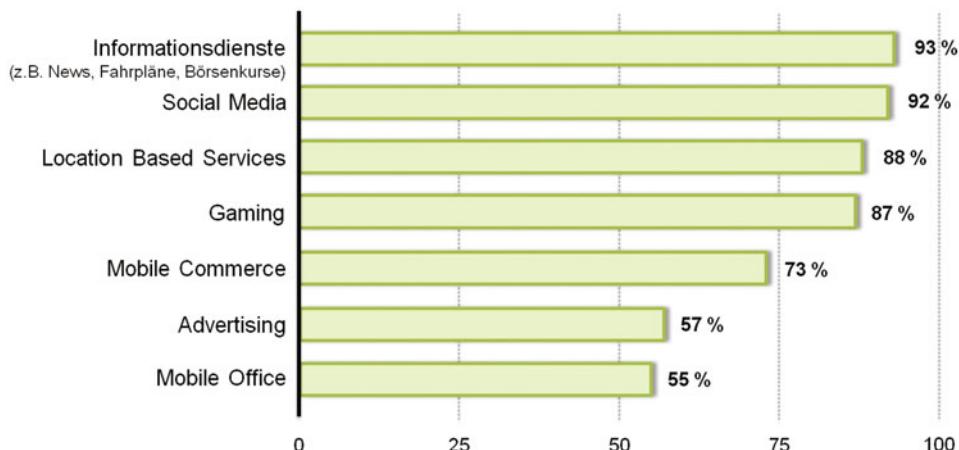
<sup>5</sup> Vgl. Euler et al. 2012 S. 117.

<sup>6</sup> Vgl. Tornack et al. 2011, S. 11.

<sup>7</sup> Vgl. Faßnacht und Ziegler 2011, S. 9.



**Abb. 2.2** Wesentliche Einsatzgebiete mobiler Anwendungen. (In Anlehnung an Hess et al. 2005, S. 9)



**Abb. 2.3** Anwendungsbereiche mobiler Applikationen. (In Anlehnung an Faßnacht und Ziegler 2011, S. 9)

Gegenwärtig haben Unternehmen aufgrund der permanent wachsenden Bandbreite an mobilen Lösungen, die sich nicht nur aufgrund des technischen Fortschritts, sondern auch durch die mittlerweile fast flächendeckende Verfügbarkeit des Internets ergibt, damit begonnen, mobile Applikationen in die verschiedenen Bereiche des betrieblichen Alltags zu integrieren.<sup>8</sup> Hierzu zählen insbesondere Applikationen zur Vertriebsunterstützung, zur Darstellung von Unternehmensdaten in Echtzeit, zur Kontaktaufnahme mit Kunden und Lieferanten oder auch zur Optimierung interner Geschäftsprozesse.<sup>9</sup> In Verbindung mit

<sup>8</sup> Vgl. GS1 Germany 2011, S. 8.

<sup>9</sup> Vgl. Rühl und Schenkel 2012.

mobilen Endgeräten, wie z. B. Smartphones oder Tablet-PCs, können mobile Businesslösungen realisiert werden. Der aus dieser Situation heraus resultierende Begriff „Mobile Business“ ermöglicht für Unternehmen vor allem im Business-to-Business- (B2B) sowie im Business-to-Consumer-Bereich (B2C) neue Potenziale und Möglichkeiten, bspw. den zeit- und ortsunabhängigen Zugriff auf Datenbestände, aber auch die Fernsteuerung von Maschinen oder Produktionsanlagen.<sup>10</sup>

Für ein besseres Verständnis der nachfolgenden Ausführungen soll zunächst der Begriff „Mobile Business“ näher betrachtet werden. Link definiert Mobile Business (M-Business) wie folgt:<sup>11</sup>

- Unter **M-Business** wird der Einsatz mobiler Endgeräte in Planungs-, Abwicklungs- und Interaktionsprozessen von Unternehmen verstanden.

Die Autoren Meier und Stormer geben folgende Begriffsbeschreibung.<sup>12</sup>

- Der Begriff **Mobile Business** . . . umfasst alle Aktivitäten, Prozesse und Applikationen, welche mit mobilen Technologien realisiert werden können.

Eine genaue Betrachtung der bisher gegebenen Eingrenzung zeigt, dass diese oftmals zu kurz greifen und eine eher enge Begriffsbeschreibung darstellen. Eine ausführliche und konkretere Bestimmung des Terminus „Mobile Business“ wird von der SAP AG gegeben. Diese definiert den Begriff folgendermaßen:<sup>13</sup>

- **Mobile Business** stellt einen Erweiterungsprozess von unternehmensrelevanten Computeranwendungen für alle Anwender wie Arbeitnehmer, Partner, Lieferanten und Kunden, unabhängig von Zeit, Ort, Vorhandensein von Netzwerkverbindung – via mobilen Geräten wie Handys, Palm- und Kleinstcomputern, sowie mobilen Industriegeräten dar. Mobile Business ist auch die Fähigkeit, existierende Wirtschaftsprozesse zu verbessern, neue Prozesse aufzustellen und somit letztendlich Umsatz zu generieren, durch eine weit verbreitete Nutzung von Computersystemen und -Ressourcen.

Die Begriffsbestimmung zeigt, dass nicht nur der Einsatz tragbarer Endgeräte innerhalb von Planungs-, Abwicklungs- und Interaktionsprozessen, sondern alle unternehmensrelevanten Computeranwendungen im Vordergrund stehen. Ebenfalls wird die bereits gegebene Eingrenzung von Meier und Stormer konkretisiert. Um die Aktualität der gegebenen Definition zu erhalten, erscheint in Bezug auf die genannten mobilen Endgeräte die Berücksichtigung von Smartphones und Tablet-PCs als sinnvoll. Abbildung 2.4 soll die Ab-

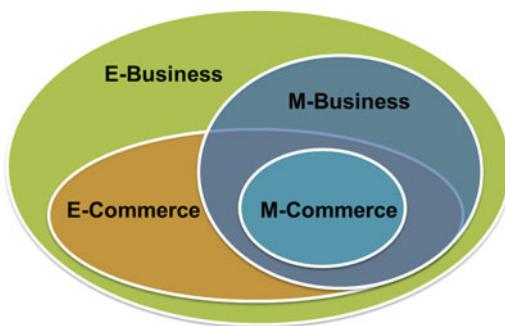
<sup>10</sup> Vgl. GS1 Germany 2011, S. 8.

<sup>11</sup> Vgl. Link 2003, S. 5.

<sup>12</sup> Vgl. Meier und Stormer 2012, S. 247.

<sup>13</sup> Vgl. Frick et al. 2002, S. 200.

**Abb. 2.4** Geschäftsfelder des E-Business. (In Anlehnung an Link 2003, S. 4)



grenzung zu den benachbarten Themengebieten E-Business sowie E- und M-Commerce verdeutlichen.

Von einer detaillierten Betrachtung und Beschreibung der in Abb. 2.4 aufgezeigten weiteren Geschäftsfelder neben dem M-Business wird abgesehen. Nachfolgend wird auf den Umsatz und die Nutzung mobiler Endgeräte, Datendienste und Applikationen eingegangen.

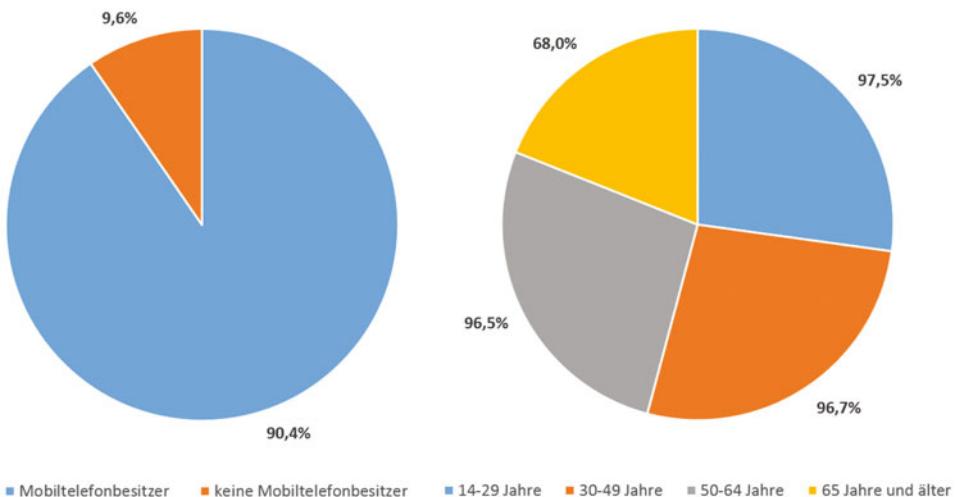
## 2.2 Umsatz und Nutzung mobiler Endgeräte, Datendienste und Applikationen

Gegen Ende des dritten Quartals im Jahr 2011 waren in der Bundesrepublik Deutschland erstmalig über 112 Mio. Mobilfunkanschlüsse geschaltet. Die Verbreitung von Mobilfunkanschlüssen im statistischen Mittel liegt damit bei 137 %. Aus diesen Zahlen resultiert der Tatbestand, dass mehr als jeder dritte Bundesbürger über zwei oder mehrere Mobilfunkanschlüsse verfügt.<sup>14</sup> Zum Ende des ersten Quartals im Jahr 2013 betrug die Anzahl der Mobilfunkanschlüsse der vier großen deutschen Netzbetreiber rund 113 Mio., davon 37 Mio. im Netz der Deutschen Telekom, 32,4 Mio. im Vodafone-Netz, 24 Mio. im E-Plus-Netz und 19,3 Mio. im O2-Netz.<sup>15</sup> Die Anzahl der jeweiligen Mobilfunkanschlüsse bezieht sich auf die Zahl der aktiv geschalteten SIM-Karten.

Weiterhin wird für das Jahr 2013 eine Zunahme des Wettbewerbs im Smartphone- und Tabletmarkt prognostiziert. Fallende Preise lassen mobile Endgeräte für den Massenmarkt erschwinglich werden sowie die Anzahl der mobilen Nutzer rapide ansteigen. Die Relevanz der mobilen Endgeräte wird somit in den Jahren 2014 und 2015 weiter stark wachsen. Aktuelle Studien des Hightechverbandes Bitkom belegen diesen anwachsenden Trend und kündigen zum Ende des Jahres 2013 einen Anstieg des Absatzes an Smartphones in Deutschland um 29 % gegenüber dem Vorjahr auf 28 Mio. Geräte an. Damit wird

<sup>14</sup> Vgl. Bitkom 2012.

<sup>15</sup> Vgl. Bundesnetzagentur 2013.



**Abb. 2.5** Mobiltelefonbesitzer in Deutschland. (In Anlehnung an Bitkom 2013b)

prognostiziert, dass voraussichtlich 4 von 5 verkauften Handys in Deutschland Smartphones sein werden. Der Umsatz würde in diesem Fall auf 8,8 Mrd. € steigen. Damit entfallen insgesamt 96 % des Mobiltelefonmarktes auf Smartphones. Nach den neusten Untersuchungen des European Information Technology Observatory (EITO) soll dabei der Durchschnittspreis von Smartphones weitgehend konstant auf 315 € bleiben.<sup>16</sup>

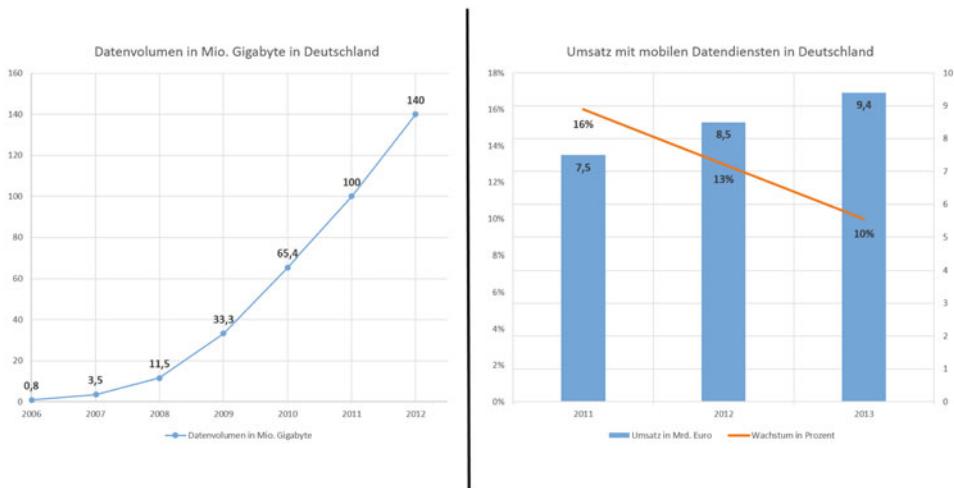
Zu Beginn des Jahres 2012 besaßen laut Statistischem Bundesamt insgesamt 90 % der privaten Haushalte mindestens ein Mobiltelefon. Der Ausstattungsgrad ist damit im Vergleich zum Jahr 2000 mit lediglich 30 % um 60 % gestiegen.<sup>17</sup> Nach Aussagen von Bitkom haben aktuell 90 % aller über 14-Jährigen in Deutschland ein Mobiltelefon, jeder zehnte Deutsche besitzt kein Mobiltelefon (Stand drittes Quartal 2013). Bitkom belegt weiter, dass der Trend vom Mobiltelefon zum Smartphone hin geht. Mittlerweile haben 40 % aller Bundesbürger ab 14 Jahren ein Smartphone, im Jahr 2012 waren es erst 34 %. Unterschiede beim Besitz eines Mobiltelefons zwischen Männern (91 %) und Frauen (90 %) sowie zwischen West- und Ostdeutschland (90 bzw. 91 %) konnten laut Bitkom nicht festgestellt werden.<sup>18</sup> Abbildung 2.5 zeigt den Besitz von Mobiltelefonen in Deutschland.

Gegenwärtig steigt auf dem Computermarkt die Nachfrage nach Tabletcomputern zusehends. Gründe hierfür liegen neben der Fusion von PC und Handy vor allem in der Entwicklung moderner und leistungsfähiger Betriebssysteme. Aktuellen Umfragen zufolge ist der deutsche Markt für Tablet-PCs im Jahr 2012 um ca. 84 % auf 2,1 Mrd. € angewachsen. Vor der Einführung des ersten iPads durch Apple lag der Anteil der Tablet-PCs am

<sup>16</sup> Vgl. Bitkom 2013a.

<sup>17</sup> Vgl. Statistisches Bundesamt 2013.

<sup>18</sup> Vgl. Bitkom 2013b.



**Abb. 2.6** Darstellung von Datenvolumen im Mobilfunk sowie des Umsatzes mit mobilen Datendiensten in Deutschland. (In Anlehnung an Bitkom 2013d, e)

Computermarkt bei knapp 9 %. Im Vergleich zum Jahr 2011 erhöhte sich die Anzahl an verkauften Geräten von 2,1 Mio. auf rund 4,4 Mio. im Jahr 2012. Zum Ende des Jahres 2013 wird ein Absatz von 5 Mio. Tablet-PCs erwartet. Des Weiteren wird eine Verminderung des Durchschnittspreises um 60 € auf 475 € vorhergesagt.<sup>19</sup>

Der bereits erwähnte starke Aufwärtstrend mobiler Endgeräte sorgte ebenfalls für einen starken Anstieg im Bereich der mobilen Internetnutzung. Nach Angaben von Bitkom wird im Jahr 2013 ein voraussichtlicher Anstieg des Umsatzes mobiler Datendienste um 10 % auf 9,4 Mrd. € erfolgen. Neben Smartphones geben hier vor allem Tablet-PCs weitere Impulse. Weltweit wird für das Jahr 2013 ein Zuwachs auf dem Markt für mobile Datendienste um 15 % auf 288 Mrd. € prognostiziert.<sup>20</sup> Die zunehmende Verbreitung von Smartphones sowie Tablet-PCs bringt zusätzliche Herausforderungen für Mobilfunkanbieter mit sich, da parallel zu dem Anstieg der mobilen Internetnutzung auch die Menge an übertragenen Daten und Informationen zunimmt. Nach Berechnungen der Bitkom wurden im Jahr 2012 rund 140 Mio. GB Daten innerhalb der deutschen Mobilfunknetze übertragen.<sup>21</sup> Netzbetreiber sind somit gezwungen, beachtliche Summen in ihre Infrastruktur zu investieren, um dem anfallenden Datenverkehr gerecht zu werden. Abbildung 2.6 zeigt den Anstieg des Datenvolumens im Mobilfunk sowie den Umsatz mit mobilen Datendiensten in Deutschland. Die Angaben des Umsatzes mit mobilen Datendiensten für das Jahr 2013 sind auf Prognosen gestützt.

<sup>19</sup> Vgl. Bitkom 2013c.

<sup>20</sup> Vgl. Bitkom 2013d.

<sup>21</sup> Vgl. Bitkom 2013e.

Mobile Aktivitäten	Großbritannien	Deutschland	Frankreich	Italien	Spanien
Zugriff auf das Internet	70%	57%	56%	60%	74%
Verwendung sozialer Medien	47%	30%	31%	35%	47%
Betrachtung von Videos	26%	21%	17%	24%	31%
Verwendung von mobilen Applikationen	64%	52%	47%	52%	67%
Scannen von QR-Codes	7%	11%	8%	7%	11%
Versenden / Empfangen von Nachrichten	92%	79%	89%	84%	72%

**Abb. 2.7** Nutzungsgrade ausgewählter mobiler Aktivitäten in Europa. (In Anlehnung an Berney 2013, S. 5)

Nach Angaben des US-Marktforschers „comScore“ hatten Ende 2011 mehr als 127,6 Mio. Smartphonenuutzer in den USA und 108 Mio. Nutzer in Europa digitale Medien über mobile Browser und Applikationen konsumiert.<sup>22</sup> Im Jahr 2012 waren Europäer hierbei durchschnittlich knapp 27 h online im Monat. Die Auswertung der am häufigsten durchgeföhrten mobilen Aktivitäten in Europa zeigt diverse Vorlieben der einzelnen Länder sowie unterschiedliche Nutzungsgrade auf.<sup>23</sup> Nachfolgende Abbildung zeigt eine Auswahl dieser mobilen Aktivitäten sowie deren Nutzungsgrade in den betrachteten Ländern (Abb. 2.7).

Der Marktforscher „comScore“ konnte weiterhin im Rahmen der Studie „Digital Future in Focus“ die Anteile der Seitenaufrufe per mobilem Endgerät an einem typischen Wochentag in Deutschland und Europa ermitteln. Die Ergebnisse zeigten, dass in Deutschland überwiegend nachts Smartphones, morgens bis zum Vorabend PCs und abends Tablet-PCs für die Internetnutzung verwendet werden. Die höchste prozentuale Nutzung zeigten Tablet-PCs um 21 Uhr.<sup>24</sup> Aus europäischer Sicht zeigte sich ebenfalls, dass insbesondere mittags PCs für die Internetnutzung verwendet werden. Smartphones und Tablet-PCs werden hauptsächlich abends benutzt.<sup>25</sup>

Betrachtungsschwerpunkt der Studie waren die fünf großen EU-Staaten Deutschland, Großbritannien, Frankreich, Spanien und Italien. Die Bewertung der erfassten Daten wurde anteilmäßig über drei Monate durchgeführt. Als Grundlage für die Studie wurden Daten verwendet, die zum einen aus bereits durchgeföhrten Erhebungen aus dem Jahr 2011 stammten und zum anderen von verschiedenen comScore-Produkten erhoben wurden. Die Datenbasis aus dem Jahr 2011 umfasst insgesamt 5.000 Handybesitzer in Deutschland und Großbritannien sowie jeweils 4.000 in Frankreich, Italien und Spanien. Voraussetzung für die Teilnahme an der Studie war ein Mindestalter von 13 Jahren. Die Auswertung der Ergebnisse erfolgte im letzten Quartal des Jahres 2012.<sup>26</sup>

Die schnelle Entwicklung sowie die große Nachfrage an mobile Applikationen sind nur einige Auslöser für diese hohen Umsatzzahlen. Allein in Deutschland sind in diesem Zusammenhang im Jahr 2012 mehr als 1,7 Mrd. Applikationen auf mobile Endgeräte

<sup>22</sup> Vgl. Rawanick und Aquino 2012, S. 28.

<sup>23</sup> Vgl. Berney 2013, S. 5.

<sup>24</sup> Vgl. Block et al. 2013b.

<sup>25</sup> Vgl. Block et al. 2013a.

<sup>26</sup> Vgl. Block et al. 2013a,b.

geladen worden. Im Vergleich zum Vorjahr entspricht dies einem Zuwachs von 80 %.<sup>27</sup> Der Umsatz von mobilen Anwendungen in Deutschland wurde im Jahr 2012 mit 430 Mio. € angegeben. Ein Jahr zuvor konnte ein Umsatz von 210 Mio. € verzeichnet werden. Die Umsätze von mobilen Applikationen setzen sich neben dem Verkaufspreis weiterhin aus kostenpflichtigen Services oder aus Erlösen durch Werbung innerhalb der Applikation zusammen. Weltweit sind rund 1,8 Mio. mobile Applikationen verfügbar, von denen ein Großteil kostenlos verfügbar ist.<sup>28</sup>

Die Darstellung der aktuellen Situation am Markt zeigt, dass nicht nur Kenntnisse über verschiedene Methoden der Softwareentwicklung vorhanden sein müssen. Vielmehr wird auch ein breites Wissen über gegenwärtige Umsatzzahlen, Nutzunggrade und technologische Neuerungen benötigt. Die Entwicklung und Vermarktung mobiler Anwendungen muss daher im Voraus geplant sowie die mit dem Entwicklungsvorhaben verbundenen Chancen und Risiken möglichst exakt bestimmt werden. Der nachfolgende Orientierungsrahmen gibt daher Handlungsempfehlungen für grundlegende Aufgaben und Tätigkeiten, die vor der Entwicklung mobiler Anwendungen anfallen.

---

## 2.3 Orientierungsrahmen für die Entwicklung mobiler Anwendungen

### 2.3.1 Ausprägungen der B2B- und B2C-Märkte

Wie bereits im vorherigen Kapitel beschrieben erfolgt der Einsatz von mobilen Businesslösungen innerhalb von Unternehmen hauptsächlich in den Bereichen B2B und B2C, wobei im B2B-Bereich der Austausch von Informationen, Waren oder Dienstleistungen zwischen zwei oder mehreren Unternehmen und im B2C-Bereich der Austausch zwischen dem Unternehmen und seinen Kunden im Vordergrund stehen. Innerhalb der Literatur wird in diesem Zusammenhang oftmals der Begriff B2E genannt, der die Beziehung zwischen einem Unternehmen und seinen Angestellten definiert.<sup>29</sup> Diese spezielle Abgrenzung wird im weiteren Verlauf nicht weiter berücksichtigt. Die Märkte B2B und B2C unterscheiden sich in vielen Bereichen, bspw. in der Nachfrage, der Anzahl der Kunden oder dem Vertrieb der erstellten Leistungen.<sup>30</sup> Für das bessere Verständnis des im vorliegenden Kapitel beschriebenen Orientierungsrahmens für die Entwicklung von mobilen Applikationen ist eine Betrachtung von B2B- und B2C-Merkmalen notwendig. Folgende Abbildung gibt einen Überblick über verschiedene Merkmale der B2B- und B2C-Märkte (Abb. 2.8):

---

<sup>27</sup> Vgl. Bitkom 2013f.

<sup>28</sup> Vgl. Bitkom 2013g.

<sup>29</sup> Vgl. GS1 Germany 2011, S. 8.

<sup>30</sup> Vgl. Tornack et al. 2011, S. 14 ff.

Merkmal	Business-to-Business (B2B)	Business-to-Consumer (B2C)
Nachfrager	Industrie-, Dienstleistungsunternehmen und sonstige Organisationen	Private Endverbraucher
Art der Nachfrage	Indirekt	Direkt
Nachfrageentwicklung	Unbeständig	Stetig
Leistungen	- Verbrauchs- und Investitionsgüter - Gebrauchs- und Produktivgüter - Systemtechnologien - Dienstleistungen	- Produkte - Dienstleistungen
Ziel der Leistungserstellung	Investive und / oder produktive Verwendung	Konsumentive Verwendung
Anzahl der Käufer	Gering	Hoch
Käuferkonzentration	Hoch	Gering
Absatzmengen	Hoch	Gering
Kaufverhalten	Rational-ökonomisch	Häufig Emotional
Kaufentscheidungen	Gruppe	Einzel
Beziehung zwischen Käufer und Verkäufer	Eng und langfristig	Eher anonym und kurzfristig
Bedeutung von Maßnahmen zur Kundenbindung	Hoch	Geringer
Komplexität der Dienstleistungen und Produkte	Hoch	Gering
Vertrieb	Direkt	Meist über Zwischenhändler
Produktmerkmale	Produkt in Produktionsprozesse integriert und eher langlebig	Produkte stehen für sich und sind eher kurzlebig

**Abb. 2.8** Merkmale der B2B- und B2C-Märkte. (In Anlehnung an Tornack et al. 2011, S. 16)

Im Zusammenhang mit den gerade gegebenen Ausprägungen der B2B- und B2C-Märkte müssen unterschiedliche Vorgehensweisen bei der Auswahl bzw. der Einführung der Applikation berücksichtigt werden.

### 2.3.2 Vorgehen bei der Auswahl und Einführung mobiler Applikationen

Die Auswahl und Einführung einer mobilen Applikation in einem Unternehmen kommt dem Auswahl- und Einführungsprozess einer sonstigen betrieblichen Software gleich. Daher stehen Unternehmen auch bei der Entscheidung für den Einsatz einer mobilen Applikation oftmals vor der Frage, ob die Applikation mit den eigenen Ressourcen selbst erstellt werden kann oder ob ein externes Softwareunternehmen bereits eine passende mobile Lösung entwickelt hat. Letzteres erweist sich gerade für Unternehmen mit geringem IT-Know-how bzw. knappen zeitlichen und technischen Ressourcen von Vorteil. Des Weiteren muss differenziert werden, ob ein Unternehmen vor der Notwendigkeit einer neuen mobilen Applikation steht und diese für die Abwicklung unternehmensinterner Geschäftsprozesse (B2B) benötigt oder ob ein Unternehmen mobile Anwendungen als eigenständiges Produkt an Endkunden (B2C) verkauft (vgl. Abb. 2.2).

Die Notwendigkeit für den Einsatz einer mobilen Applikation in einem Unternehmen resultiert oftmals aus externen Standpunkten heraus. Die Forderung nach immer kürzer werdenden Prozesslaufzeiten sowie Anforderungen an hohe Qualität und Zuverlässigkeit bzgl. der angebotenen Produkte oder Dienstleistungen und die zunehmende Flexibilisierung von Arbeitsumgebungen und -prozessen spiegeln nur einige treibende Faktoren für

den Bedarf an mobilen Lösungen wider.<sup>31</sup> Im Gegensatz hierzu steht der Vertrieb mobiler Applikationen an Endkunden. Diese Anwendungen basieren einerseits auf neuen und innovativen Ideen, andererseits können sie aber auch das vereinfachte Abbild einer bereits bestehenden PC- oder Internetanwendung darstellen.

Nachfolgend werden zur Auswahl und Einführung mobiler Applikationen zwei unterschiedliche Vorgehensweisen vorgestellt, die je nach Ausgangssituation (B2C oder B2B) und Ausrichtung der Anwendung verschiedene Ausprägungen aufweisen. Die Vorgehensweisen sollen grundlegende Vorentscheidungen und ein strukturiertes Vorgehen bei der Beschaffung oder der Entwicklung mobiler Applikationen aufzeigen.

Das Vorgehen bei der Entwicklung einer mobilen Anwendung für den B2C-Markt wird, wie bereits zuvor beschrieben, üblicherweise durch eine neuartige Idee oder die Adaption bestehender PC- oder Internetanwendungen für mobile Endgeräte ausgelöst. Die daran anknüpfende Marktanalyse soll ermitteln, ob die zuvor generierte Idee umgesetzt werden kann oder ob am Markt bereits eine identische oder ähnliche Anwendung vorhanden ist. Für den Fall, dass am Markt eine vergleichbare mobile Anwendung besteht, müssen Unternehmen, die auf den Vertrieb von mobilen Anwendungen angewiesen sind, ggf. erneut neue Ideen generieren. Ist die mobile Applikation jedoch noch nicht am Markt vorhanden, stehen Unternehmen vor einer Make-or-Buy-Entscheidung. Unternehmen können demnach entweder die Umsetzung der initialen Idee mit den eigenen Ressourcen durchführen oder die Entwicklung der mobilen Applikation einem externen Softwarehaus oder -dienstleister überlassen. Unabhängig von der Entscheidung über Eigenentwicklung oder Fremdbezug muss das Unternehmen eine geeignete Zielplattform für die mobile Anwendung auswählen. Wird für die Umsetzung der Idee ein externes Softwarehaus hinzugenommen, erfolgt üblicherweise in Absprache mit dem jeweiligen Entwicklungsteam die Wahl einer oder mehrerer geeigneter Zielplattformen. Das Vorgehen endet mit dem Beginn des Softwareentwicklungszyklus, auf den an dieser Stelle nicht weiter eingegangen wird.

Im Gegensatz zur Vorgehensweise für den B2C-Markt startet der Ablauf des Einführungsprozesses für den B2B-Markt aufgrund einer Notwendigkeit einer neuen mobilen Applikation. Ausgerichtet auf die vorliegende Problemstellung soll in einem ersten Schritt eine Marktanalyse durchgeführt und hierbei bereits am Markt vorhandene Applikationen identifiziert werden. Zur weiteren Bewertung und Analyse der erhobenen Applikationen sollte das Unternehmen die Anbieter kontaktieren und ggf. mögliche Konditionen bei der Einführung der Anwendung besprechen. Kann keine geeignete Anwendung aufgrund der Marktanalyse identifiziert werden, steht das Unternehmen ebenfalls vor einer Make-or-Buy-Entscheidung. Der Ablauf entspricht ab diesem Zeitpunkt dem Vorgehen bei der Entwicklung einer mobilen Anwendung für den B2C-Markt.

Nachfolgend werden die einzelnen Punkte der beiden Vorgehensmodelle nochmals aufgegriffen und erläutert.

---

<sup>31</sup> Vgl. Linhoff-Popien und Vercales 2012, S. 5.

### 2.3.2.1 Durchführung einer Marktanalyse im B2B- und B2C-Markt

Die Durchführung einer Marktanalyse dient einerseits zur Orientierung auf dem B2B- oder B2C-Markt für mobile Applikationen und stellt andererseits Informationen bereit, die über das weitere Vorgehen bei der Auswahl oder Entwicklung einer mobilen Anwendung entscheiden können. Eine besonders einfache Methode, die schnell zu guten Ergebnissen führt, bildet die Analyse von Online-Stores, auf denen mobile Anwendungen vertrieben werden. Zu den bekanntesten Online-Stores zählen der App-Store von Apple, der Play-Store von Google und der Windows Phone Store von Microsoft. Aktuellen Studien zufolge liegt der Play-Store mit einer Million Anwendungen<sup>32</sup> knapp vor dem App-Store mit derzeit 900.000 Anwendungen<sup>33</sup>. Der Windows Phone Store liegt mit 170.000 Applikationen damit an dritter Stelle.<sup>34</sup> Da die Suche nach mobilen Anwendungen innerhalb der Online-Stores ohne Registrierung möglich ist, können Unternehmen eine schnelle und kostensparende Analyse durchführen und erhalten auf diese Weise grundlegende Informationen, wie z. B. eine Beschreibung über den Funktionsumfang, Name des Anbieters oder den Preis der Anwendung. Aufgrund der hohen Anzahl an mobilen Applikationen innerhalb der Online-Stores ist die Suche nach einer bestimmten Anwendung jedoch sehr zeitintensiv. Weiterhin können die Funktionen der Anwendungen ohne vorherige Registrierung nicht getestet werden.

Die Suche über Onlineportale stellt eine weitere Möglichkeit dar, bestehende mobile Anwendungen zu identifizieren und Informationen über deren Leistungsumfang zu erheben. Die Betreiber solcher Onlineportale haben es sich zur Aufgabe gemacht, neue mobile Anwendungen zu testen und zu bewerten. Die Ergebnisse der Analyse werden daran anschließend über einen Onlinekatalog gelistet und der Allgemeinheit zur Verfügung gestellt. Einige Portale erweitern diese Beiträge zusätzlich um Screenshots und weitere anwendungsbezogene Informationen. Ebenfalls wie bei den App-Stores können registrierte Benutzer Anmerkungen zu denen im Katalog gelisteten Anwendungen abgeben. Im Gegensatz zu den App-Stores können aufgrund der Testberichte bessere Entscheidungen über den Funktionsumfang und möglichen Einsatz der mobilen Applikation getroffen werden. Ein Beispiel für ein solches Onlineportal bildet „AppBrain“, welches hauptsächlich Testberichte und Bewertungen über Android-Anwendungen listet.<sup>35</sup>

Das Auffinden einer bereits vorhandenen Applikation auf dem B2C-Markt bedeutet nicht zwingend, dass durch deren Leistungsumfang ein Bedarf nach einer mobilen Anwendung vollumfänglich abgedeckt wird. Erst durch die genaue Betrachtung des Konkurrenzprodukts kann eine Aussage über dessen Erfolg am Markt getroffen werden. Bevor somit die eigene Idee verworfen wird, bietet sich die Analyse der Konkurrenzprodukte hinsichtlich der Bedienung, des Designs, des Erlösmodells sowie der Rezensionen der Nutzer an. Insbesondere Fehler bei der Konkurrenz oder der Wunsch einzelner Nutzer nach

<sup>32</sup> Vgl. Dietl 2013a.

<sup>33</sup> Vgl. Dietl 2013b.

<sup>34</sup> Vgl. Microsoft Corp. 2013.

<sup>35</sup> [www.appbrain.com](http://www.appbrain.com).

einer Erweiterung oder Änderung des Funktionsumfangs zeigen, dass evtl. ein Bedarf nach einer gleichartigen, aber verbesserten mobilen Applikation besteht. Werden keine vergleichbaren Applikationen am Markt identifiziert, gilt es, diese Marktsituation genauso zu analysieren, wie es bei einem Überangebot an ähnlichen Anwendungen notwendig ist.<sup>36</sup>

Im nachfolgenden Abschnitt wird auf die Make-or-Buy-Entscheidung bei der Einführung von mobilen Anwendungen in Unternehmen eingegangen.

### 2.3.2.2 Make-or-Buy-Entscheidung

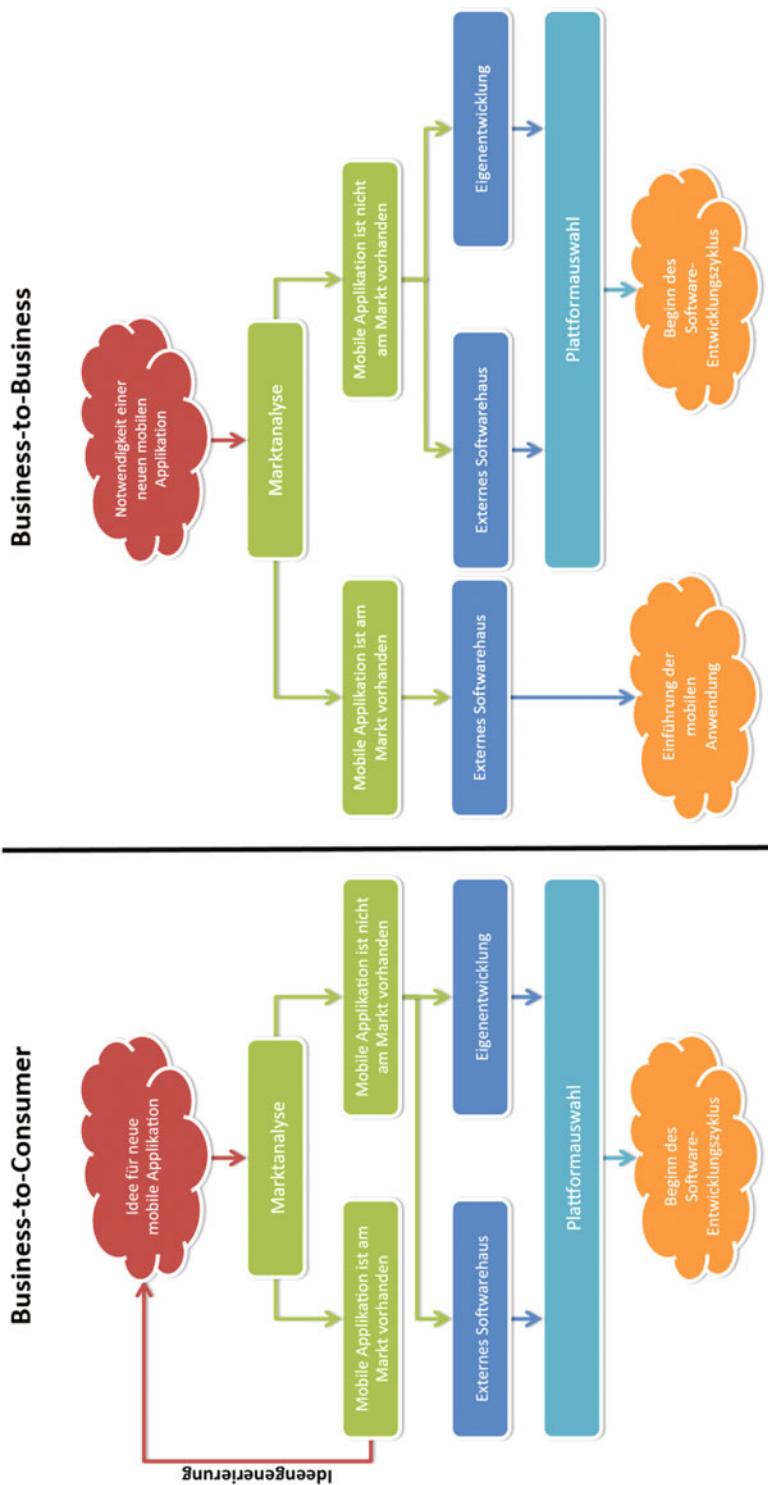
In Bezug auf Abb. 2.9 wird ersichtlich, dass Unternehmen nach der Durchführung einer Marktanalyse vor der Entscheidung der Eigenentwicklung oder des Fremdbezugs einer mobilen Anwendung stehen. Unternehmen stehen somit vor der Herausforderung, die unterschiedlichen Aspekte und Konsequenzen durch die Eigenentwicklung oder den Fremdbezug zu bewerten. Ein wichtiges Argument ist hierbei die Frage nach der Wirtschaftlichkeit sowie nach strategischen und qualitativen Aspekten der jeweiligen Alternativen. Die Frage nach dem Make-or-Buy einer mobilen Anwendung kann somit nicht generell beantwortet werden. Folgende Abbildung soll unterschiedliche Strategieansätze bei der Entwicklung und dem Betrieb von allgemeinen IT-Produkten und -Leistungen verdeutlichen.

Wie aus Abb. 2.10 ersichtlich bietet sich bei geringer Bedeutung der IT-Lösungsansätze für die Unternehmensziele und Kernprozesse eine „Buy-Strategie“ an. Oftmals handelt es sich hierbei um den Kauf von Standardanwendungen zur Unterstützung der Buchhaltung oder der Lagerwirtschaft. Der Einführung einer neuen IT-Lösung wird insbesondere dann eine hohe Bedeutung zugewiesen, wenn wichtige Kernprozesse des Unternehmens betroffen sind und es sich vorwiegend um unternehmensindividuelle Aufgaben- und Tätigkeitsbereiche handelt. Beispiele für solche IT-Lösungen sind Informationssysteme für die Produktentwicklung oder spezifische Kundeninformationssysteme. Für den Fall, dass interne als auch externe Ressourcen koordiniert werden müssen, ist abzuwägen, welche Strategie angewendet werden soll. Ein Beispiel ist die Einführung von Schnittstellen zwischen unterschiedlichen Anwendungssystemen.<sup>37</sup>

Die Eigenentwicklung sowie der Fremdbezug von IT-Lösungen weisen einige Vorteile und Nachteile auf, die Unternehmen dazu veranlassen, Make-or-Buy-Entscheidungen sorgfältig vorzubereiten (vgl. Abb. 2.11). Die Bewahrung des eigenen IT-Know-how, die Sicherstellung des Datenschutzes und der Datensicherheit sowie fehlende Lizenzkosten stellen nur einige Vorteile der Eigenentwicklung von Softwarelösungen dar. Nachteile der Eigenentwicklung sind neben der hohen Belastung der IT-Abteilung, lange Projektlaufzeiten sowie evtl. anfallende, zusätzliche Schulungskosten für die Mitarbeiter. Die bessere Steuerung von IT-Kosten, die Abwälzung der Risiken auf den externen Dienstleister und die Entlastung des eigenen Personals sprechen für den Einsatz externer Softwareunternehmen. Die Abhängigkeit von diesen Unternehmen, geringe Kontrollmöglichkeiten

<sup>36</sup> Vgl. Knüppfer et al. 2013, S. 10.

<sup>37</sup> Vgl. Gadatsch 2012, S. 145.



**Abb. 2.9** Vorgehensweisen bei der Auswahl oder Entwicklung mobiler Applikationen für den B2B- und den B2C-Markt



**Abb. 2.10** Make-or-Buy-Strategien. (In Anlehnung an Gadatsch 2012, S. 145)

	Make	Buy
Vorteile	<ul style="list-style-type: none"> <li>+ IT-Know-how wird im Unternehmen gehalten</li> <li>+ Keine Abhängigkeit von Fremdfirmen</li> <li>+ Datenschutz und Datensicherheit</li> <li>+ Keine Lizenzkosten</li> </ul>	<ul style="list-style-type: none"> <li>+ Bessere Steuerbarkeit von IT-Kosten</li> <li>+ Abwälzung von Risiken auf den externen Dienstleister</li> <li>+ Entlastung des Personalwesens</li> <li>+ Konzentrationsmöglichkeit auf das Kerngeschäft des Unternehmens</li> </ul>
Nachteile	<ul style="list-style-type: none"> <li>- Hohe Projektlaufzeit</li> <li>- Nachträgliche Anpassungen können zusätzliche Kosten hervorrufen</li> <li>- Hohe Belastung der eigenen IT-Abteilung</li> <li>- Evtl. sind zusätzliche Schulungsmaßnahmen notwendig</li> </ul>	<ul style="list-style-type: none"> <li>- Abhängigkeit von Fremdfirmen</li> <li>- Gefahr des Missbrauchs schutzwürdiger betrieblicher Daten</li> <li>- Verzicht auf eigene IT-Kompetenz</li> <li>- Geringe Kontrollmöglichkeiten</li> <li>- Lizenzkosten</li> </ul>

**Abb. 2.11** Vor- und Nachteile bei der Eigenentwicklung oder dem Fremdbezug von IT. (Vgl. Stahlknecht und Hasenkamp 2005, S. 451 f.)

und die Gefahr des Missbrauchs von internen Unternehmensdaten bilden Nachteile beim Fremdbezug von Software.

### 2.3.2.3 Auswahl mobiler Betriebssysteme

Aus der Entscheidung für die Eigenentwicklung oder den Fremdbezug einer mobilen Anwendung resultiert die Frage, für welches mobile Betriebssystem die Applikation entwickelt werden soll. Während bei den Anfängen der mobilen Anwendungsentwicklung hauptsächlich ein bestimmtes Betriebssystem fokussiert wurde, werden aktuelle Applikationen

überwiegend plattformübergreifend entwickelt. Die Herausforderung bei der Entwicklung mobiler Anwendungen besteht somit in der Heterogenität der mobilen Betriebssysteme. Zur bestmöglichen Vermarktung der Anwendung und für die Sicherstellung der Lauffähigkeit der Applikation auf verschiedenen Endgeräten ist daher die Kenntnis über unterschiedliche Betriebssysteme, Entwicklungsumgebungen und Programmiersprachen notwendig.<sup>38</sup> Fehlt dieses Wissen in den Unternehmen, bietet sich generell die Vergabe des Entwicklungsvorhabens an ein externes Softwarehaus an.

Wie aus Abb. 2.9 ersichtlich muss vor dem Beginn des Softwareentwicklungszyklus, aufgrund einer am B2B- oder B2C-Markt fehlenden Applikation, die Auswahl einer mobilen Plattform getroffen werden. Für den Fall, dass auf dem B2B-Markt für die vorliegende Problemstellung eine entsprechende Anwendung identifiziert werden konnte, entfällt der Auswahlprozess und die Einführung der mobilen Anwendung in das Unternehmen kann durchgeführt werden. Hierbei müssen sich Unternehmen oftmals an den Anforderungen der mobilen Applikation und den Bedingungen des externen Softwareunternehmens ausrichten, wodurch ggf. weitere Kosten bei der Einführung entstehen können. Ebenfalls ist eine nachträgliche Anpassung an die Unternehmens-IT oder an unternehmensinterne Abläufe und Geschäftsprozesse nicht immer realisierbar.

Abbildung 2.12 zeigt die Marktanteile mobiler Betriebssysteme in Deutschland und auf dem EU5-Markt, der sich aus Großbritannien, Deutschland, Frankreich, Italien und Spanien zusammensetzt. Die dargestellten Verkaufsdaten stellen die Situation am Markt für mobile Betriebssysteme für drei Monate, Juni bis August 2013, dar und wurden von dem englischen Marktforscher „Kantar Worldpanel ComTech“ erhoben. Die Zahlen zeigen, dass Googles Betriebssystem Android aktuell mit jeweils 78,7 % auf dem deutschen Markt und mit 70,1 % auf dem europäischen Markt die meisten Marktanteile aufweisen kann. An zweiter Stelle befindet sich das iOS-Betriebssystem von Apple mit 9,5 % auf dem deutschen und 16,1 % auf dem europäischen Markt. Drittgrößtes Betriebssystem ist Microsoft Windows Phone 8 mit 8,8 % in Deutschland und 9,2 % in Europa. Android nimmt somit eine marktbeherrschende Position ein und kann auf dem europäischen Markt weitere Marktanteile gewinnen. Apple kann auf dem europäischen Markt ebenfalls einen leichten Zuwachs aufweisen, verliert aber auf dem deutschen Markt 1,6 % der Anteile im Vergleich zum Vorjahr. Microsoft kann eines der besten Ergebnisse aufweisen und kann in Deutschland einen Anstieg der Marktanteile um fünf Prozent und in Europa um 4,2 % verzeichnen.<sup>39</sup>

Ältere Betriebssysteme wie das BlackBerry Operation System (OS) von RIM oder das Palm OS verlieren immer mehr Anteile am Markt. Dies ist auf die rapide Weiterentwicklung moderner Betriebssysteme, wie bspw. Android und iOS, zurückzuführen. Weitere Gründe sind u. a. mangelnde Personalisierungsmöglichkeiten sowie eingeschränkte Anbindungen an mobile Dienste. Die Festigung sowie der Ausbau von Marktanteilen bereits bestehender Hersteller von Betriebssystemen macht es zudem für Neueinsteiger schwer, Marktanteile zu gewinnen.

<sup>38</sup> Vgl. Gerlicher 2012, S. 161.

<sup>39</sup> Vgl. Sunnebo 2013.

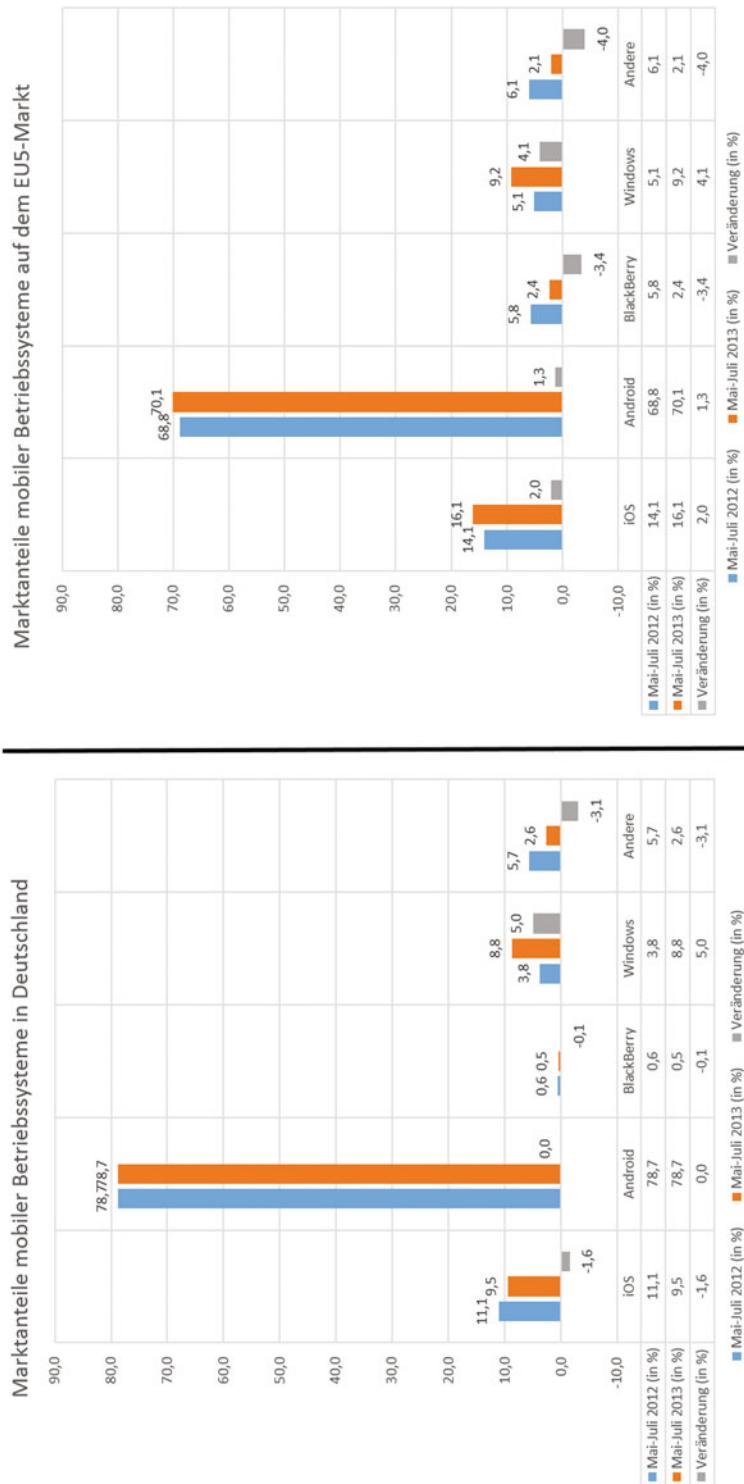


Abb. 2.12 Marktanteile mobiler Betriebssysteme in Deutschland und auf dem EU5-Markt. (Vgl. Sunnebo 2013)

Unternehmen können solche Marktstatistiken einerseits für eine generelle Orientierung am Markt und andererseits als Entscheidungshilfen für das Entwicklungsvorhaben zu Hilfe nehmen. Für die endgültige Auswahl einer Plattform müssen jedoch zusätzliche interne und externe Faktoren berücksichtigt werden, bspw. Präferenzen der späteren Nutzer der Applikation bzgl. des mobilen Betriebssystems sowie die Betrachtung der firmeninternen Endgeräte hinsichtlich der darauf installierten Betriebssysteme. Des Weiteren werden durch das mobile Betriebssystem der Vertrieb der mobilen Applikation über fest vorgeschriebene App-Stores oder Online-Marktplätze festgelegt (siehe Abschn. 2.3.2.1). Dadurch müssen bei der Auswahl eines Betriebssystems auch die damit verbundenen Vertriebswege und Konditionen überprüft werden.

### 2.3.3 Honorarkosten bei der Entwicklung mobiler Applikationen

Zur besseren Orientierung bei der Entwicklung mobiler Applikationen soll abschließend der Kostenaspekt betrachtet werden. Marktforscher des „HighText Verlags“ haben hierzu im Jahr 2013 eine Studie über die anfallenden Honorare bei der Entwicklung mobiler Anwendungen in Deutschland veröffentlicht.<sup>40</sup> Zur besseren Einstufung der Honorarkosten erfolgte eine Einteilung in drei unterschiedliche Komplexitätsstufen: einfache, durchschnittliche und komplexe Applikationen. Einfache Applikationen zeichnen sich durch wenig Logik, keinen Backendzugriff, minimalistisches Design und die überwiegende Verwendung von Standardkomponenten aus. Durchschnittliche mobile Anwendungen verwenden ebenfalls kein eigenes Backend, können aber auf fremde Backends zugreifen. Im Vergleich zu einfachen Anwendungen können sie mehrere Bildschirme und vereinzelte Konfigurationsmöglichkeiten aufweisen. Die Entwicklung einer komplexen Anwendung beinhaltet neben der Konzeption und Umsetzung eines eigenen Backends, die Einbindung von Menüs und umfangreicheren Konfigurationsmöglichkeiten sowie die Anpassung der Layouts nach eigenen Wünschen. Zusätzlich sind Funktionen und Komponenten zum Datenschutz und zur Datensicherheit notwendig. Nachfolgende Abbildung zeigt die erhöhen Durchschnittspreise für die mobile Anwendungsentwicklung aus dem Jahr 2011 im Vergleich zu einer aktuellen Studie aus dem Jahr 2013, die ebenfalls auf einer Umfrage des „HighText Verlags“ basiert (siehe Abb. 2.13).<sup>41</sup>

Während im Jahr 2011 die Preise für die Entwicklung mobiler Anwendungen in Deutschland unterschiedlich stark ausgeprägt waren, haben sich die Entwicklungskosten für Android- und iOS-Anwendungen im Jahr 2013 weitestgehend angenähert. Für Nischenbetriebssysteme, wie z. B. das Betriebssystem Symbian von Nokia, können die Kosten jedoch bis zu 45 % höher sein. Für die Entwicklung und Festlegung von Kosten spielen mehrere Faktoren eine Rolle, bspw. das verwendete Betriebssystem oder der Umfang der Anwendung. Weiterhin wurden durch die Studien unterschiedliche Preise identifiziert,

<sup>40</sup> Vgl. Rauscher 2013.

<sup>41</sup> Vgl. Rauscher 2013.

Komplexitätsstufe	Jahr	Minimum	Durchschnitt	Maximum
Einfache Anwendung	2011	760,00 €	16.500,00 €	97.000,00 €
	2013	601,00 €	7.754,00 €	78.790,00 €
Durchschnittliche Anwendung	2011	2.450,00 €	23.000,00 €	105.000,00 €
	2013	809,00 €	13.397,00 €	65.933,00 €
Komplexe Anwendung	2011	6.000,00 €	79.000,00 €	520.000,00 €
	2013	802,00 €	48.804,00 €	382.215,00 €

**Abb. 2.13** Durchschnittspreise für die mobile Anwendungsentwicklung in Deutschland. (In Anlehnung an Graf 2011; Rauscher 2013)

wenn eine Agentur, ein kleineres Softwarehaus oder freiberufliche Entwickler mit der Entwicklung beauftragt wurden. Grundsätzlich lassen sich zwei Berechnungsmodelle unterscheiden: der Pauschalpreis und der Preis pro Funktion. Die Studien haben ergeben, dass ein Großteil der in Deutschland entwickelten Anwendungen nach dem Preis pro Funktion kalkuliert wurde.<sup>42</sup>

## 2.4 Resümee

Unternehmen, die sich auf die Herstellung und Verbreitung mobiler Applikationen spezialisiert haben, müssen, um langfristig auf dem B2B- oder B2C-Markt bestehen zu können, plattformübergreifende Programme mit Alleinstellungsmerkmal entwerfen. Aus dem Unterhaltungssektor kann als Beispiel das Spiel „Angry Birds“ aufgeführt werden. Das Spiel wurde ursprünglich für mobile Apple-Endgeräte entwickelt und ist seit dem Erscheinungsdatum im Jahr 2009 nun für alle mobilen Betriebssysteme zum Download vorhanden. Insgesamt kann das Spiel des Herstellers „Rovio“ aktuell über eine Milliarde Downloads verzeichnen.<sup>43</sup> Voraussetzung für diese Erfolge im Mobile Business sind u. a. moderne Softwareplattformen, gesicherte Übertragungskapazitäten sowie der garantierte Schutz der Daten.<sup>44</sup>

## Literatur

- Berney, P.: Mobile Barometer Q2 2013. ComScore Inc., Reston (2013)  
 Block, B., McCarthy, C., Mohamud, A.: Europe digital future in focus 2013. Key Insights from 2012 and What They Mean for the Coming Year. ComScore Inc., Reston (2013a)

<sup>42</sup> Vgl. Graf 2011; Rauscher 2013.

<sup>43</sup> Vgl. Focus Online 2012.

<sup>44</sup> Vgl. Scheer 2012, S. 9.

- Block, B., McCarthy, C., Mohamud, A.: Future in Focus Digitales Deutschland 2013. Erkenntnisse aus 2012 und was diese für das kommende Jahr bedeuten. ComScore Inc., Reston (2013b)
- Bundesnetzagentur: Anzahl der Mobilfunkanschlüsse in Deutschland nach Netzbetreiber vom 1. Quartal 2008 bis zum 4. Quartal 2012. <http://de.statista.com/statistik/daten/studie/183182/umfrage/anzahl-der-mobilfunkanschluesse-in-deutschland-nach-netzbetreiber> (März 2013). Zugriffen 11. April 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2012): Neuer Rekord bei Mobilfunkanschlüssen in Deutschland, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_71315.aspx](http://www.bitkom.org/de/markt_statistik/64046_71315.aspx) (2012). Zugriffen 11. April 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013a): Smartphones sorgen für 96 % des Handy-Umsatzes. Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_75052.aspx](http://www.bitkom.org/de/markt_statistik/64046_75052.aspx) (2013). Zugriffen 11. April 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013b): 63 Millionen Handy-Besitzer in Deutschland, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_77178.aspx](http://www.bitkom.org/de/markt_statistik/64046_77178.aspx) (2013). Zugriffen 2. Okt. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013c): Tablet-Verkäufe übertreffen Erwartungen, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_75052.aspx](http://www.bitkom.org/de/markt_statistik/64046_75052.aspx) (2013). Zugriffen 11 April 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013d): Umsätze mit mobilen Datendiensten steigen stark an, Berlin. [http://www.bitkom.org/de/presse/30739\\_75060.aspx](http://www.bitkom.org/de/presse/30739_75060.aspx) (2013). Zugriffen 2. Okt. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013e): Schub fürs mobile Breitband, Berlin. [http://www.bitkom.org/de/presse/8477\\_76810.aspx](http://www.bitkom.org/de/presse/8477_76810.aspx) (2013). Zugriffen 2. Okt. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013f): Rekord bei App-Downloads, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64026\\_75628.aspx](http://www.bitkom.org/de/markt_statistik/64026_75628.aspx) (2013). Zugriffen 2. Okt. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom 2013g): Zahlungsbereitschaft für Apps steigt, Berlin. [http://www.bitkom.org/de/themen/54866\\_74940.aspx](http://www.bitkom.org/de/themen/54866_74940.aspx) (2013). Zugriffen 2. Okt. 2013
- Dietl, W.: Apple bleibt Mobile Advertising-Spitzenreiter. <http://mobile-studien.de/2013/07/apple-bleibt-mobile-advertising-spitzenreiter> (Juli 2013a). Zugriffen 23. Sept. 2013
- Dietl, W.: Apple: Mit 900.000 Apps noch immer die Nr. 1. <http://mobile-studien.de/2013/06/apple-mit-900-000-apps-noch-immer-die-nr-1> (Juni 2013b). Zugriffen 23. Sept. 2013
- Euler, M., Hacke, M., Hatherz, C., Steiner, S., Vercales, S.: Herausforderungen bei der Mobilisierung von Business Applikationen und erste Lösungsansätze. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 107–125. Springer, Berlin (2012)
- Faßnacht, C., Ziegler, S.: Mobile Anwendungen in der ITK-Branche. Umfrage-Ergebnisse. Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V., Berlin (2011)
- Focus Online: Angry Birds knackt Milliardenmarke. [http://www.focus.de/finanzen/news/netzsplitter-angry-birds-knackt-milliardenmarke\\_aid\\_751774.html](http://www.focus.de/finanzen/news/netzsplitter-angry-birds-knackt-milliardenmarke_aid_751774.html) (2012). Zugriffen 2. Okt. 2013
- Frick, O., Hofmann, M., Kramer, A., Netzel, A.: Mobile Business bei SAP. In: Teichmann, R., Lehner, F. (Hrsg.) Mobile Commerce. Strategien, Geschäftsmodelle, Fallstudien. Springer, Berlin (2002)
- Gadatsch, A.: IT-Controlling. Praxiswissen für IT-Controller und Chief-Information-Officer. Springer, Berlin (2012)

- Gerlicher, A. R. S.: Die Grenzen des Browsers durchbrechen. Hybride Anwendungsentwicklung für mobile Endgeräte. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 161–177. Springer, Berlin (2012)
- Graf, J.: Studie: Was App-Entwicklung in Deutschland wirklich kostet. <http://www.ibusiness.de/aktuell/db/709614jg.html> (2011). Zugegriffen 1. Okt. 2013
- GS1 Germany GmbH: Mobile Business. Neue Geschäftsmöglichkeiten für kleine und mittlere Unternehmen. [http://www.prozeus.de/imperia/md/content/prozeus/broschueren/prozeus\\_broschuere\\_mobilebusiness\\_rz\\_web.pdf](http://www.prozeus.de/imperia/md/content/prozeus/broschueren/prozeus_broschuere_mobilebusiness_rz_web.pdf) (2011). Zugegriffen 20. Aug. 2013
- Hess, T., Figge, S., Hanekop, H., Hochstatter, I., Hogrefe, D., Kaspar, C., Rauscher, B., Richter, M., Riedel, A., Zibull, M.: Technische Möglichkeiten und Akzeptanz mobiler Anwendungen – Eine interdisziplinäre Betrachtung. *Wirtschaftsinformatik* 47(1), 6–16 (2005)
- Knüppfer, W., Fritsch, M., Matthes, A.: Von der Idee zur eigenen App. Ein praxisorientierter Leitfaden für Unternehmer mit Checkliste, eBusiness-Lotse Metropolregion Nürnberg. [http://www.nik-nbg.de/fileadmin/redaktion/Hinterlegte\\_Dokumente\\_Homepage/Leitfaden\\_-\\_Von\\_der\\_Idee\\_zur\\_eigenen\\_App.pdf](http://www.nik-nbg.de/fileadmin/redaktion/Hinterlegte_Dokumente_Homepage/Leitfaden_-_Von_der_Idee_zur_eigenen_App.pdf) (Juni 2013). Zugegriffen 25. Sept. 2013
- Kuassi, L., Bischel, M.: Anwendungssicht mobiler Geschäftsanwendungen. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 125–147. Springer, Berlin (2012)
- Linhoff-Popien, C., Vercales, S.: Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 3–16. Springer, Berlin (2012)
- Link, J.: Mobile Commerce. Gewinnpotenziale einer stillen Revolution. Springer, Berlin (2003)
- Meier, A., Stormer, H.: eBusiness & eCommerce. Management der digitalen Wertschöpfungskette, 3. Aufl. Springer, Berlin (2012)
- Microsoft Corp.: Microsoft by the numbers. A collection of visual statistics about microsoft products and services. <http://www.microsoft.com/en-us/news/bythenumbers/index.html> (2013). Zugegriffen 23. Sept. 2013
- Rauscher, H. S.: Studie: Das kostet die App-Entwicklung in Deutschland 2013. <http://www.ibusiness.de/aktuell/db/857182hr.html> (2013). Zugegriffen 1. Okt. 2013
- Rawanick, S., Aquino, C.: Mobile Future in Focus. Key Insights from 2011 and What they Mean for the Coming Year. ComScore Inc., Reston (2012)
- Rühl, C., Schenkel, T.: Best Practices für die Entwicklung mobiler Unternehmens-Apps. <http://www.heise.de/developer/artikel/Best-Practices-fuer-die-Entwicklung-mobiler-Unternehmens-Apps-1627012.html> (2012). Zugegriffen 20. Aug. 2013
- Scheer, A.-W. (Hrsg.): Mobile applications – fit for business. IM – Fachzeitschrift für Information Management und Consulting. 27(1), 8–9 (2012)
- Stahlknecht, P., Hasenkamp, U.: Einführung in die Wirtschaftsinformatik, 11. Aufl. Springer, Berlin (2005)
- Statistisches Bundesamt: 90 % der Privathaushalte haben mindestens ein Handy. [https://www.destatis.de/DE/PresseService/Presse/Pressemitteilungen/zdw/2013/PD13\\_025\\_p002pdf.pdf?\\_\\_blob=publicationFile](https://www.destatis.de/DE/PresseService/Presse/Pressemitteilungen/zdw/2013/PD13_025_p002pdf.pdf?__blob=publicationFile) (2013). Zugegriffen 2. Okt. 2013
- Sunnebo, D.: Windows phone nears double digit share across Europe. <http://www.kantarworldpanel.com/global/News/news-articles/Windows-Phone-nears-double-digit-share-across-Europe> (2013). Zugegriffen 1. Okt. 2013
- Tornack, C., Christmann, S., Hagenhoff, S.: Tendenzielle Unterschiede zwischen B2B und B2C-Anwendungen für mobile Endgeräte, Arbeitsbericht der Professur für Anwendungssysteme und E-Business. Arbeitsbericht Nr. 3/2011. Göttingen (2011)
- Zeidler, A., Eckl, R., Trumler, W., Marquart, F.: Mobile Apps für industrielle Anwendungen am Beispiel von Siemens. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 61–80. Springer, Berlin (2012)

---

# Strategien und Geschäftsmodelle für mobile Applikationen

3

Christian Aichele

*Die Vorgehensweise zur Etablierung mobiler Applikationen im B2B und B2C*

---

## Zusammenfassung

Erster Schritt in der Generierung mobiler Applikationen ist die Definition der Strategie. Aufbauend auf der Strategie wird das Geschäftsmodell konzipiert und die Projektumsetzung geplant.

---

### 3.1 Die Generierung einer Strategie für mobile Applikationen

Eine Strategie für die Entwicklung und Marktabtiblierung mobiler Applikationen dient zur Positionierung des eigenen Unternehmens in Bezug auf Umfang und Funktionalität der mobilen Applikation und die zu erreichenden Ziele mit der Einführung der Anwendung. Die Initialzündung zur Entwicklung einer App-Strategie kann dabei extrinsischer oder intrinsischer Natur sein. Der extrinsische Antrieb kann durch eine eigene oder auch fremde Marktevaluation und Marktnachfrage durch Kunden oder verbundene Unternehmen erfolgen. Intrinsische Anregungen kommen von einzelnen Mitarbeitern oder Gruppen von Mitarbeitern und entstehen zumeist auch durch Beobachtungen, Evaluationen oder Interaktion und Kommunikation.

---

C. Aichele (✉)

Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern, Zweibrücken, Deutschland  
E-Mail: christian.aichele@fh-kl.de

- Eine **App-Strategie** ist eine konzipierte Vorgehensweise zur Entwicklung und Etablierung mobiler Applikationen in Bezug auf definierte Ziele.

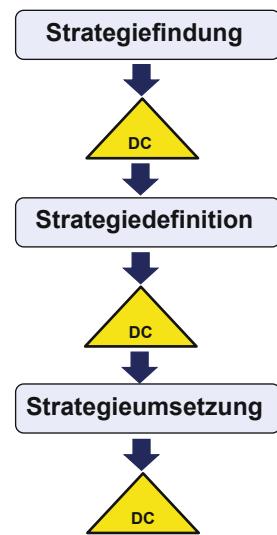
Auch die spontane und intuitive Entscheidung, eine Strategiefindung zu starten, gehört zur intrinsischen Art. Die Strategiefindung sollte in Workshops mit einer limitierten Dauer und einem definierten Team durchgeführt werden. Ideal sind Workshops mit einer Bruttodauer (inklusive aller Pausen) von minimal 4 bis maximal 8 h. Die Mitglieder des Teams sollten interne Experten für mobile Applikationen sowie Entscheidungsträger aus dem Management sein. Falls keine internen Experten vorhanden sind, können auch externe Experten hinzugezogen werden. Die ideale Teamgröße besteht aus minimal fünf Mitgliedern bis maximal zehn Teilnehmer. Zur Strategiefindung sollten minimal ein Workshop bis maximal drei Workshops durchgeführt werden. Die Phase der Findung wird mit einer Entscheidung zur weiteren Vorgehensweise abgeschlossen (Go/No Go). Dieser Meilenstein wird „Decision Gate“ genannt. Nach der erfolgreichen Strategiefindung wird die Phase der Strategiedefinition gestartet. Zu den Aufgaben der Definition gehören die Beschreibung der mobilen Applikation ggf. mit der Erstellung eines Mock-up und die Erarbeitung eines Business Cases. Abgeschlossen wird die Phase mit dem „Decision Gate Strategiedefinition“.

- Ein **Mock-up** ist ein Prototyp einer Applikation ohne Funktionalität und zeigt damit insbesondere das äußere Erscheinungsbild und die Benutzerführung (GUI) auf.
- Ein **Business Case** ist die Untersuchung der Rentabilität einer App-Strategie und enthält ein Mission Statement, die App-Beschreibung, das Marktpotenzial, die Vertriebs- bzw. Absatzziele, die Umsatz- und Kostenannahmen und die Berechnung des Ergebnisses und der Rentabilität für einen definierten Zeitraum.

In die darauffolgende Phase der Strategieumsetzung gehören alle Aufgaben bis zum eigentlichen Projektstart der App-Entwicklung. Dies beinhaltet insbesondere die Erstellung von Lasten- und Pflichtenheft, die Kalkulation der Entwicklungskosten und die Definition des Geschäftsmodells. Die Phase schließt mit dem „Decision Gate Strategieumsetzung“ und führt zur Projektfreigabe oder zum Projektstop (siehe Abb. 3.1).

- Ein **Lastenheft** (Synonyme u. a.: Requirements Specification, Anforderungsanalyse) beschreibt alle Anforderungen an die Leistungen einer mobilen Applikation. Das Lastenheft wird in der Regel durch den Auftraggeber erstellt.
- Ein **Pflichtenheft** (Synonyme u. a.: Conceptual Design, Sollkonzept, Feinkonzept) beschreibt die konkrete Vorgehensweise zur Erstellung der mobilen Applikation. Das Pflichtenheft wird in der Regel durch den internen oder externen Auftragnehmer erstellt und durch den Auftraggeber abgenommen.

**Abb. 3.1** Vorgehensweise  
Strategie für mobile  
Applikationen



- ▶ Ein **Geschäftsmodell** (engl. Business Model) beschreibt die Vorgehensweise zur Etablierung der mobilen Applikation im Markt oder im Unternehmen. Damit werden die Phasen zur konkreten Umsetzung der Ziele aus dem Business Case definiert.

### 3.1.1 Strategiefindung

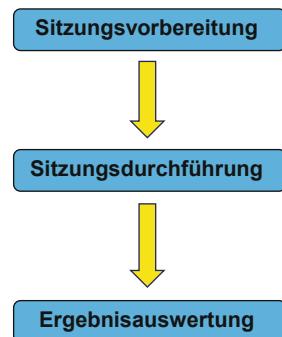
Bei Vorhandensein einer Anforderung zum Einsatz einer mobilen Applikation muss eine adäquate Strategie gefunden und entwickelt werden. Neben den Entscheidungsträgern für eine App-Entwicklung sollte in den ersten Workshops auch die notwendige Expertise aus dem App-Umfeld eingebracht werden. Hier können auch externe Experten integriert werden. Diese sollten aber nicht die Interaktion und Kommunikation in den Workshops dominieren, sondern nur pointiert Beiträge leisten.

Geeignete Methoden zur Eingrenzung und Fokussierung auf die relevanten Aspekte sind die Arbeitstechniken Brainstorming und Mindmap.

- ▶ **Brainstorming** ist eine Methode zur Ideenfindung bei der unkommentiert und nicht bewertet Ideen aller Teilnehmer aufgenommen werden. In einem zweiten Schritt können diese Ideen dann nach ihrer Wichtigkeit geordnet werden.

Der Initiator der Strategiefindung lädt zu dem Workshop die passenden Teilnehmer ohne Nennung des Themas ein. Die Auswahl der Teilnehmer ist dabei von hoher Bedeutung, da jeder weitere Durchgang die Dynamik der Ideenfindung reduziert und die Erfolgsquote eher suboptimal sein wird. In der Brainstormingsitzung, die von der Anzahl der Teilnehmer (5–10) und der Dauer (4–8 h brutto) limitiert sein sollte, sollten bestimmte Prinzipien beachtet werden:

**Abb. 3.2** Phasen des Brainstorming



- Kritik oder Unmutsäußerungen an den einzelnen Beiträgen bzw. Ideen sind verboten und sollten von dem Sitzungsleiter unmittelbar unterbunden werden.
- Argumentation und Gegenargumentation behindern das Ziel des Brainstormings, über eine große Bandbreite Ideen zu entwickeln. Auch hier muss der Sitzungsleiter (Moderator) schnellstmöglich Diskussionen und Dissonanzen beenden.
- Quantität geht vor Qualität, die Anzahl der Ideen und Beiträge ist entscheidend. Die Chance, die richtigen Ideen aus einer großen Quantität auch zu berücksichtigen, ist signifikant höher.
- Wichtig ist, dass der Fantasie der Teilnehmer freien Lauf gelassen werden sollte. Dieser dynamische Sturm der Gehirne (engl. Brainstorm) ermöglicht Emotionen und Intuitionen, die neue Aspekte und Ideen hervorbringen können.
- In einem zweiten Schritt sind Ideen weiterzuentwickeln und auszuformulieren.
- Durch Assoziationen werden ähnliche Ideen gebündelt, weitere Ideen entwickelt und Priorisierungen ermöglicht.

Die Phasen des Brainstormings sind in Abb. 3.2 dargestellt.

### Beispiel

Dieses Vorgehen soll an einem Beispiel zur Entwicklung einer mobilen Applikation verdeutlicht werden. Ein Beratungs- und Softwareunternehmen der Energiewirtschaft will als Erweiterung seines Dienstleistungs- und Produktangebots eine mobile Applikation anbieten. Der Strategiefindungsprozess wurde durch Anfragen und Aussagen von Kunden aus der Energiewirtschaft und durch die aktuellen technologischen Möglichkeiten zur Realisierung einer mobilen Applikation initiiert.

Die Energiewirtschaft ist seit vielen Jahren durch gesetzliche und regulatorische Vorgaben in einem Umbruch- und Neuordnungsprozess.

Seit Mitte der 90er Jahre sind die Energiemarkte Europas einem mehr oder weniger radikalen Wandel ausgesetzt. Durch die Vorgaben der EU und die Umsetzung dieser Vorgaben in nationale Vereinbarungen und Gesetze wurden die Energieunternehmen zur Liberalisierung und Deregulierung gezwungen. Eine erste Welle neuer Stromanbieter, die sich Ende

der 90er Jahre versucht haben zu etablieren, ist schon wieder vom Markt verschwunden. Die gesellschaftsrechtliche und informatorische Trennung von Erzeugung und Vertrieb von den Netzen ist letzten Endes für den Verbraucher ohne große Konsequenzen verpufft. Man kann zwar mittlerweile seinen Strom- und auch Gasanbieter ohne größere Probleme wechseln und auch Ökotarife (z. B. 100 % Atomstromfrei) buchen, aber der Hauptgrund für einen Wechsel ist der Preis. Und wer heute der Günstigste ist, kann schon morgen zu den teuren Anbietern gehören.<sup>1</sup>

Die große Vision der smarten Energie (Smart Energy) ist die zentrale und vorwiegend dezentrale Erzeugung des Stroms aus regenerativen Quellen. Dabei sollen sich Stromerzeugung und Stromverbrauch durch intelligente Netze (Smart Grids), zentrale und dezentrale Speichertechnologien und eine dynamische Steuerung des Verbraucherverhaltens durch intelligente Technologien in Smart Homes und adäquate Preisgestaltung durch den momentanen Erzeugungsquantitäten und -qualitäten angepasste Preise (z. B. durch dynamische Tarife) in einem Gleichgewicht befinden.

Der zukünftige Einsatz digitaler Zähler (sogenannter Automatic Meter oder Smart Meter) ermöglicht die Ermittlung sekundengenauer Verbrauchs- und Leistungswerte. Die Energieunternehmen können damit die Effizienz ihrer Abrechnungsprozesse erhöhen. Es wird nur in Rechnung gestellt, was auch verbraucht wird. Auch die Vorgaben des Gesetzgebers, wie z. B. Monatsrechnungen oder auch das Angebot von mindestens zwei Stromtarifen für einen Verbraucher, die ihm Energieeinsparungen ermöglichen sollen, werden damit eingehalten.<sup>2</sup> Aber profitiert auch der Verbraucher davon?

Mit Ausnahme von einigen wenigen technisch-affinen Verbrauchern wohl eher nicht. Die momentane Möglichkeit über z. B. Energy Cockpits die Sekundenverbräuche zu optimieren, potenzielle Energieräuber zu erkennen werden nur marginale Optimierungen erbringen. Zumal die Option digitale Zähler mit solchen Zusatz-Features zu erhalten auch größtenteils noch kostenpflichtig ist und die angebotenen Tarife sich in die schon Tag- und Nachtarife (HT: Hoch- und Niedertarif) beschränken.<sup>3</sup>

Unter diesen Rahmenbedingungen sollte die mobile Applikation einen Mehrnutzen für die Unternehmen der Energiewirtschaft oder die Kunden bringen.

In einem ersten Schritt sollen Ideen mit der Methode des Brainstormings entwickelt werden (siehe Abb. 3.3).

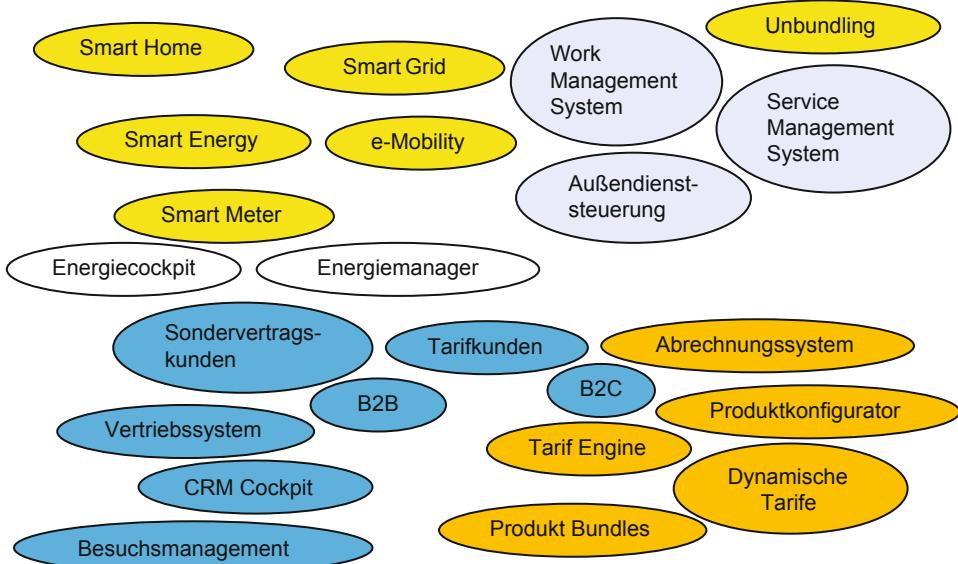
### Beispiel

Neben generischen Themen, wie Smart Grid, Smart Energy und Unbundling, die eher die Rahmenparameter für die Zielrichtung der mobilen Applikation darstellen, gibt es mehrere Bereiche, die für eine Anwendung geeignet erscheinen. Da sind zum einen Smart Home Applikationen denkbar, wie die Steuerung und das Management elektronischer Endgeräte in Häusern bzw. Gebäuden, und zum anderen Anwendungen wie ein Energiecockpit oder einen Energiemanager, die die ermittelten Verbrauchs-

<sup>1</sup> Vgl. Aichele 2012b, S. 1.

<sup>2</sup> siehe EnWG 2005.

<sup>3</sup> Vgl. Aichele 2012b, S. 1 f.



**Abb. 3.3** Ergebnis des Brainstormings „Mobile Applikation Energiewirtschaft“

und Leistungsdaten für den Endkunden transparent visualisieren. Diese beiden Produktmöglichkeiten sind für Energievertriebe, als einer der Protagonisten der Energiewirtschaft, oder für den Endkunden geeignet. Für Netzunternehmen, ein weiterer Akteur der Energiewirtschaft, die den Strom von den Erzeugern bis zum Verbraucher transferieren, kommen die Anwendungen aus den Bereichen „Work Management System“, „Service Management System“ und „Außendienststeuerung“ infrage. Bei diesen Systemen handelt es sich um mobile Unterstützungen für die Durchführung von Instandhaltungs-, Störfallbeseitigungs- und Serviceaufträgen. Die zentralen Backendsysteme senden die Aufträge an die mobilen Devices der Servicemitarbeiter, die, je nach Auftragsdurchführungsstatus, Rückmeldungen über die Apps erfassen können. Der nächste größere Anwendungsbereich ist die mobile Unterstützung der Vertriebsmitarbeiter mit mobilen Vertriebssystemen, einem CRM Cockpit (Customer Relationship Management) oder einem Besuchsmagementsystem, die den Mitarbeitern ermöglichen, die aktuellen Kundendaten abzurufen und Kundenanfragen bzw. Kundenaufträge direkt vor Ort aufzunehmen. Eine weitere Möglichkeit für eine mobile Applikation stellen Produktkonfiguratoren dar, die von den Kunden selbst, von Vertriebs- oder Außendienstmitarbeitern sowie auch im Innendienst eingesetzt werden könnten. Sie ermöglichen dem jeweiligen Anwender, in einer relativ kurzen Zeit Tarife und Produkte entsprechend den Anforderungen und Wünschen und den gegebenen Vorgaben zu gestalten und daraus in einem weiteren Schritt individuelle Verträge zu gestalten.

Nach der Sammlung der Produktideen ging es in der nächsten Phase um die Bewertung und Priorisierung der Möglichkeiten. Die 5 potenziellen Anwendungen sollten auf wenige Alternativen reduziert werden, um den notwendigen Aufwand in der restlichen Phase der Strategiefindung zu limitieren.

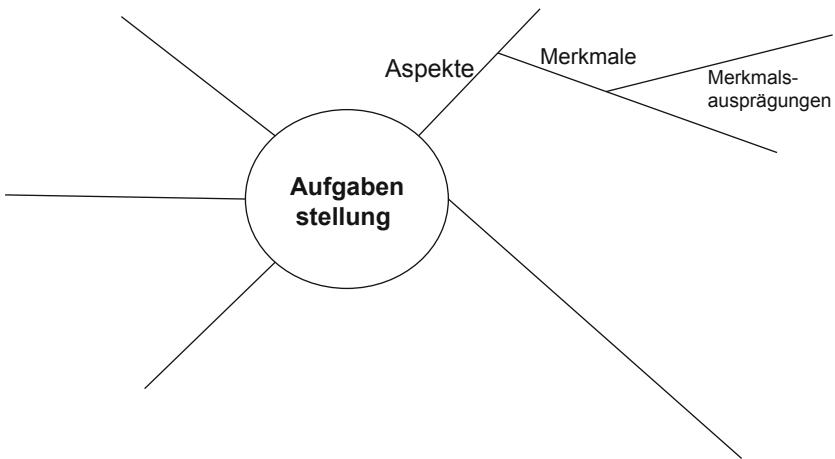
Smart Home Applikationen und auch Energiecockpits kommen als Kundenbindungsinstrumente für Energievertriebe sowie als Endkundenapplikationen infrage. Beide Anwendungen erfordern eine Einbindung in die technische Infrastruktur, die sich als überwiegend heterogen darstellt. Der Markt für diese Anwendungen ist groß und wird von Geräteherstellern und auch von Energiewirtschaftsunternehmen angegangen. Dabei ist die Bandbreite von Haushaltsgeräteherstellern über Hersteller von Smart Metern und reinen Elektronikfirmen sowie Energievertrieben sehr groß. Die schon vorhandenen Applikationen haben unterschiedlichste aber oft auch redundante Funktionalitäten und werden kostenlos in Zusammenhang mit Stromverträgen als auch kostenpflichtig angeboten. Die Generierung einer mobilen Applikation mit neuen mehrwertschöpfenden Funktionalitäten und einer eindeutigen Differenzierung erscheint schwierig. Insofern werden diese beiden Produktideen nicht weiter spezifiziert.

Work Management Systeme (oder Außendienststeuerung und Service Management Systeme) sind eine erhebliche Arbeitserleichterung für die Work Force (Service- bzw. Außendienstmitarbeiter), indem Aufträge den einzelnen Mitarbeitern qualitativ unter zeitlichen und logistischen Aspekten und unter Berücksichtigung der eigenen Qualifikation zugeordnet werden. Informationen, Fragen und Rückmeldungen können über die mobile Applikation zeitnah erfolgen. Solche Systeme stellen das logische Software Enhancement von Backendsystemen für das Instandhaltungs- bzw. das Servicemanagement dar und müssen in die proprietären Systemlandschaft eingebunden werden. Eine Stand-alone-Entwicklung macht ggf. für rein projektorientierte und zeitlich limitierte Vorgehen wie einen Smart Meter Rollout Sinn.<sup>4</sup> Die eigentliche mobile Applikation ist abhängig von den spezifischen Vorgaben der Projektvorhaben bzw. der proprietären Softwareanwendungen. Deshalb müsste eine Applikation in diesem Bereich detaillierter und fokussierter strukturiert sein oder eher ein Framework anbieten, das an spezifische Anforderungen schnell angepasst werden kann. Das Strategiefindungsteam entschied aufgrund der vorhandenen Rahmenparameter und des limitierten Marktpotenzials diese Alternative nicht weiterzuverfolgen.

Übertragen auf die vertrieblichen Prozesse trifft dieses Szenario auch für mobile Applikationen im Bereich Vertrieb zu. CRM Cockpits, Vertriebssysteme oder Besuchsmanagementsysteme sind die logische Erweiterung der Backendvertriebssysteme. Sie stellen eine sinnvolle Erweiterung und Unterstützung für die Vertriebsmitarbeiter dar, sind aber in die proprietären Anwendungssysteme einzubinden. Da fast alle Hersteller von Standardsoftware in diesem Bereich schon den Ansatz verfolgen, mobile Enhancements auf dem Markt zu etablieren, stellte für das Strategiefindungsteam diese Alternative kein erfolgversprechendes Szenario dar.

Mobile Produktkonfiguratoren, mit denen Vertriebsmitarbeiter, Callcentermitarbeiter aus dem Innendienst aber auch Kunden selbst sich individuelle Tarife (Strom und möglicherweise weitere Energiesparten) und Produkte gestalten können, stellen insbesondere unter den regulatorischen und gesetzlichen Vorgaben und den Möglichkeiten, die die Energiewende offeriert, ein überlegenswertes Differenzierungspotenzial

<sup>4</sup> Siehe zu Smart Meter Rollout: Aichele und Doleski 2013.



**Abb. 3.4** Mindmap

dar. Da bisher dynamische Tarife nicht zwingend vorgeschrieben waren und die gesamte Entwicklung in Abhängigkeit der Einführung von Smart Metern erst in den Kinderschuhen steckt, ist in diesem Fall eine extensive und dynamische Expansion zu erwarten.<sup>5</sup> Aufgrund des weiten Anwendungsbereiches und des erheblichen Marktpotenzials mit den entsprechenden Differenzierungsmöglichkeiten entschied das Strategiefindungsteam nur diese eine Produktalternative weiter zu spezifizieren.

In einem zweiten Teil des Workshops bzw. einem folgenden, zweiten Workshop kann aufbauend auf den eruierten Ideen eine weitere Fokussierung erfolgen. Dafür ist die Mindmap-Methode sehr gut geeignet.

► **Mindmap** ist eine kognitive Technik, die Aspekte zu einem Themengebiet visualisiert und zielgerichtete Fokussierungen ermöglicht.

Im Zentrum der Mindmap steht die Aufgabenstellung (siehe Abb. 3.4). Alle Aspekte zu dieser Aufgabenstellung werden ausgehend von dem zentralen grafischen Objekt (Kreis, Ellipse, Rechteck oder andere) an Ästen (grafische Linie) notiert. Zu den einzelnen Aspekten kann es ein oder mehrere Merkmale geben, die auf Ästen ausgehend von dem Aspektast dargestellt werden. Ausprägungen der Merkmale werden auf weiteren Ästen ausgehend von dem Merkmalast beschrieben. Aspekte, Merkmale oder ggf. Merkmalsausprägungen können durch Angabe von Zahlen (1, 2, 3 ...) priorisiert werden. Droht von einem Aspekt oder einem Merkmal oder ggf. einer Merkmalsausprägung Gefahr, wird das durch einen stilisierten Blitz verdeutlicht. Ist ein Aspekt, ein Merkmal oder eine Merkmalsausprägung von besonderer Bedeutung, wird dies durch eine Unterstreichung visualisiert. Termine und Terminabhängigkeiten zwischen Aspekten, Merkmalen und ggf. Merkmalsausprägungen

<sup>5</sup> Siehe zu dynamischen Tarifen: Motsch 2012, S. 229–258.

werden durch Angabe von fettgedruckten Terminen dargestellt. Assoziationen bzw. Verbindungen zwischen Aspekten, Merkmalen und Merkmalsausprägungen werden durch Pfeile mit gestrichelten (ggf. auch durchgezogenen) Linien realisiert. Ideen aus der ersten Phase des Workshops, dem Brainstorming, ohne eine Zuordnung zu Aspekten, Merkmalen oder Merkmalsausprägungen, werden am Rand der Mindmap vorläufig skizziert. In der finalen Mindmap sollte eine Zuordnung erfolgen.

### Beispiel

Nachdem das Strategiefindungsteam „Mobile Applikationen für die Energiewirtschaft“ sich für das Anwendungsgebiet mobiler Produktkonfiguratoren entschieden hatte, wurde in mehreren weiteren Workshops die Produktidee detailliert. Im ersten Schritt wurde dazu eine Mindmap entwickelt (siehe Abb. 3.5).

Mögliche Kundensegmente für die mobile Applikation „Produktkonfigurator für die Energiewirtschaft“ sind:

- Energievertriebe: Unternehmen der Energiewirtschaft, die Strom und andere Energiearten an den Endkunden liefern, wie z. B. Stadtwerke.
- Vergleichsportale: Hier können Endkunden ihren Wunschtarif konfigurieren und sich den passenden Versorger für diesen Tarif mit dem günstigsten Preis suchen lassen.
- Endkunden: Zu unterscheiden sind große Kunden mit einem hohen Verbrauch (sogenannte Geschäftskunden, Industriekunden oder in dem Branchenjargon Sondervertragskunden/SVK) und normale Haushaltskunden mit einem relativ geringen Verbrauch und damit einem geringen Umsatz mit marginaler Marge (im Branchenjargon Tarifkunden/TK).
- Energiehändler.

Der Produktkonfigurator macht in einem ersten Schritt Sinn für den Vertriebsaußen-dienst der Energievertriebe, um insbesondere in Verhandlungen mit SVK Tarife und Produkte zu gestalten. In einem zweiten Schritt könnten die Energievertriebe ihren Tarifkunden eine Konfiguration mit vorgegebenen Produktbausteinen erlauben (ähnlich dem Konfigurator eines Automobilherstellers) und so dem Kunden individualisierte Produkte anbieten. Um dabei den personellen Aufwand für den Vertrieb relativ gering zu halten, kann der Konfigurator auf dem eigenen Portal (eigene Internetseite) über Customer Self Services von dem Kunden selbsttätig aufgerufen werden. Potenzielle Anfragen müssen dann durch das Callcenter telefonisch, per E-Mail oder Messenger beantwortet werden. Vergleichsportale, die unabhängig von Energievertrieben deren Angebote übergreifend und objektiv als Information potenziellen Endkunden zur Verfügung stellen, können durch die Integration des Produktkonfigurators auf dem Portal ihre Attraktivität erhöhen und den Nutzern Mehrwert anbieten. Endkunden aus dem Bereich SVK können neben großen Industriekunden, die direkt von den Vertrieben über persönlichen Kontakt beraten werden, auch kleinere Geschäftskunden wie Handwerksbetriebe, Restaurants und Gaststätten oder Bäckereien sein. Für solche Geschäftskunden macht die selbsttätige Konfiguration über Portale auch Sinn. Energiehändler kaufen

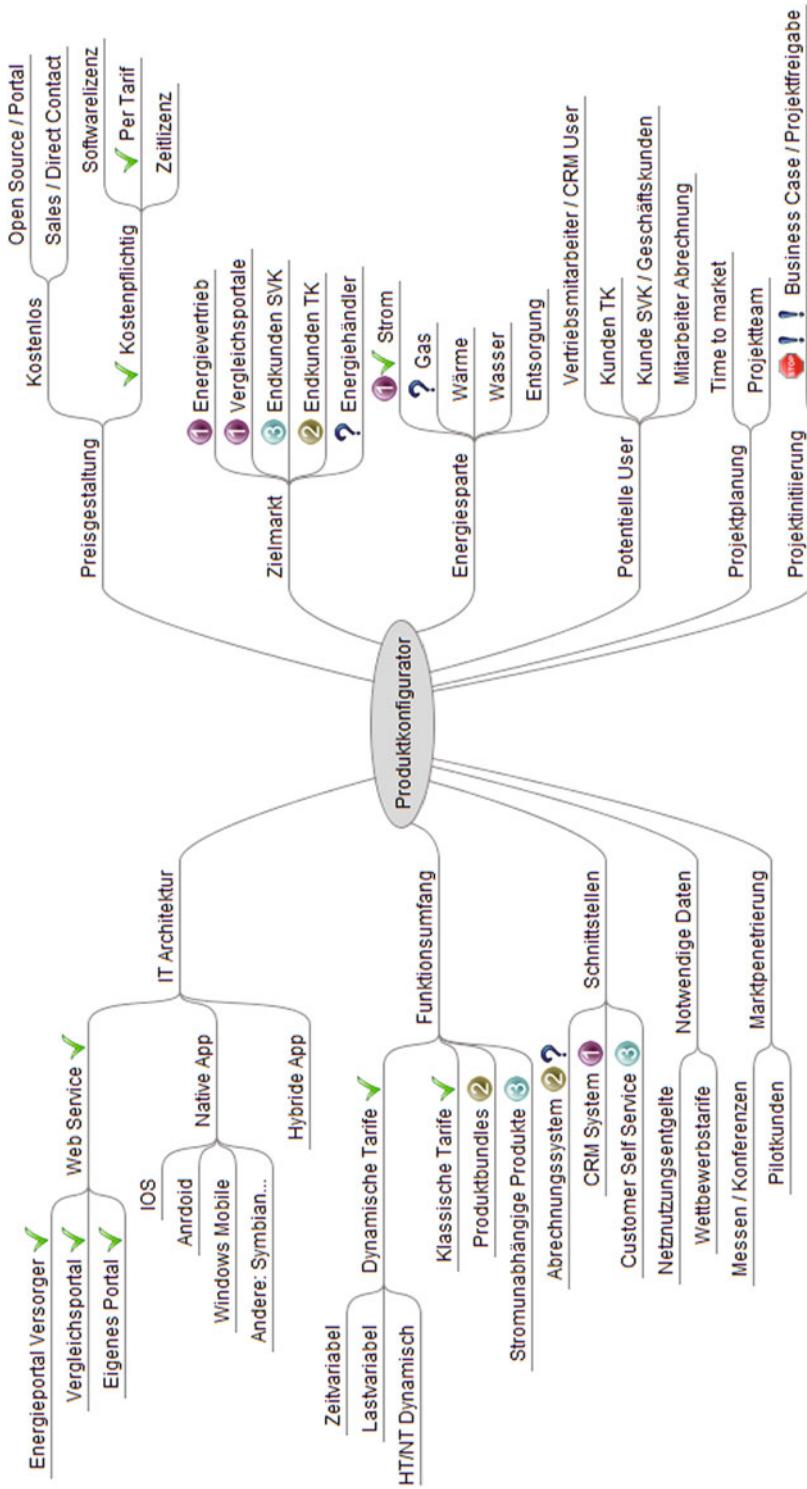


Abb. 3.5 Mindmap „Mobilier Produktkonfigurator“

Bezugsrechte für Energie von den Erzeugern und verkaufen Energie an Endkunden, die im Regelfall Energievertriebe und große Industrikunden darstellen, weiter. Eine mögliche Produktkonfiguration könnte auch hier infrage kommen, jedoch wären die technischen Anforderungen an die Funktionalität eines Konfigurators weitaus größer, die Produktbausteine und Module müssten eine immense Granularität aufweisen. Aus diesen Gründen wurden die potenziellen Kundensegmente wie folgt von dem Strategiefindungsteam priorisiert:

1. Energievertriebe, Innendienst (Callcenter, Abrechnung) bzw. eigene Portale und Vergleichsportale
2. Endkunden TK über Customer Self Services
3. Endkunden SVK über Customer Self Services

Mögliche Energiesparten, die von dem Konfigurator abgedeckt werden, sind Strom, Gas, Wasser, Wärme und Entsorgung. In Produktbundles könnten verschiedene Sparten auch miteinander kombiniert werden. Als Zusatzprodukte kommen auch energiefremde Artikel infrage, die als Give-aways, Add-ons oder in Form von Contractingverträgen an die Endkunden verkauft werden können. Die Bandbreite beginnt bei Artikeln wie Taschenlampen, LED-Leuchtmittel, LED- oder Energiesparlampen und endet bei Windkraftanlagen für einzelne Liegenschaften, Photovoltaikanlagen und Mikro-Blockheizkraftwerken. Da insbesondere in der Sparte Strom durch den vermehrten Einsatz der Smart Meter und der gesetzlichen und regulatorischen Vorgaben der Druck für ein breiteres und dynamischeres Produktspektrum am größten ist, entschied sich das Strategiefindungsteam, sich in einem ersten Schritte auf Strom zu konzentrieren. Eine modulare und objektorientierte Programmierung soll flexible und schnelle Erweiterungsmöglichkeiten auf andere Energiesparten und weitere Produkte ermöglichen.

Aus den potenziellen Kundensegmenten ergeben sich auch die möglichen Anwender der App. Dies sind Vertriebsmitarbeiter, Abrechnungsmitarbeiter und CRM-User der Energievertriebe. Im Rahmen des Einsatzes in Vergleichsportalen oder unternehmenseigenen Portalen können auch Tarif- und Sondervertragskunden (insbesondere kleinere Geschäftskunden) zu den direkten Anwendern des Konfigurators zählen.

Die Preisgestaltung wurde in dem Strategiefindungsteam kontrovers diskutiert. Zum einen gibt es die Möglichkeit des kostenpflichtigen App-Einsatzes in Form einer einmaligen Softwarelizenz ggf. mit Wartungskosten, einer zeitlich limitierten Lizenz und einer anwendungsbezogenen Lizenz, bei der jede Produktgenerierung Gebühren erfordert. Zum anderen besteht die Option, die App kostenlos dem Markt zur Verfügung zu stellen und z. B. über eine Open-Source-Strategie die multiple Erweiterung der Anwendung zu triggern oder über den eigenen Vertrieb Kunden die App kostenlos im Rahmen von Beratungs- oder Dienstleistungsprojekten zu offerieren. Letzten Endes fiel die Entscheidung des Managements für das Kundensegment der Energievertriebe auf die kostenpflichtige Variante und eine Gebühr per Produktgenerierung und für Vergleichsportale auf eine Gebühr pro generierten Vertrag durch einen konfigurierten Tarif.

Die mobile Applikation „Produktkonfigurator für die Energiewirtschaft“ soll auf weitestgehend allen mobilen Devices (Tablet-PC, Smartphone, Convertible Notebooks, Notebooks) lauffähig sein. Als zugrunde liegende IT-Architektur kommen Native Apps für die jeweils proprietären Betriebssysteme, Hybride Apps und Web Services in Frage.<sup>6</sup> Da die zu konfigurierenden Produkte auch (Energie-)Tarife darstellen, die aufgrund gesetzlicher, erzeugungsseitiger und wirtschaftlicher Vorgaben hohe Aktualisierungsraten in den Berechnungsparametern aufweisen, und da der Konfigurator auch über stationäre Devices (z. B. PCs im Callcenter und in der Abrechnung) eingesetzt werden soll, wurde der Webservice als optimale Entwicklungsoption gewählt. Der Webservice kann über jedes mobile und stationäre Device mit einem beliebigen Webbrowser aufgerufen werden und ist durch die Integration mit einer zentralen Datenbank immer auf dem gleichen, aktuellen Stand.

Der Mindestfunktionsumfang der App in der ersten Phase soll die Möglichkeit der Generierung dynamischer Tarife (zeit- und lastvariabel, variable Niedrig- und Hochpreistarife/NT, HT) und der klassischen Tarife sein. Relativ schnell soll in einer zweiten Version die Möglichkeit der Produktbundles offeriert werden, zuerst mit anderen Energiearten wie Gas, Wasser und Wärme (in Abhängigkeit des Angebots des einzelnen Energievertriebs bzw. Stadtwerks) und in einem nächsten Schritt mit einfachen, energieunabhängigen Produkten (z. B. LED-Lampen, Leuchtmittel, Energiecockpits u. a.). Das Anbieten von energieunabhängigen, komplexen Produkten wie Windkraftanlagen oder Mikro-Blockheizkraftwerken soll mit den zukünftigen Anwendern abgestimmt werden.

Als Schnittstellen kommen die Anbindungen an:

- CRM-Systeme (bidirektional: Kunden werden im CRM angelegt, CRM-System startet den Konfigurator, ggf. wird eine Bezugsnummer übergeben, Tarif/Produkt wird im Konfigurator gestaltet, Tarif/Produkt wird an CRM-System übergeben),
- CRM-Systeme (unidirektional: Kunden werden im CRM-System angelegt, CRM-User startet den Konfigurator, Tarif/Produkt wird im Konfigurator gestaltet, Tarif/Produkt wird an CRM-System übergeben),
- Abrechnungssysteme (bidirektional: Kunden werden im Abrechnungssystem angelegt, Abrechnungssystem startet den Konfigurator, ggf. wird eine Bezugsnummer übergeben, Tarif/Produkt wird im Konfigurator gestaltet, Tarif/Produkt wird an das Abrechnungssystem übergeben),
- Abrechnungssysteme (unidirektional: Kunden werden im Abrechnungssystem angelegt, Abrechnungsmitarbeiter startet den Konfigurator, ggf. wird eine Bezugsnummer eingegeben, Tarif/Produkt wird im Konfigurator gestaltet, Tarif/Produkt wird an das Abrechnungssystem übergeben),
- Customer Self Service (Einbindung des Konfigurators in das Portal eines Versorgers oder eines Vergleichsportals).

Aufgrund der im Regelfall vorhandenen Schnittstelle der proprietären CRM-Systeme zu Abrechnungssystemen und der relativ einfachen Einbindung eines Webservices

---

<sup>6</sup> Vor- und Nachteile der unterschiedlichen Derivate werden in Kap. 4 erläutert.

Project Charter				
<b>Projektname</b>				
<b>Genehmigung</b>	<b>Name</b>	<b>Funktion</b>	<b>Datum</b>	<b>Unterschrift</b>
Antragsteller				
Genehmigt		Projektmanager		
Genehmigt		Sponsor		
<b>Kontext und Background (Ausgangslage)</b>				
<b>Erwartete Business Benefits (Geschäftsmodell)</b>				
Projektstartdatum		<b>Projektenddatum</b>		
<b>Projektziele</b>				
<b>Projektergebnisse</b>				
<b>Projektumfang (Scope)</b>				
	enthält			
	ist nicht enthalten			

**Abb. 3.6** Project Charter Teil 1

in ein Portal wurde von dem Strategiefindungsteam die unidirektionale Schnittstelle zu CRM-Systemen favorisiert. Durch die Realisierung der Datenübergabe mit XML (Extensible Markup Language<sup>7</sup>) ist die Realisierung der Schnittstelle zu verschiedenen CRM-Systemen sowie auch die Einbindung in beliebige Portale gewährleistet.

Notwendige Daten für den Konfigurator sind neben den Produktbausteinen, die Netznutzungsentgelte, die MSB/MSD-Entgelte<sup>8</sup>, die gesetzlichen Steuern und Abgaben und ggf. dezidierte Wettbewerbsdaten.

Die Marktpenetrierung soll über die Positionierung auf Messen und in Konferenzen sowie über die Akquisition von Pilotkunden erfolgen.

Für die Projektinitiierung ist eine Projekt freigabe auf Basis eines verifizier- und validerbaren Business Cases notwendig. Der Business Case wird in der nächsten Phase der Strategiefindung erarbeitet. Die Projektplanung soll während der Strategieumsetzung starten.

Der abschließende Schritt in der Phase Strategiefindung beinhaltet die Erstellung des Project Charters. In diesem werden die App, die Zielgruppe, der Lieferumfang, die Funktionalität, die Vermarktungsstrategie und die Meilensteintermine beschrieben (siehe Abb. 3.6 und 3.7).

<sup>7</sup> XML ist eine normierte, textbasierte Datendarstellung, die zur plattform- und implementationsunabhängigen Datenübergabe genutzt werden kann.

<sup>8</sup> Messstellenbetreiber/Messstellendienstleister.

<b>Erfolgsfaktoren (Critical Success Factors)</b>				
<b>Methode und Vorgehensweise</b>				
Projektressourcen				
Lenkungsausschuss				
Sponsor				
Projektleiter				
Projektmitarbeiter				
Externe Experten				
Andere				
<b>Kostenschätzung</b>				
Kosten				
Mitarbeitertage				
<b>Risiken</b>				
<b>Annahmen</b>				
<b>Einschränkungen und Abhängigkeiten</b>				
<b>Reporting</b>				
Meetings		<b>Frequenz</b>	<b>Teilnehmerkreis</b>	
Lenkungsausschuss				
Projektteam				
<b>Reports</b>				
Projektbericht				
Open Issues				

**Abb. 3.7** Project Charter Teil 2

- In dem **Project Charter** werden die Ausgangslage, die Zielsetzung, das Umfeld, die Ergebnisse, die Kosten und der Nutzen sowie die Projektorganisation eines Entwicklungsprojektes beschrieben. Der Project Charter kann die Status Vorschlag, Antrag und Auftrag haben.

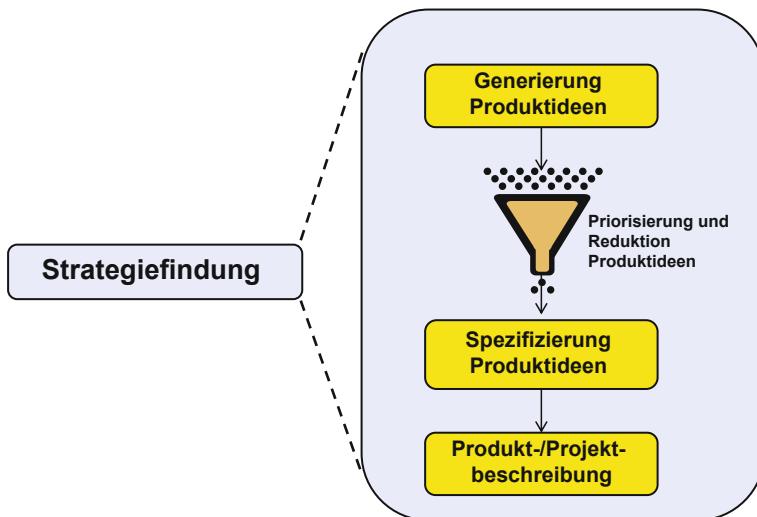
In der Phase Strategiefindung hat der Project Charter den Status Vorschlag.

### Beispiel

Für die App „Produktkonfigurator Energiewirtschaft“ wurde ein vereinfachter Project Charter erstellt (siehe Abb. 3.8). Zielsetzung war eine weitere Detaillierung in der Phase Strategiedefinition. Für die App wurde auch der vorläufige Name „e-configurator“ von dem Strategiefindungsteam festgelegt.

Damit besteht die Phase der Strategiefindung aus vier Schritten und dem Decision Gate (siehe Abb. 3.9):

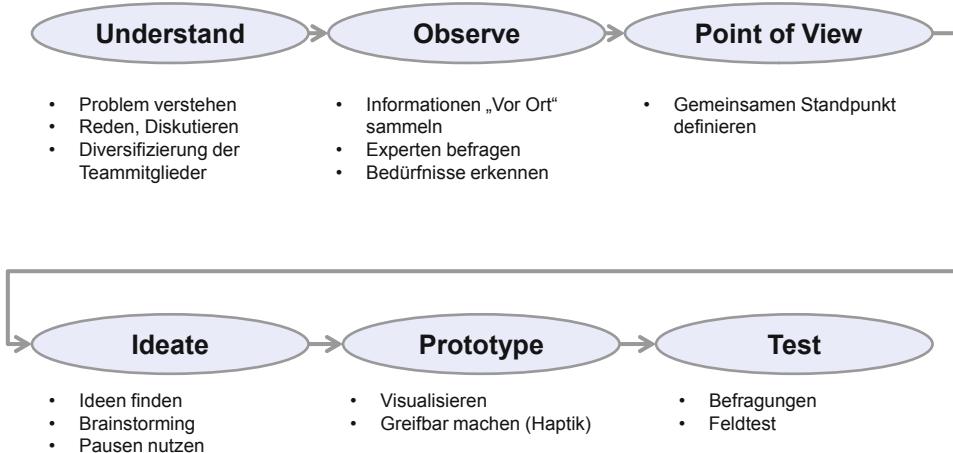
- Der Generierung der Produktideen. Hierfür wird die Methode „Brainstorming“ eingesetzt. Alternativ zu einem Präsenzbrainstorming bietet sich auch ein webbasiertes Brainstorming an (über Chatrooms, Messenger oder Foren). Die Kreativitätstechnik „Design Thinking“ erweitert die Brainstormingmethode mit einer definierten Vorge-

**Abb. 3.8** Project Charter e-configuretor**Abb. 3.9** Einzelschritte in der Phase Strategiefindung

hensweise und designorientierten Prototypen, die auch für einen ersten Feldtest genutzt werden können (siehe Abb. 3.10).<sup>9</sup>

- Der Priorisierung und Reduktion der Produktideen. Anschließend zu der Brainstormingphase oder in einem weiteren Workshop werden die Ideen geclustert und priorisiert. Durch Informationen über die Marktsituation, Wettbewerberprodukte und vorgegebene Rahmenbedingungen können die einzelnen Ideen in eine gewichtete Reihenfolge gebracht werden. Das Strategiefindungsteam entscheidet über die Brisanz der

<sup>9</sup> Siehe zu Design Thinking: Hasso-Plattner-Institut 2013.



**Abb. 3.10** Design Thinking

Alternativen und wählt ein bis mehrere Alternativen zur weiteren Detaillierung aus. Je weniger Alternativen hier ausgewählt werden, umso stringenter kann die weitere Vorgehensweise erfolgen.

3. Der Spezifizierung der Produktideen. Je Produktalternative wird mit der Methode Mindmap das potenzielle Geschäftsmodell skizziert. Auf Basis des geschätzten Nutzens und der erforderlichen Kosten werden hier ggf. weitere Produktalternativen aus der weiteren Betrachtung ausgenommen.
4. Die Produkt- oder Projektbeschreibung. Die verbliebenen oder die verbliebene Produktalternative werden oder wird in einem Project Charter beschrieben.
5. Entscheidung für oder gegen die Fortführung des Projekts (Decision Gate Strategiefindung).

Das Design Thinking ermöglicht über die Visualisierung der Prototypen eine frühzeitige Einbeziehung der späteren Anwender. Die Haptik der Applikation kann durch das Greifbar machen frühzeitig erfahren werden. Dabei werden nicht nur die Bildschirmabfolgen per Mock-ups dargestellt, sondern durch den Einsatz von Smartphones, Tablets oder reinen Modellen von mobilen Devices wird eine realitätsnahe Simulation erreicht.

### 3.1.2 Strategiedefinition

Hauptaufgabe der Phase Strategiedefinition ist die Erstellung eines detaillierten Businessplans mit einem Business Case, der das App-Projekt quantifiziert.

- **Ein Businessplan** beschreibt ein Geschäftsmodell, die Maßnahmen und Vorgehensweisen, die ergriffen werden müssen, um das Geschäftsmodell zu realisieren. Der Businessplan enthält Kennzahlen, die das Geschäftsmodell quantifizieren. Diese Quantifizierung kann zu Teilen bzw. vollständig dem Business Case entsprechen.

**Die typische Struktur eines Businessplans für eine Unternehmensgründung bzw. für die Entwicklung eines neuen Produktes gliedert sich wie folgt:**

1. Einleitung
2. Das Unternehmen // ***Das Produkt***
  - 2.1 Unternehmensgründer und Gründungsidee // ***Gründungsidee***
  - 2.2 Zielsetzung
  - 2.3 Rechtsform // Projektorganisation
  - 2.4 Standort
3. Technologie und Produkte/Dienstleistungen
  - 3.1 Stand der Technik
  - 3.2 Innovation
  - 3.3 Produkte/Dienstleistungen // ***Applikation***
4. Der Markt
  - 4.1 Entwicklung und Struktur des Gesamtmarktes
  - 4.2 Nachfrager
    - 4.2.1 Marktsegmente
    - 4.2.2 Das geeignete Marktsegment
  - 4.3 Konkurrenten
  - 4.4 Differenzierung
  - 4.5 Erfolgsfaktoren
  - 4.6 Risiken
5. Marketing
  - 5.1 Marketingziele
    - 5.1.1 Absatzplanung
    - 5.1.2 Marktanteile
  - 5.2 Marketingmix
    - 5.2.1 Produkte // ***Applikation***
    - 5.2.2 Preise
    - 5.2.3 Distribution
    - 5.2.4 Absatzförderung
6. Geschäftsbeziehungen

## 7. Forschung und Entwicklung

### 7.1 Ziele

### 7.2 Ressourcen

## 8. Personalentwicklung und Organisation // **Projektorientation**

### 8.1 Organigramm und Stellenbeschreibung // **Projektruktur**

### 8.2 Personalentwicklung

## 9. Finanzierung

### 9.1 Finanzbedarfsplanung

### 9.2 Finanzierungsplanung

### 9.3 Liquiditätsplanung

## 10. Gründungsbilanz und Jahresabschlüsse // **Absatz – und umsatzplan**

### 10.1 Umsatzplan // **Absatz – und umsatzplan**

#### 10.1.1 Jahresumsatz Jahr 1

#### 10.1.2 Jahresumsatz Jahr 2

#### 10.1.3 Jahresumsatz Jahr 3

### 10.2 Ergebnisrechnung (GuV)

### 10.3 Gründungsbilanz // **entfällt**

### 10.4 Abschluss des ersten Jahres // **entfällt**

### 10.5 Break-Even- und Kennzahlenanalyse

### 10.6 Abschluss des zweiten Jahres // **entfällt**

### 10.7 Abschluss des dritten Jahres // **entfällt**

## 11 Anhang

### Anhang 1: Beschreibung der Produkt- und Serviceleistungen

#### 1.1 Serviceleistungen

##### 1.1.1 ...

##### 1.1.2 ...

#### 1.2 Produktleistungen

##### 1.2.1 .....

##### 1.2.2 .....

#### 1.3 Seminare

### Anhang 2: CVs Management // Projektleitung

#### Anhang 2.1 CV ...

#### Anhang 2.2 CV ...

### Anhang 3: Literaturverzeichnis

- Ein Businessplan ist für größere Entwicklungsprojekte in einem Umfang von mehreren hundert Entwicklertagen und mit einer Projektmitarbeiteranzahl im zweistelligen Bereich sowie einer Laufzeit größer einem Jahr erforderlich. Bei kleineren Projekten, die insbesondere im Bereich der mobilen Applikationen typisch sind, reicht die Erstellung eines Business Cases.

Die oben angeführte Struktur eines Businessplans enthält die wichtigsten Punkte für die Gründung eines Unternehmens oder für die Entwicklung eines Produktes bzw. einer Applikation. Überschriften bzw. Gliederungspunkte, die für Applikationen relevant sind, können durch die kursive Darstellung des Textes nach einem Doppelschrägstrich (double slash: //Text) erkannt werden. Zum Teil entfallen für Applikationen wichtige Punkte für Unternehmen (wie z. B. Gründungsbilanzen und Jahresabschlüsse). Dies wird durch die kursive Darstellung des Textes „entfällt“ nach einem Doppelschrägstrich (double slash: //entfällt) verdeutlicht. Ein Businessplan hat in Abhängigkeit des Investitionsvorhabens einen Umfang von 50 bis zu mehreren hundert Seiten.

Die Gliederung eines Business Cases ist in der Regel einfacher und enthält weniger Einzelpunkte. Der Business Case ist von geringerem Umfang (Text 10–15 Seiten, Präsentation 10–15 Seiten) und enthält ein Kalkulationsmodell, in dem die Rendite der einzelnen Anwendung ggf. unter Zuhilfenahme verschiedener Szenarien oder Varianten detailliert errechnet wird. Ein Business Case kann auch ausschließlich in Präsentationsform erstellt werden.

---

#### Business Case

1. Mission Statement
2. App-Beschreibung
3. App Marktpotenzial
4. App Vertriebsziele
5. Business Case Annahmen und Varianten
6. Business Case Varianten
7. SWOT Analyse
8. Status Service Portfolio und Partner
9. Marketing und Sales Activities
10. Recommended Approach

Das Mission Statement beschreibt die Zielsetzung der App in dem Marktumfeld und wie das Unternehmen sich mit der App positionieren möchte. In diesem Fall kann das Unternehmen den App-Erzeuger oder auch den App-Anwender darstellen. Der Umfang eines Mission Statements sollte eine halbe bis maximal eine ganze Seite betragen.

- Ein **Mission Statement** ist die Erklärung einer Organisation über ihre Positionierung im Marktumfeld, wie sich die Organisation selbst sieht und wie die App dazu beitragen kann, die Organisationsziele zu erreichen.

**Beispiel**

Das Mission Statement für die Applikation Produktkonfigurator für die Energiewirtschaft „e-configurator“ lautet wie folgt:

Wir werden als erfahrener Beratungs- und Lösungshaus für die Branche Energiewirtschaft angesehen. Die Applikation e-configurator stellt die qualitative Referenz im jeweiligen Anwendungsfall dar.

Die App-Beschreibung wird als Extrakt aus dem Project Charter entwickelt und enthält grundsätzliche Aussagen zu der Zielsetzung, der Funktionalität der App, dem Lieferumfang, den Sales Channels, der Vermarktsungsstrategie und zu den wichtigsten Meilensteinen bzw. Terminen. Idealerweise kann die App-Beschreibung auf einer Seite erfolgen, das Maximum von zwei Seiten sollte nicht überschritten werden.

**Beispiel**

Für die Applikation e-configurator entspricht die App-Beschreibung des Business Cases dem (reduzierten) Project Charter aus Abb. 3.8.

Das Marktpotenzial umfasst alle möglichen Kunden in einer definierten Region für die mobile Applikation. Zumeist werden Wahrscheinlichkeiten der Anzahl der Kunden, die aus dem gegebenen gesamten Potenzial gewonnen werden können, für die weiteren Berechnungsschritte angenommen. Hier sollte eher ein konservativer Ansatz gewählt werden. Ggf. werden unter Zuhilfenahme von verschiedenen Szenarien pessimistische, konservative und optimistisch-progressive Ansätze für die Entscheidungsträger dargestellt. Entscheidend für die Höhe der Wahrscheinlichkeit ist zum einen ob der Zielmarkt sich im B2B oder B2C-Segment befindet und zum anderen die Wettbewerbssituation und die Aktualität der Applikation insbesondere in Hinsicht auf die potenzielle Nachfrage. Weitere Parameter, die berücksichtigt werden müssen, sind die vorhandenen oder notwendigen Vertriebskanäle, die Marktpenetrierungsmöglichkeiten und -strategien, die vorhandenen Finanzmittel, die verbundenen Partner (-unternehmen) und ggf. vorhandene Benchmarks aus historischen Business Cases anderer Anwendungen, die zur Verifizierung und Validierung herangezogen werden können.

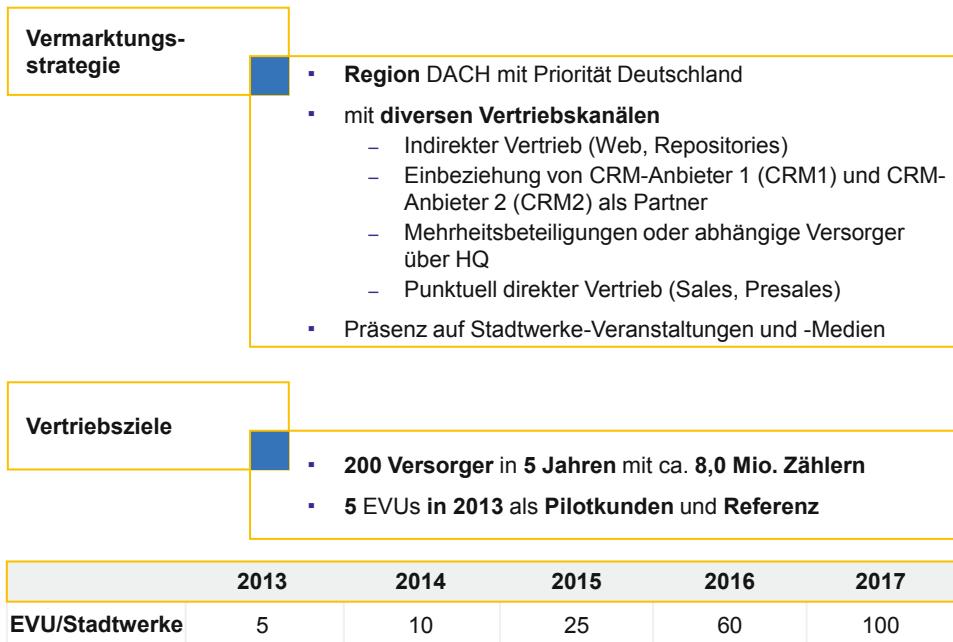
**Beispiel**

Für die App „e-configurator“ kommen als wesentlicher Markt die Energievertriebsunternehmen, d.h. insbesondere die Stadtwerke, infrage. Durch in Energiewirtschaftsunternehmen durchgeführte Beratungs- und Softwareeinführungsprojekte ist zum Teil ein indirekter bzw. direkter Zugriff auf diese Unternehmen möglich. Größere Energiekonzerne besitzen Mehrheits- und Minderheitsbeteiligungen an Energievertrieben bzw. Stadtwerken und haben damit die Möglichkeit der Einflussnahme.

e-configuretor Marktpotential	<ul style="list-style-type: none"> <li>ca. 800 kleine und mittlere Stadtwerke, davon ca. 120 Stadtwerke aus Energieversorgungsunternehmen 1 (EVU1) Partnerumfeld, ca. 20 Stadtwerke (EVU2), einige 100 Stadtwerke (EVU3)</li> <li>ca. 100 größere Versorger/Stadtwerke, davon 20 sehr große weitere ca. 60 Energievertriebe</li> <li>Region ACH mit weiterem Potential</li> </ul>					
Wahrscheinlichkeit	Anzahl Versorger			Anzahl Kunden/Zähler		
	Gesamt	Mehrheitsbeteiligung	Minder- u. Nichtbeteiligung	Gesamt	Mehrheitsbeteiligung	Minder- u. Nichtbeteiligung
Hoch	90	45	45	3.600.000	1.800.000	1.800.000
Mittel	720	360	360	28.800.000	14.400.000	14.400.000
Niedrig	150	75	75	6.000.000	3.000.000	3.000.000
in Bearbeitung	20	10	10	800.000	400.000	400.000

**Abb. 3.11** Marktpotenzial e-configuretor

Das Chart Marktpotenzial e-configuretor in Abb. 3.11 besteht aus einem qualitativen und einem quantitativen Teil. In dem qualitativen Teil werden die Parameter zur Findung der Kennzahlen für die quantitative Analyse angeführt. Der gesamte Markt besteht aus ca. 980 Stadtwerken und reinen Energievertrieben. Durch bisherige Projekte und damit verbundene Referenzen besteht zu einem Teil dieser Unternehmen ein besserer Zugang. Neben dem deutschen Markt gibt es noch den österreichischen und schweizerischen Markt, der nicht näher quantifiziert wird (Region ACH), da die dortige Einführung dynamischer und komplexer Produkte erst zu späteren Zeiträumen geplant ist. Für den quantitativen Teil wurden von dem Strategiedefinitionsteam die Wahrscheinlichkeiten in „Hoch“, „Mittel“ und „Gering“ unterschieden und die entsprechende Anzahl an Unternehmen geschätzt. Potenzielle Kunden, zu denen schon Kontakt besteht und die sich in der Akquisitionsphase befinden, finden sich in der Zeile „In Bearbeitung“ wieder. Da in dem Energiesektor die Anzahl der (Strom-)Zähler ein entscheidender Indikator für die Größe der Unternehmen bzw. das Kundenvolumen der Unternehmen ist und ggf. der Zähler als Abrechnungsfaktor eine Rolle spielen könnte, wird in einem weiteren Segment der Tabelle die entsprechende Umrechnung der Unternehmenszahl in Zähler ausgewiesen (Annahme: Durchschnittswert je Stadtwerk 40.000 Zähler). Aus den Zeilen Wahrscheinlichkeit „Hoch“ und „In Bearbeitung“ werden in einem nächsten Schritt die geplanten Absatzzahlen abgeleitet.



**Abb. 3.12** Vertriebsziele e-configurator

- Die Berechnung des Business Cases wird am besten in einem passenden Programm (Kalkulationssoftware oder Tabellenkalkulation) durchgeführt. Dafür werden die quantifizierten Marktpotenziale als Berechnungsbasis erfasst.

Aus dem Marktpotenzial werden die Vertriebsziele mit konkreten Absatzzahlen abgeleitet. Die Vertriebsziele beinhalten eine Vertriebs- oder Vermarktungsstrategie mit einer Aufzählung qualitativer Maßnahmen zur Erreichung der Absatzzahlen. Als Übersichtspunkt wird der gesamte Absatz in dem geplanten Absatzzeitraum angeführt. In einer tabellarischen Form werden dann die Absatzzahlen pro Periode des Planzeitraums aufgezeigt (siehe Abb. 3.12).

### Beispiel

Die Vermarktungsstrategie des e-configurators betrifft die Region DACH (Deutschland, Österreich/A, Schweiz/CH) mit einem primären Fokus auf Deutschland. Die wichtigsten Vertriebskanäle sind über das Internet (App-Repositories, App-Stores u. a.), über bestehende Kunden und deren Beteiligungen an Energieversorgungsunternehmen (EVU) bzw. Stadtwerken und punktuell der direkte Vertrieb, insbesondere im Zusammenhang mit Beratungsprojekten. Das Absatzziel sind 200 Kunden in den nächsten fünf Jahren und davon die Gewinnung von 5 Referenzkunden in dem ersten

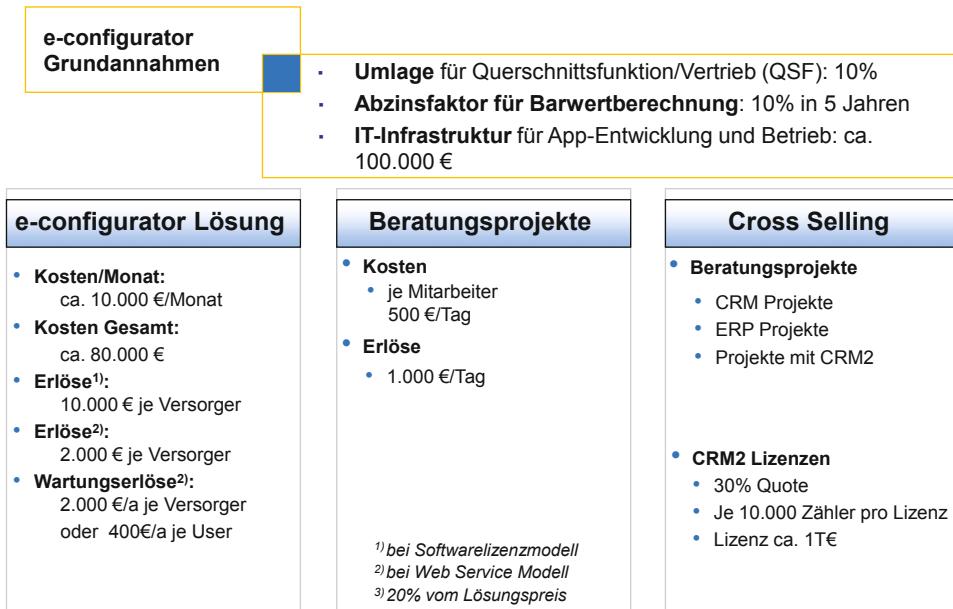


Abb. 3.13 Grundannahmen Business Case e-configurator

Absatzjahr. Die 200 Kunden sind dabei auf den Absatzzeitraum von fünf Jahren in einem einjährigen Raster aufgeteilt (siehe Abb. 3.12).

Zur Berechnung der Erlöse und Kosten des Business Cases müssen Parameter bzw. Grundannahmen definiert werden (siehe Abb. 3.13). Mit der in verifizierbaren Intervallen möglichen Änderung der Parameter ist auch eine Variation der Ergebnisse realisierbar.

### Beispiel

Der Business Case e-configurator wird in 3 Varianten erstellt. Die alternativen Szenarien sind „Softwarelizenzmodell“, „Web Service“ und „Accelerator“. Bei dem Softwarelizenzmodell wird die Anwendung als Software an den Kunden geliefert. Der Kunde betreibt die Software auf eigenen Webservern bzw. Portalen und kann den Quellcode im Rahmen der vorgegebenen Wartungsbedingungen beliebig anpassen. Das Softwarelizenzmodell ist mit einem laufenden Wartungsvertrag verbunden. Bei dem Web-Service-Modell wird die Applikation auf einem Webserver des Unternehmens betrieben. Kunden loggen sich über das Web in die Applikation ein und können die Ergebnisse durch die XML-basierte Schnittstelle in eigenen Systemen weiter bearbeiten. Bei diesem Modell arbeiten viele Kunden mit einer einzigen App. Dementsprechend sind die Gebühren geringer bzw. fallen nur bei Nutzung an. Auch das Web-Service-Modell beinhaltet einen Wartungsvertrag. Das Modell Accelerator basiert auf der

Annahme, dass durch Zurverfügungstellung der App Beratungsprojekte generiert werden können bzw. dass die App im Rahmen von Beratungsakquisitionen oder -projekten als Differenzierungspotenzial eine äußerst positive Wirkung erzielen kann. Die Grundannahmen für den Business Case e-configurator sind (siehe Abb. 3.13):

- Die indirekten Kosten für die notwendigen Querschnittsfunktionen (QSF) wie Administration und Vertrieb betragen 10 % der direkten Kosten für Entwicklung und Betrieb der App.
- Der Abzinsfaktor für die Barwertberechnung beträgt 10,1 %. Der Barwert zeigt den Wert einer Investition zum Startzeitpunkt. Erreicht wird das durch die Abzinsung der zukünftigen Ein- und Auszahlungen.
- Die einmaligen Investitionskosten für die IT-Infrastruktur (Soft-, Net- und Hardware) für die App-Entwicklung und den zukünftigen Betrieb der App betragen 100 Tsd. €.

Für die eigentliche App „e-configurator“ fallen folgende Kosten und Erlöse an:

- Die gesamten Entwicklungskosten werden mit 80 Tsd. € geschätzt. Die Kosten für die weiteren Entwicklungsstufen und den Betrieb der App sind 10 Tsd. € je Monat.
- Bei einem reinen Softwarelizenzmodell betragen die Erlöse 10 Tsd. € je Kunde. Bei einem Web-Service-Modell 2 Tsd. € je Kunde. Die Wartungsgebühr beträgt 20 % der einmaligen Erlöse pro Jahr.

In dem Szenario „Accelerator“ werden die Umsätze über Beratungsprojekte generiert.

Die Annahmen dafür sind:

- Die Kosten je Mitarbeiter betragen 500 € pro Tag.
- Die durchschnittlichen Erlöse in Beratungsprojekten werden mit 1000 € je Mitarbeitertag definiert.

Durch Cross-Selling-Effekte werden weitere Effekte erwartet:

- Durch den Vertrieb und Verkauf der App e-configurator werden Projekte im Bereich Customer Relationship Management (CRM) und Enterprise Resource Planning (ERP) getriggert. Die ERP-Projekte können in den Funktionsbereichen Instandhaltung, Kundenservice oder Abrechnung anfallen.
- Durch die Vertriebspartnerschaft mit einem CRM-Softwarehersteller (CRM2) und die Anpassung der CRM-Software an die Bedürfnisse von Energievertrieben bzw. Stadtwerken ergeben sich auch positive Aspekte für die App e-configurator. Die Verkaufschance für den e-configurator ist bei einem Kunden der CRM-Software enorm erhöht. Je verkaufter CRM-Lizenz wird eine Erfolgsquote für einen Folgeverkauf des e-configurators mit 30 % angenommen. Da die CRM-Lizenzen für klein- und mittelständische Stadtwerke passgenau sind, wird eine Lizenzgebühr von 1.000 € je angefangenen 10.000 Zähler als adäquat angesehen. D. h. ein kleines Stadtwerk mit 15.000 Stromzählern müsste eine Gebühr von 2.000 € für die App entrichten.

Der Business Case kann in verschiedene, potenziell mögliche Szenarien aufgeteilt werden. Diese Varianten können auf unterschiedlichen Geschäftsmodellen oder unterschiedlicher Ausprägung der Berechnungsparameter beruhen. So können zum Beispiel auf Basis pessi-



**Abb. 3.14** Varianten Business Case e-configurator

mistisch oder optimistisch angenommener Absatzzahlen die folgenden Szenarien erstellt werden:

- Worst-Case-Ansatz (pessimistische Absatzzahlen)
- Normal-Case-Ansatz (Mittelwert)
- Best-Case-Ansatz (optimistische Absatzzahlen)

Entsprechend unterschiedliche Varianten des Business Cases können auch auf unterschiedlich eingeschätzten Erlöszahlen beruhen. Auch die entsprechende Kombination der unterschiedlichen Annahmen mehrerer Parameter können in den Szenarien generiert werden.

### Beispiel

Für den Business Case e-configurator wurden die drei unterschiedlichen Geschäftsmodelle

1. Accelerator
2. Web Service
3. Softwarelizenz

als Basis verwendet. Die Szenarien werden in Abb. 3.14 kurz erklärt und die grundlegenden Vor- und Nachteile gegenüber gestellt.

Die Darstellung des Business Cases bzw. eines Business-Case-Szenarios erfolgt idealerweise auf einer Seite. Die grundsätzlichen Annahmen (Geschäftsmodell und Parameter) des Business Cases werden noch einmal in der Übersicht dargestellt (ggf. nur skizziert). Die Darstellung des eigentlichen Business Cases erfolgt in tabellarischer Form. Die Angabe der zugrunde gelegten Absatzzahlen erfolgt in dem Kopf der Tabelle. In dem Mittelteil werden die Umsätze und Kosten detailliert je Betrachtungsperiode aufgelistet. Daraus ergeben sich die Ergebnisse. In dem Fußteil werden die verwendeten Kennzahlen oder Key Performance Measures (KPI) angeführt. Diese können zum Beispiel sein:

- Der Return on Invest (ROI): Der ROI ist eine Spitzenkennzahl (rechentechnisch verknüpftes Kennzahlensystem) zur Ermittlung des Erfolgs eines Unternehmens ermittelt am Gewinn im Verhältnis zum eingesetzten Kapital (siehe Abb. 3.15).
- Der Kapital- oder Barwert (englisch: Net Present Value/NPV): Der Kapitalwert zeigt den Wert einer Investition zum Startzeitpunkt. Erreicht wird das durch die Abzinsung der zukünftigen Ein- und Auszahlungen.
- Die Internal Rate of Return (IRR, deutsch: Interne Zinsfuß-Methode): Dynamische Investitionsrechnung zur Ermittlung der jährlichen Rendite bei variierenden Erträgen.

---

### Beispiel

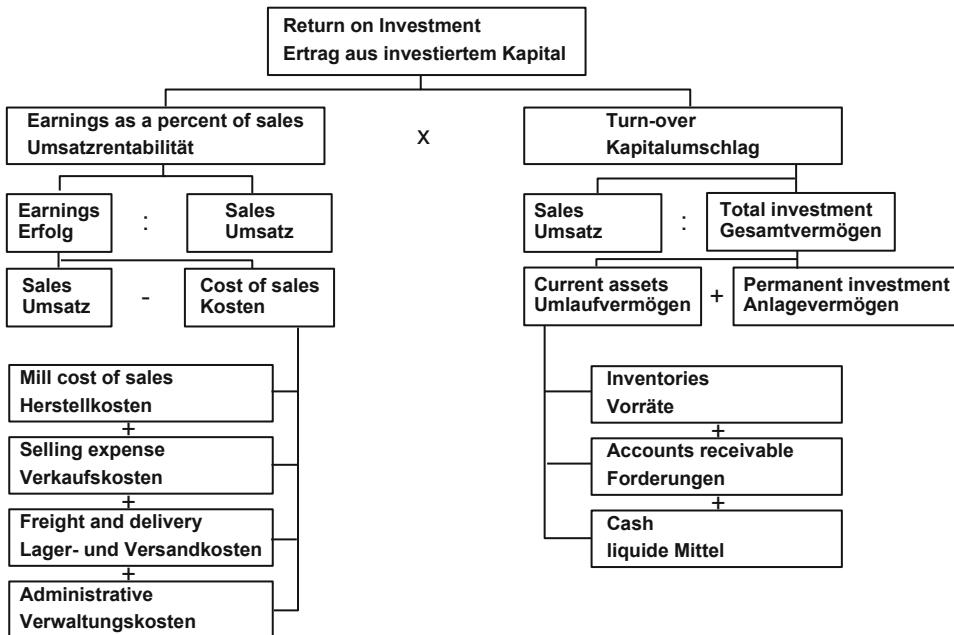
Der Business Case „Accelerator“ der App e-configureator ist in Abb. 3.16 dargestellt.

In dem oberen Teil der Abbildung sind die grundlegenden Annahmen bzw. Parameter überblicksartig aufgelistet. Die Berechnung des Business Cases ist in der Tabelle enthalten. In dem ersten Jahr fallen nur die Entwicklungskosten an. In den folgenden Jahren (Projektjahre) werden Beratungsprojekte angeführt, die mit Hilfe der App akquiriert werden konnten. Aus den Projekten werden Umsätze in der Beratung (CRM-Beratung) und im Bereich Cross-Selling durch Verkauf von CRM-Lizenzen und die um ein Jahr später anfallenden Wartungsgebühren je Lizenz generiert. In den Projektjahren fallen nur Mitarbeiterkosten an. Aus den Umsätzen und Kosten ergeben sich für das erste Entwicklungsjahr und die folgenden Projektjahre die jeweiligen Ergebnisse. Als KPI ergeben sich der Barwert mit 57 Tsd. € und der IRR mit 33 %.

---

### Beispiel

Der in Abb. 3.17 dargestellte Business Case „Web Service“ sieht das Geschäftsmodell des Vertriebs und Betriebs der mobilen Applikation über das Internet vor. Da bei den meisten Interaktionen kein direkter Kontakt zu den Kunden bzw. Anwendern besteht, sind die Cross-Selling-Effekte marginal (Zeilen CRM-Beratung, Cross-Selling). Bei den Kosten fallen insbesondere die Weiterentwicklungs- und Wartungskosten für die App selbst an. Als KPI ergeben sich der Barwert mit 138 Tsd. € und der IRR mit 40 %.

**Abb. 3.15** Kennzahlensystem ROI

<b>e-configurator Accelerator</b>	<b>Annahmen</b> <ul style="list-style-type: none"> <li>ca. 20 Stadtwerke in 5 Jahren</li> <li>CRM2 Lizenz mit jeweils 10 Usern</li> <li>Berateraufteilung: 100 % Eigen</li> <li>Erlöse/Tag: 1.000 €</li> <li>Kosten/Tag: 500 €</li> </ul>					
<b>Projektjahr</b>	<b>2012</b>	<b>2013</b>	<b>2014</b>	<b>2015</b>	<b>2016</b>	<b>2017</b>
Anzahl Versorger		3	5	4	4	4
<b>Umsatz</b>						
e-configurator (Lizenz und Wartung)		0	0	0	0	0
CRM Beratung		50	50	50	50	50
Cross Selling (CRM2 Beratung, Lizizenzen)		3	9	15	15	18
<b>Summe</b>		<b>53</b>	<b>59</b>	<b>65</b>	<b>65</b>	<b>68</b>
<b>Kosten</b>						
e-configurator (Wartung)		0	0	0	0	0
CRM Beratung		25	25	25	25	25
eigener Mitarbeiter		0	0	0	0	0
Fremdmitarbeiter		0	0	0	0	0
Sonstiges		0	0	0	0	0
<b>Summe</b>		<b>25</b>	<b>25</b>	<b>25</b>	<b>25</b>	<b>25</b>
zzgl. Investitionen (Web Service und Client)	-80					
<b>Ergebnis</b>	<b>-80</b>	<b>28</b>	<b>34</b>	<b>40</b>	<b>40</b>	<b>43</b>
<b>Barwert</b>	<b>57</b>					
<b>IRR</b>	<b>33%</b>					

**Abb. 3.16** Business Case e-configurator „Accelerator“

e-configuretor Web Solution		Annahmen				
		<ul style="list-style-type: none"> <li>▪ Vertrieb über Web als reiner Service</li> <li>▪ Ca. 200 Serviceuser in 5J 20% Wartung</li> <li>▪ Abzinsfaktor: 10,1%</li> <li>▪ QSF/Vertriebskosten: 10%</li> </ul>			<ul style="list-style-type: none"> <li>▪ ca. 5 Stadtwerke in 5 Jahren CRM2 Lizenz mit jeweils 10 Usern</li> <li>▪ Berateraufteilung: 100 % Eigen</li> <li>▪ Erlöse/Tag: 1.000 €</li> <li>▪ Kosten/Tag: 500 €</li> </ul>	
<b>Einführungsjahr</b>		<b>2012</b>	<b>2013</b>	<b>2014</b>	<b>2015</b>	<b>2016</b>
Anzahl Versorger		5	10	25	60	100
<b>Umsatz</b>						
e-configuretor (Lizenz und Wartung)		10	22	56	136	240
CRM Beratung		10	10	10	10	10
Cross Selling (CRM2 Beratung, Lizizenzen)		3	3	3	3	3
<b>Summe</b>		<b>23</b>	<b>35</b>	<b>69</b>	<b>149</b>	<b>253</b>
<b>Kosten</b>						
e-configuretor (Lizenz und Wartung)		10	16	23	24	45
CRM Beratung		5	5	5	5	5
eigene Mitarbeiter		0	0	0	0	0
Fremdmitarbeiter		5	5	10	15	20
<b>Summe</b>		<b>20</b>	<b>26</b>	<b>38</b>	<b>44</b>	<b>70</b>
zzgl. Investitionen (Web Service und Client)		-80				
<b>Ergebnis</b>		<b>-80</b>	<b>3</b>	<b>9</b>	<b>31</b>	<b>105</b>
<b>Barwert</b>		<b>138</b>				
<b>IRR</b>		<b>40%</b>				

**Abb. 3.17** Business Case e-configuretor „Web Service“

Geschäftsmodell des Business-Case-Szenarios „Softwarelizenz“ ist ein direkter und aktiver Vertrieb der mobilen Applikation e-configuretor als lizenzierte Software. Aufgrund der geschätzten hohen Absatzzahlen ergeben sich positive Effekte für vor- oder nachgelagerte Beratungsprojekte im Bereich CRM sowie Cross-Selling-Effekte für den zusätzlichen Verkauf von CRM Softwarelizenzen. Die Kosten für die Weiterentwicklung und Wartung der App sind entsprechend den geplanten Absatzzahlen höher als in den beiden anderen Szenarien. Die KPI für diese BC-Variante sind ein Barwert von 1.323 Tsd. € und ein IRR von 143 %. Rein auf die Analyse der KPI reduziert, müsste die Entscheidung für den Business Case „Softwarelizenz“ eine klare Sache sein (siehe Abb. 3.18).

Die quantitative Betrachtungsweise des Business Cases berücksichtigt nicht die qualitativen Faktoren. Ein optimaler Indikator oder Key Performance Measure enthält keine Aussage über die verbundenen Risiken und Chancen, wobei die Quantifizierung dieser Risiken oft sehr subjektiver Natur ist. Eine Möglichkeit, diese Punkte zumindest ins Kalkül zu ziehen, ist die rein verbale, qualitative Auseinandersetzung mit der Thematik. Geeignete Methoden hierfür sind die Chancen-Risiken-Analyse oder die SWOT-Analyse (engl.: Strengths – Weaknesses – Opportunities – Threats/deutsch: Stärken – Schwächen – Möglichkeiten – Drogungen).

e-configuretor Software		Annahmen					
		<ul style="list-style-type: none"> <li>▪ Aktiver Lizenzvertrieb</li> <li>▪ Ca. 200 Lizenznehmer in 5J</li> <li>20% Wartung</li> <li>▪ Abzinsfaktor: 10,1%</li> <li>▪ QSF/Vertriebskosten: 10%</li> </ul>				<ul style="list-style-type: none"> <li>▪ ca. 20 Stadtwerke in 5 Jahren CRM2 Lizenz mit jeweils 10 Usern</li> <li>▪ Berateraufteilung: 100 % Eigen</li> <li>▪ Erlöse / Tag: 1.000 €</li> <li>▪ Kosten / Tag: 500 €</li> </ul>	
<b>Einführungsjahr</b>		<b>2012</b>	<b>2013</b>	<b>2014</b>	<b>2015</b>	<b>2016</b>	<b>2017</b>
Anzahl Versorger			5	10	25	60	100
<b>Umsatz</b>							
e-configuretor (Lizenz und Wartung)		50	110	280	680	1.200	
CRM Beratung		50	70	90	120	200	
Cross Selling (CRM2 Beratung, Lizzenzen)		3	9	15	15	18	
<b>Summe</b>		<b>103</b>	<b>189</b>	<b>385</b>	<b>815</b>	<b>1.418</b>	
<b>Kosten</b>							
e-configuretor (Lizenz und Wartung)		20	25	55	90	150	
CRM Beratung		25	35	45	60	100	
eigener Mitarbeiter		0	0	0	0	0	
Fremdmitarbeiter		20	30	40	50	60	
<b>Summe</b>		<b>65</b>	<b>90</b>	<b>140</b>	<b>200</b>	<b>310</b>	
zzgl. Investitionen (Web Service und Client)		-80					
<b>Ergebnis</b>		<b>-80</b>	<b>38</b>	<b>99</b>	<b>245</b>	<b>615</b>	<b>1.108</b>
<b>Barwert</b>		<b>1.323</b>					
<b>IRR</b>		<b>143%</b>					

**Abb. 3.18** Business Case e-configuretor „Softwarelizenz“

Chancen	Risiko
<ol style="list-style-type: none"> <li>1. Kundenbindung</li> <li>2. Erschließen neuer Marktpotentiale</li> <li>3. Realisierung von Zusatzerlösen aus Dienstleistungen</li> <li>4. Zukunftssicherheit</li> </ol>	<ol style="list-style-type: none"> <li>1. Projektrisiken</li> <li>2. Organisatorische Risiken</li> <li>3. Technische Risiken</li> <li>4. Partner Risiken</li> <li>5. Geschäftsrisiken</li> </ol>

**Abb. 3.19** Chancen-Risiken-Analyse

► Die **Chancen-Risiken-Analyse** untersucht die Chancen und Risiken der externen Einflüsse auf ein Geschäftsmodell. Die Darstellung erfolgt in tabellarischer Form durch Gegenüberstellung der Chancen mit den Risiken (siehe Abb. 3.19).

► Die **SWOT-Analyse** stellt die ermittelten Chancen und Risiken externer Einflüsse in Bezug zu den Stärken und Schwächen, die in einer internen Analyse ermittelt werden. Durch diese Synthese sollen die Fragestellungen geklärt werden:

		Strengths	Weaknesses
		S1	W1
		S2	W2
		S3	W3
		S4	W4
		S5	.....
		.....	
Opportunities		O	S
O1		O1-S2	O1-W1
O2		O2-S3	O1-W2
O3		O3-S4	O2-W1
.....			O3-W4
		T	S
Threats		T	S
T1		T1-S2	T1-W3
T2		T2-S2	T2-W4
.....			

**Abb. 3.20** SWOT-Analyse

- Wie können wir unsere Stärken bei den gegebenen Chancen optimal einsetzen?
- Wie können wir durch unsere Stärken die Risiken minimieren oder vermeiden?
- Wie können wir bei den gegebenen Chancen unsere Schwächen in Stärken umwandeln?
- Wie können wir bei unseren Schwächen und den gegebenen Risiken die Gefahr eines Scheiterns reduzieren bzw. vermeiden?

Die Darstellung der SWOT-Analyse erfolgt in Matrix-Form (siehe Abb. 3.20).

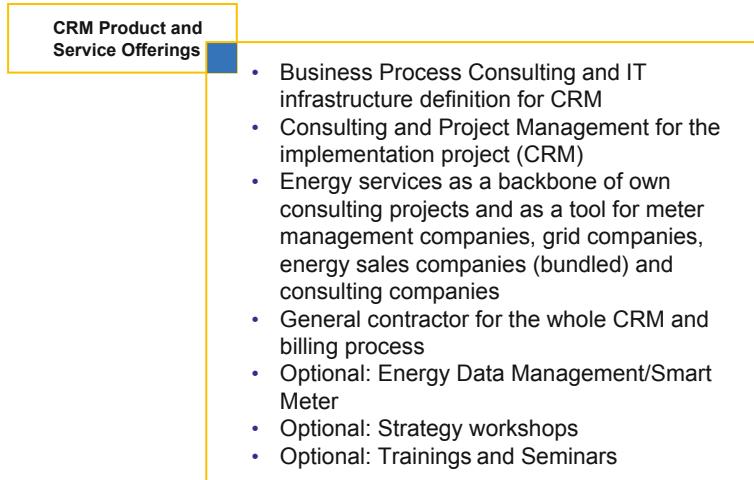
### Beispiel

In Abb. 3.21 ist die SWOT-Analyse für die mobile Applikation e-configurator dargestellt. Einigen der externen Chancen und Drohungen sind mehrere Stärken und Schwächen zugeordnet.

- Die SWOT-Analyse ergibt dann optimale Ergebnisse, wenn jeder Chance und jeder Drohung der externen Analyse in jedem Matrixfeld (Chancen-Stärken, Chancen-Schwächen, Drohungen-Stärken und Drohungen-Schwächen) mindestens eine passende Strategie zugeordnet ist (siehe Abb. 3.20 und 3.21). Für das Matrixfeld Drohungen-Schwächen ist hierbei auch der Aspekt, wie die Schwäche für diese Drohung ggf. in eine Stärke gewandelt werden kann, von existenzieller Bedeutung.

Strengths	Weaknesses
<b>S1:</b> Branchen Know-how <b>S2:</b> Markt Know-how <b>S3:</b> Reputation des Unternehmens <b>S4:</b> Wettbewerbsvorteil im EVU Umfeld <b>S5:</b> Methoden Know-how <b>S6:</b> CRM Beratungs Know-how	<b>W1:</b> Ressourcenmangel <b>W2:</b> Know how der Branchenprozesse <b>W3:</b> Zeitdruck <b>W4:</b> Fehlende Organisation <b>W5:</b> Lokaler Start <b>W6:</b> Finanzielle Ausstattung
Opportunities	OW
<b>C1:</b> Markt fordert integrierte Lösungen <b>C2:</b> Abrechnungssysteme zu statisch <b>C3:</b> IT und Beratungspartner gesucht <b>C4:</b> Hype Thema	<b>C1-S1-S2-S6:</b> Partnerschaft mit CRM2 <b>C2-S5:</b> e-Configurator dynamisch <b>C3-S1-S2-S3-S4-S5-S6:</b> Partnerschaft anbieten <b>C4-S3:</b> Public Relation und Marketing
Threats	TW
<b>T1:</b> Keine Mitarbeiter am Markt verfügbar <b>T2:</b> Marktreife erst in einigen Jahren <b>T3:</b> Angst der EVUs vor Innovation <b>T4:</b> Aufkommen von Nachahmern	<b>T1-W1-W3-W4:</b> Nicht Einhalten von zugesagten Prioritäten (z.B. Ressourcen) <b>T2-W5:</b> Testmarkt aufbauen <b>T3-W2-W5:</b> EVU Partner überzeugen <b>T4-W1-W2-W3-W4-W6:</b> Keine Projekte

**Abb. 3.21** SWOT-Analyse e-Configurator



**Abb. 3.22** Produkt- und Serviceportfolio

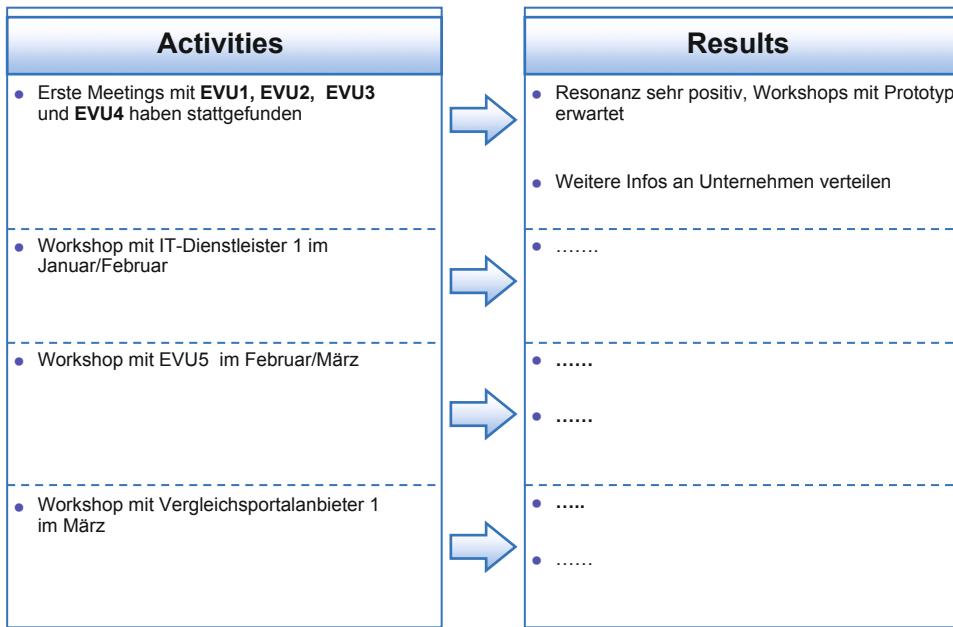


**Abb. 3.23** Partner

Für den Business Case können optional noch das aktuelle Produkt- und Serviceportfolio und die Auflistung der bestehenden Partnerschaften in den Bereichen Produkt und Services angeführt werden.

### Beispiel

Das neben der geplanten App e-configurator vorhandene Produkt- und Serviceportfolio ist in der Abb. 3.22 und die vorhandenen Partner sind in Abb. 3.23 aufgelistet.



**Abb. 3.24** Aktivitäten und Ergebnisse

Konkret auf die Applikation bezogene, schon durchgeführte bzw. geplante Aktivitäten und deren Ergebnisse werden abschließend dargestellt.

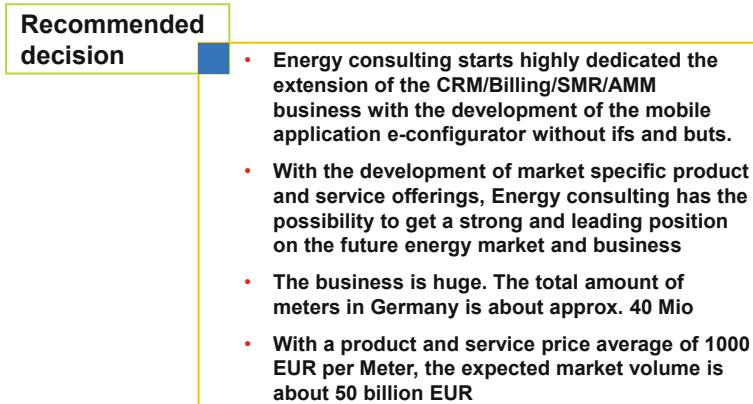
#### Beispiel

Für die App e-configurator wurden gemeinsam mit Energieversorgungsunternehmen (EVU) und anderen potenziellen Anwendern/Kunden schon erste Workshops durchgeführt bzw. befinden sich in Planung (siehe Abb. 3.24).

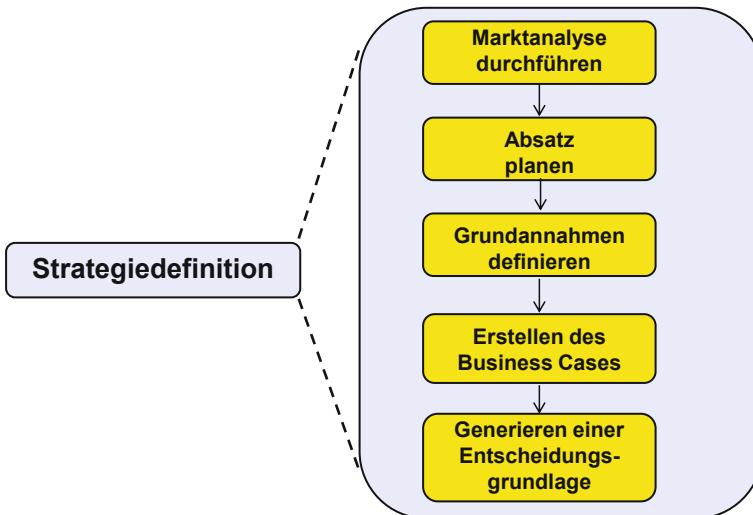
Das Strategiedefinitionsteam erstellt auf Grundlage des Business Cases eine Entscheidungsgrundlage, die den potenziellen Sponsoren ein oder mehrere Alternativen offeriert. Ziel sollte es sein, dass in dem Workshop mit der finalen Präsentation des Business Cases eine Entscheidung gefällt wird oder zumindest das weitere Vorgehen definiert wird.

#### Beispiel

Den Entscheidungsträgern des Unternehmens Energy Consulting wurde auf Basis des erstellten Business Cases ein Investment in die App e-configurator empfohlen (siehe Abb. 3.25). Dadurch wird die Marktpositionierung in der Energiewirtschaft entscheidend gestärkt und Cross-Selling-Projekte in den Bereichen CRM (Customer



**Abb. 3.25** Entscheidungsempfehlung



**Abb. 3.26** Phasen der Strategiedefinition

Relationship Management), SMR (Smart Meter Reading) und AMI/AMM (Advanced Metering Infrastructure / Advanced Metering Management) ermöglicht.<sup>10</sup>

Damit besteht die Phase der Strategiedefinition aus den Schritten (siehe Abb. 3.26):

1. Der Analyse des Marktes
2. Der Planung der Absatzzahlen

<sup>10</sup> Smart Meter Reading ist ein Begriff für sämtliche Produkte und Prozesse im Bereich elektronischer Stromzähler (Smart Meter). Advanced Metering Infrastructure beinhaltet die Informations- und Kommunikationsinfrastruktur für den Betrieb von Smart Metern. Siehe hierzu: Aichele 2012b und Aichele und Doleski 2013.

3. Der Definition der Grundannahmen
4. Der Erstellung des Business Cases
5. Der Generierung einer Entscheidungsgrundlage
6. Entscheidung für oder gegen die Fortführung des Projekts (Decision Gate Strategiedefinition)

### 3.1.3 Strategieumsetzung

Aufgaben der Strategieumsetzung sind die detaillierte Entwicklungsplanung, die Festlegung des zu präferierenden Geschäftsmodells, die Definition der Projektmanagementmethode und die Planung der Kosten, des Budgets und der Ressourcen.

Die Entwicklungsplanung beinhaltet die Konzeption der Applikation mit allen notwendigen Modellen und Erläuterungen zu den Modellen.

► **Modelle** repräsentieren ein vereinfachtes Abbild der betriebswirtschaftlichen Realität. Modelle sind zugänglicher, leichter manipulierbar, billiger, bekannter, vertrauter oder den jeweiligen Absichten des Modellsubjekts dienlicher und förderlicher als das Original.<sup>11</sup>

Als Modellierungsmethoden für das Fachkonzept einer App-Entwicklung sind folgende Methoden geeignet<sup>12</sup>:

- Darstellung von Geschäftsprozessen durch die Erweiterte Ereignisgesteuerte Prozesskette (eEPK) auf Basis der Informationssystemarchitektur ARIS (Architektur integrierter Informationssysteme)
- Darstellung von Geschäftsprozessen mit Business Process Model and Notation (BPMN), sogenannte BPMN-Prozesse
- Darstellung von Geschäftsprozessen durch Aktivitätsdiagramme auf Basis der Informationssystemarchitektur UML (Unified Modeling Language)
- Darstellung der Informationsflüsse durch Sequenzdiagramme auf Basis von UML
- Darstellung der Datenstruktur mit Entity-Relationship-Modellen (ERM)
- Darstellung der Systemstruktur durch Klassendiagramme und Anwendungsfalldiagramme auf Basis der Informationssystemarchitektur UML

► **Modellierungsmethoden** ermöglichen eine problembezogene und eine grafische Darstellungen der Realität in Form von Modellen. Sie enthalten die wesentlichen Beschreibungsobjekte zur Darstellung betriebswirtschaftlicher Zusammenhänge.<sup>13</sup>

► Eine **Informationssystemarchitektur** bezeichnet die Konzeption und Definition der Struktur eines Informationssystems (sehr oft ein IT-System) sowie die für den Nutzer

<sup>11</sup> Vgl. Aichele 2012b, S. 79.

<sup>12</sup> Details zu den Modellierungsmethoden siehe Aichele 2012b, S. 79–114, und Oestereich 2001 und Allweyer 2009 und Stevens und Pooley 2003 und weitere Literatur.

<sup>13</sup> Vgl. Aichele 2012b, S. 78.

des Systems möglichen Interaktionen und schließlich die An- und Zuordnung sowie die Benennung der in dem System enthaltenen Informationseinheiten und Funktionen.

Eine mehr technische Sichtweise der Definition beschreibt Informationsarchitektur als eine spezielle Form der IT eines Unternehmens, die zur Erreichung ausgewählter Ziele oder Funktionen entworfen wurde.<sup>14</sup>

Je nachdem, ob die Erstellung der App ausschließlich mit internen Ressourcen oder auch mit externen Ressourcen durchgeführt wird, spricht man eher von einem Fachkonzept oder einem Pflichtenheft. In einem Pflichtenheft werden die Aufgabenbestandteile externer Dienstleister detailliert definiert. Das Pflichtenheft ist auch oft Grundlage eines Dienstleistungsvertrags. Bei einer ausschließlichen Inhouseentwicklung werden die Rahmenbedingungen, die Vorgaben und Detailanforderungen typischerweise in dem Fachkonzept vorgegeben. Aber auch hier können als Synonyme die Begriffe Pflichtenheft, Conceptual Design, Sollkonzept oder Feinkonzept Anwendung finden.

Die Hauptbestandteile eines Fachkonzepts sind:

- Ziel- und Aufgabenstellung der mobilen Applikation, Abgrenzung
- Vorgehensmodell, Projektmanagementmethode
- Geschäftsprozesse bzw. Abläufe der Applikation (UML-Aktivitätendiagramm, BPMN, eEPK oder Ablaufdiagramme<sup>15</sup>)
- Geschäfts- und Anwendungsvorfälle der Applikation (UML-Anwendungsdiagramm)
- Art der Applikation (Native, Hybrid, Webservice), Betriebssystem (Operation System, OS)
- Schnittstellen, Integration von Sensoren und Akten (UML-Sequenzdiagramm)
- Objekte, Module und Methoden des Systems (UML-Klassendiagramm)
- Zusammenspiel und Integration der einzelnen Komponenten

---

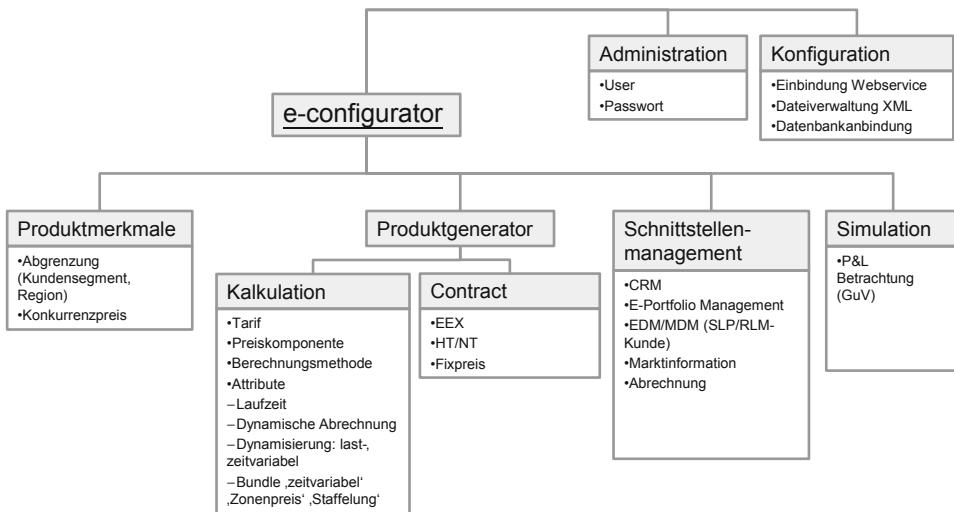
### Beispiel

Für die Erstellung der App e-configurator wurde das Vorgehensmodell des „Experimentellen Prototyping“ ausgewählt. Ziel des Experimentellen Prototyping ist das schnelle Erstellen eines lauffähigen Prototyps auf dessen Basis Erfahrungen in Pilotprojekten gesammelt werden können. Diese Erfahrungen durch die potenziellen Endnutzer sollen dann zu detaillierteren Systemspezifikationen führen. Als Projektmanagementmethode wurde eine Kombination von SCRUM und Extreme Programming verwendet. SCRUM, eine agile Projektmanagementmethode (wird auch als Vorgehensmodell der Systementwicklung bezeichnet), hat das Ziel, schnell, kostengünstig und qualitativ hochwertig Softwareapplikationen zu entwickeln. Dabei werden die geforderten Funktionen aus Anwendersicht formuliert und die Realisierung erfolgt in kurzen Intervallen (sogenannten Sprints). Extreme Programming (XP) stellt das Lösen

---

<sup>14</sup> Vgl. Laudon et al. 2009, S. 61.

<sup>15</sup> Z. B. Programmablaufpläne nach DIN 66001, siehe Wikipedia 2013.

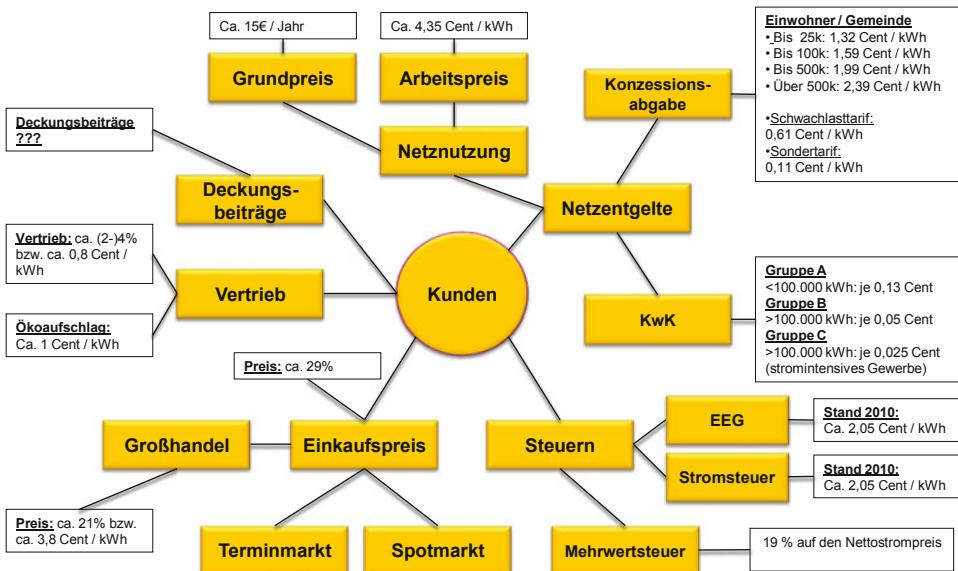


**Abb. 3.27** First Draft Klassendiagramm e-configureator

einer Programmieraufgabe in den Vordergrund und nähert sich dabei den Anforderungen des Kunden (der potenziellen Anwender) in kleinen Schritten. Die Modellierung der Abläufe erfolgte mit der Business Process Model and Notation, der Datenstrukturen mit der Entity-Relationship-Methode und der einzelnen Module/Funktionalitäten mit dem UML-Klassendiagramm. Zur Konzeption der Schnittstellen wurde das UML-Sequenzdiagramm genutzt. Die verbale Beschreibung wurde mit Hilfe von Stichwortprotokollen der einzelnen Workshops und Meetings auf das Notwendigste reduziert. In der Phase der Konzeption fanden mehrfach wöchentlich Teammeetings statt. Ein Extrakt des Fachkonzepts ist den Abbildungen Abb. 3.27 und 3.28 zu entnehmen.

### Beispiel

Das Klassendiagramm des e-configureator (Abb. 3.27) zeigt die Bestandteile (Module) der Applikationen auf. Dabei kann ein Modul eine Klasse bzw. ein Klassencluster darstellen. In der Administration werden die User verwaltet. Die Konfiguration ist für die Dateienverwaltung, die Einbindung des Webservices und die Datenbankanbindung verantwortlich. In dem Modul Produktmerkmale werden die Energietarife bzgl. der Region und des Kundensegmentes abgegrenzt. Der Produktgenerator besteht aus einem Kalkulationsteil und dem eigentlichen Vertrag (Contract). Der Vertrag berücksichtigt die Ergebnisse der Kalkulation und kann z. B. in Anbindung an aktuelle Börsenpreise (EEX: European Energy Exchange; Strombörse mit Sitz in Leipzig) festgelegt werden und auch als zeitvariabler Tarif mit definierten Preisen für den Tag und die Nacht (HT/NT-Tarif; Hochpreistarif für den Tag und Niedrigpreistarif für die Nacht). In dem Modul Schnittstellenmanagement werden die Einstellungen der Schnittstellen zu Customer-



**Abb. 3.28** Preiskomponenten Stromkunden (TK ohne LM)

Relationship-Management-Systemen (CRM), zu Energy-Data-Management-Systemen (EDM), zu Meter-Data-Management-Systemen (MDM), zu Marktinformationsportalen, zu den Abrechnungssystemen und zu dem Energy-Portfolio-Management-System des Energieunternehmens definiert. In dem Energy Portfolio Management werden die eingekauften Stromchargen des EVU administriert. Jedem Tarifprodukt müssen zur Kalkulation die mengenentsprechenden Einkaufsvolumen an Strom zugeordnet werden. Das Einkaufsportfolio hat damit direkten Einfluss auf den zu kalkulierenden Preis. In dem Baustein Simulation wird der Gewinn und Verlust (Profit & Loss, P&L) auf Basis unterschiedlichster Annahmen extrapoliert. Die Applikation soll für den Tarifkunden (TK) sowie für den Sondervertragskunden (SVK) die entsprechenden Produktkonfigurationen erlauben. Tarifkunden sind alle Kunden mit einem Jahresverbrauch kleiner 100.000 kWh, wobei auch für kleinere Gewerbetreibende ggf. ein SVK-Vertrag möglich ist (z. B. Bäckereien, Sonnenstudios, Handwerksbetriebe u. a.). In der Regel ist Strom bei Sonderverträgen günstiger zu beziehen. Die Berechnung der Preise für Tarifkunden basiert auf dem sogenannten Standardlastprofil (SLP), mit dem für alle Kunden aus dem Segment das Verbrauchsverhalten vorhergesagt wird. Aus diesem Grund werden diese Kunden auch als SLP-Kunden bezeichnet. Nur in wenigen Fällen werden für diese Kunden segmentierte SLP verwendet, d. h. einem Einpersonenhaushalt werden die gleichen Tarife wie einer Großfamilie angeboten, die beide auf demselben Standardlastprofil basieren. Durch die vermehrt digitale Erfassung der Stromverbräuche mit Smart Metern ist eine größere Diversifizierung der Strompreise bzw. Stromtarife auf Basis

von individuellen Lastprofilen oder segmentierten Standardlastprofilen zu erwarten.<sup>16</sup> Bei den Sondervertragskunden werden Stromverbrauch und auch Leistungswerte über die Registrierende Leistungsmessung (RLM) erfasst. Das sind in der Regel Stromzähler, die über Fernauslesung die 15 min Werte Verbrauch und Leistungsspitze an das Energieversorgungsunternehmen melden (z. B. über SIM-Karten und GSM). Aus diesem Grund werden die Sondervertragskunden auch als RLM-Kunden benannt.

Die App e-configurator hat in dem Kalkulationsmodul sämtliche Preiskomponenten für Strom zu berücksichtigen. In der Abb. 3.28 sind die Preiskomponenten für Tarifkunden ohne Leistungsmessung (nur Verbrauchswerte werden zurückgemeldet) aufgeführt. Die Hauptbestandteile sind dabei:

- der Einkaufspreis,
- die Netzentgelte: Grundpreis und Arbeitspreis für die Netznutzung (wird an das Stromnetzunternehmen abgeführt), die Konzessionsabgabe (wird an die Gemeinde oder Stadt, die dem Stromnetzunternehmen die Konzession zur Nutzung, Aufbau und Ausbau des Stromnetzes vertraglich überlassen hat, abgeführt) und die Abgaben für die Kraft-Wärme-Kopplung (KWK),
- Steuern: Stromsteuer, Erneuerbare Energien Abgabe (EEG, zur Querfinanzierung des Mehrpreises für Stromeinspeisungen aus erneuerbaren Energiequellen) und die Mehrwertsteuer,
- die Vertriebskosten
- und der zu erzielende Deckungsbeitrag.

Damit besteht die Phase der Strategiumsetzung aus den Schritten (siehe Abb. 3.29):

1. Auswahl des Vorgehensmodells
2. Definition der Projektmanagement-Methode
3. Erarbeitung und Definition der Konventionen für die Modellierung und für das Fachkonzept
4. Erstellung des Fachkonzepts
5. Entscheidung für oder gegen die Fortführung des Projekts (Decision Gate Strategiumsetzung)

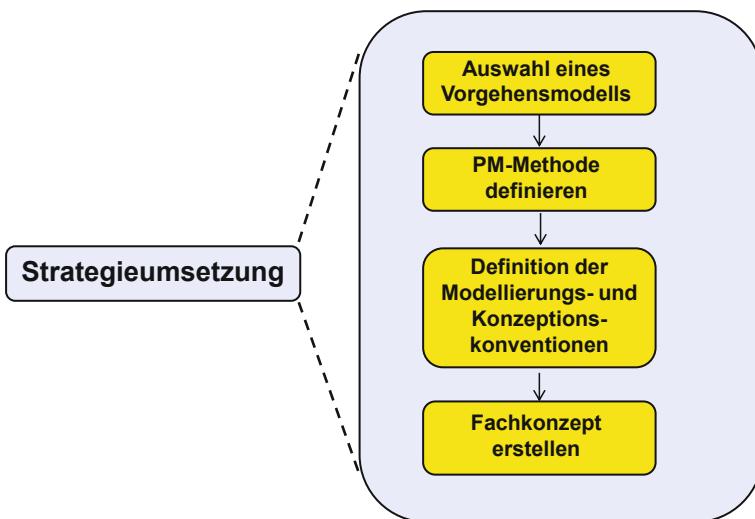
---

## 3.2 B2C und B2B Geschäftsmodelle für mobile Applikationen

Das App-Geschäftsmodell beschreibt die Art und Weise, wie mit der mobilen Applikation ein Nachfragemarkt bedient werden soll. Dabei muss nicht allein der Umsatz und Gewinn im Fokus stehen. Auch das kostenlose Überlassen der App in Form von Open-Source-Modellen kann aus unterschiedlichsten Beweggründen die Basis eines Geschäftsmodells sein.

---

<sup>16</sup> Siehe zu dynamischen Tarifen: Motsch 2012, S. 229–258.



**Abb. 3.29** Einzelschritte in der Phase Strategieumsetzung

► Unter einem **Geschäftsmodell für mobile Applikation** wird die Zielsetzung einer Vereinigung zur Erstellung und Verbreitung von Apps verstanden. Die Vereinigung kann dabei die Bandbreite von einer losen, temporären Zusammenarbeit bis zu einer betriebswirtschaftlichen Unternehmung annehmen. Zur Erfüllung dieser Zielsetzung müssen Organisationen und Strukturen definiert werden.

### 3.2.1 B2C-Geschäftsmodelle

Das Geschäftsmodell im B2C scheint relativ simpel zu sein. Ein App-Entwickler oder ein App-Unternehmen bieten dem Verbraucher oder Endkonsumenten mobile Applikationen über Online-Stores an. Der Endkonsument erhält beim Herunterladen der gewünschten App eine Lizenz und der Storeanbieter teilt sich mit dem Unternehmen oder Entwickler zu einem definierten Prozentsatz die Erlöse.

Dieses B2C-Geschäftsmodell wird als **B2C Lizenzmodell** definiert.

Aber neben den kostenpflichtigen Apps gibt es noch zahlreiche kostenlose Applikationen. Einige der kostenlosen Apps stellen limitierte Versionen der kostenpflichtigen, zu lizenzierenden App dar und sind damit Appetizer bzw. Werbeträger für die Vollversion (**B2C Presales-Modell**).

Andere Applikationen haben die Zielsetzung, Werbebotschaften an den Anwender zu übertragen (**B2C Advertising-Modell**). Durch Sammeln des Anwendernutzungsverhaltens wird versucht diese Werbung zu individualisieren und damit den Erfolg der Werbung zu erhöhen. Nach Einverständnis des Users kann sich die Datensammlung der Nutzung auf die eigentliche App beschränken oder ggf. auf das gesamte Mobile Device erstrecken. Hier

sind durch den Datenschutz (siehe Kap. 6) große Einschränkungen auferlegt. Je nachdem, wo der App-Anbieter (regulärer Store oder beliebige Web Page) lokalisiert ist, greift der nationale Datenschutz ggf. nicht. Auch durch das Einverständnis des Users bei der Installation der App bzw. bei Abgreifen attraktiver Add-ons kann dieser gewünschte Datenschutz jeweils wieder umgangen werden. Die App-Anbieter nutzen diese Daten zur kommerziellen Weitergabe an Dritte. Diese Drittunternehmen haben damit die Möglichkeit die Daten zur Weitervermarktung und Weiterverwendung auch außerhalb der Werbemöglichkeiten zu nutzen. Dieses Modell wird mit **B2C Data Gathering Modell** bezeichnet.

Die kostenlosen Zurverfügungstellung von Applikationen aus moralisch-ethischen, sozialen und anderen Gründen wird **B2C Open Source Modell** genannt. Dabei können die Anbieter öffentliche Organisationen, Unternehmen und Privatpersonen sein. Für Unternehmen und auch Privatpersonen eröffnet sich dadurch die Möglichkeit die eigene Reputation und Bekanntheit zu steigern und zu einem späteren Zeitpunkt in ein anderes B2C-Modell überzugehen. Öffentliche Organisationen nutzen solche Apps zur Informationsbereitstellung, für die Bildung und ggf. zur mobilen Unterstützung der öffentlich-rechtlichen Geschäftsprozesse (Datensammlung über mobile Formulare).

Natürlich sind die Modelle zum Teil auch binär oder mehrfach miteinander kombinierbar. So kann es durchaus Sinn machen ein B2C Lizenzmodell mit dem B2C Advertising-Modell zu kombinieren, ggf. zu einem reduzierten Lizenzpreis.

Die B2C-Geschäftsmodelle sind mittlerweile mehr oder weniger etabliert und ausgereift, da sich der gesamte App-Markt in diesem Segment entwickelt hat. Erfolgreiche Apps müssen genügend attraktiv sein, um den Konsumenten auch längerfristig zu binden. Dazu sollten sie nicht nur dem Spieltrieb und dem Informationsbedürfnis gewidmet sein, sondern auch Mehrwert für die digitale und analoge Welt liefern.

### 3.2.2 B2B-Geschäftsmodelle

Die Verwendung von Apps für die organisatorischen Anforderungen von Unternehmen hat mit einiger Verzögerung im B2C-Markt begonnen. Den Unternehmen fehlte es insbesondere an einer vollständigen und nachhaltigen Strategie für die Integration von Apps in die Geschäftsprozesse. Mobile Anwendungen sind auch in der Vergangenheit in den Prozessen der Unternehmen eingesetzt worden. Insbesondere in den folgenden funktionalen Bereichen findet ein mobiler Systemeinsatz statt:

- **Vertrieb (Außendienst):** Die Abbildung der CRM-Daten (Kundenstammdaten, Aktionen, Angebote, Auftragserfassung u. a.) ermöglicht den Vertriebsmitarbeitern die interaktive Angebotserstellung und Auftragserfassung. Als mobiles Device kommen Notebooks, Netbooks, Ultrabooks zum Einsatz. Die Verbindung zu den Backbone-systemen wird über GSM (Mobilfunk für Online- oder Einwahlverbindungen) und LAN/WLAN (für die vorbereitende und nachträgliche Synchronisierung der Daten) realisiert. Eine Onlineverbindung ist in der Regel nicht realisiert, die Synchronisierung der Daten erfolgt einmal oder mehrfach am Tag.

- **Logistik (Lagerwirtschaft):** Lieferungs- und Warendaten (z. B. Lieferungsavis) können von der externen oder eigenen Logistik an das zu beliefernde Werk (oder Produktionslinie) gemeldet werden. Diese Aktualität ist insbesondere für Just-in-time- oder Just-in-Sequence-Prozesse (generisch: Supply-Chain-Management-Prozesse) notwendig, damit die Produktionssteuerung entsprechend angepasst werden kann. Als mobile Devices kommen PDAs (Personal Digital Assistants), MDEs (Mobile Datenerfassung), Handhelds oder in Fahrzeugen fest eingebaute Erfassungsgeräte zum Einsatz. Dabei handelt es sich meist um robuste Geräte, die über Funk oder GSM/GPRS (oder aktuellere Mobilfunkübertragungstechnologien wie LTE) kommunizieren. Auch in den Lägern selbst können solche Devices die mobile Erfassung von Ein-, Um- und Auslagerungen unterstützen. Die Kommunikation erfolgt im Lager online über Funk, WLAN oder Mobilfunk.

Folgende Mobilfunkübertragungstechnologien werden in der Industrie verwendet:

**GSM:** Global System for Mobile Communications, je nach Datenübertragungskomponenten zwischen 14 bis 384 Kbit/s

**GSM/GPRS:** Erweiterung des GSM-Standards mit General Packet Radio Service, max. 171,2 Kbit/s

**GSM/Edge:** Erweiterung des GSM-Standards mit Enhanced Data Rates for GSM Evolution, 384 Kbit/s

**UMTS:** Universal Mobile Telecommunications System, 42 Mbit/s

**LTE:** Long Term Evolution, 300 Mbit/s

- **Instandhaltung und Service:** Der Instandhaltungsaufdienst oder der Serviceaufdienst erhalten über mobile Systeme Auftragsdaten mit Detailinformationen und können die Aufträge online oder zeitnah zurückmelden. Die Bandbreite der mobilen Geräte ist von Notebooks bis zu Handhelds möglich. Die Kommunikation kann online über Mobilfunk oder zu mehreren (beliebigen oder definierten) Zeitpunkten über LAN/WLAN erfolgen.
- **Weitere Funktionen:** Der Einsatz von mobilen Systemen erfolgt fallweise auch in den Prozessen Supplier Relationship Management (z. B. in Einkaufsverhandlungen) und Quality Management (z. B. bei Lieferantenaudits). Dabei handelt es sich um Einzelfallanwendungen, die über mobile Devices (Notebook) und Einwahlkommunikationsverbindungen (Mobilfunk und VPN-Einwahl; Virtual Private Network) die Nutzung der Backbone-Applikationen ermöglicht. Im engeren Sinne handelt es sich hierbei nicht um eine mobile Applikation, einzig das Device ermöglicht einen mobilen Zugriff auf die zentrale Anwendung.

Was unterscheidet aber diese Anwendungen von mobilen Apps? Diese Anwendungen sind dediziert und nur auf proprietärer Hardware anwendbar. Sie optimieren Teilschritte bzw. Teilfunktionen eines Geschäftsprozesses im Hinblick auf die Aktualität der Daten und Informationen und auf die Reaktionsschnelligkeit. Sie setzen keine Änderung oder Erweiterung der bisherigen Prozesse voraus.

Dahingegen sind Apps hardwareneutral und ermöglichen gekapselte Funktionalitäten für neue prozesserweiternde Zielsetzungen. Darüber hinaus verfügen Apps über einen eigenen Speicher und können auch offline angewendet werden. Apps ermöglichen im Sinne von „Bring your own device (BYOD)“ die Verwendung der individuellen Smartphones und Tablets der Unternehmensmitarbeiter. BYOD erlaubt den Unternehmen kostenreduzierte IT-Architekturen zu realisieren.

Ein potenzielles Geschäftsmodell ist die Information der Mitarbeiter über Neuerungen im Unternehmen (neue Produkte, neue Richtlinien, neue Strategien und Modelle). Dabei handelt es sich streng genommen nicht um ein B2B-Modell, sondern um ein B2E-Modell (Business to employee). In der Folge werden unternehmensinterne Anwendungsfälle wie Business-to-Employee für den App-Einsatz unter B2B subsumiert. Unternehmen können diese Informations-Apps über das Intranet einfach an die Mitarbeiter verteilen. Interaktionsmöglichkeiten geben ein Feedback über die Annahme dieser Anwendungen. Gamification und attraktive Visualisierung der Inhalte erhöhen die Anwendungsresonanz.

- Mit **Gamification** wird die Integration von digitalen Spielen oder Spielelementen in einen spielfremden Kontext bezeichnet. Die Spiele sollen die Motivation in der Nutzung der Applikation steigern.

Diese Anwendungsmöglichkeit wird als Geschäftsmodell **B2B Internal Information** bezeichnet.

Ein weiterer Anwendungsfall im Hinblick auf die Mitarbeiter ist die Nutzung von Apps für den Bereich eLearning. Auch hier sind mit dem Konzept des BYOD und kontextfremden Erweiterungen der Apps mit Spielen, Tipps für das tägliche Leben (z. B. Sport, Abnehmen, Freizeit, Energiesparen u. a.) Motivationssteigerungen der Mitarbeiter und Beschleunigungen der Lernerfolge das Resultat (Geschäftsmodell **B2B eLearning**).

Die Informationsverteilung per App ist auch ein adäquates Mittel, um Geschäftspartner, potenzielle Partner oder andere Organisationen über Neuerungen im Unternehmen zu informieren. Insbesondere dann, wenn die App über Zusatzfunktionalitäten verfügt, die den Adressaten attraktive Features anbieten, ist dieser Weg der Interaktion Erfolg versprechender als eine E-Mail-Werbung oder Links auf die eigenen Webseiten. Personalization und Gamification sind mit Apps einfacher möglich. Durch die bidirektionale Integration dieser Apps über QR-Codes<sup>17</sup> in Webmedien bzw. den eigenen Webseiten ist die Abgrenzung der medialen Inhalte komplex und ggf. ist die App nicht mehr als eigenständige mobile Anwendung erkennbar. Dieses Geschäftsmodell wird **B2B External Information** genannt.

Die Funktionsbereiche Vertrieb, Logistik, Instandhaltung, Service, ggf. Einkauf und Qualitätsmanagement, in denen mobile Anwendungen schon Realität sind, haben die Möglichkeit diese Anwendungen auch über mobile Apps zu realisieren. Dies hat zum einen den Vorteil der Nutzung des BYOD-Konzeptes und damit die Unabhängigkeit der norma-

<sup>17</sup> QR-Code (Quick Response) ist ein zweidimensionaler Code, über den u. a. ein Link zu Webseiten möglich ist oder das Herunterladen von Apps angestoßen werden kann.

lerweise notwendigen Bereitstellung von Hardware, Netware und Kommunikationstechnologie. Damit können erhebliche Kosteneinsparungen realisiert werden. Diese Substitution wird als Geschäftsmodell **B2B Mobile Substitution in Business Processes** definiert.

Interessanter sind aber die Anwendungsfälle, die bestehende Geschäftsprozesse durch den Einsatz von Apps verändern und dadurch optimieren. Die Geschäftsprozesse werden durch den App-Einsatz bereichert und erweitert. Das Geschäftsmodell **B2B Business Process Enrichment und Enlargement** erfordert im ersten Schritt das Erkennen der Potenziale. Dieser Ansatz erfordert das Reengineering der Prozesse und ist mit einem Business Process Reengineering für den Einsatz von mobilen Apps vergleichbar.

**Business Reengineering** oder **Business Process Reengineering (BPR)** bedeutet fundamentales Überdenken und radikales Redesign von Geschäftsprozessen. Das Ergebnis sind Verbesserungen in entscheidenden und messbaren Leistungsgrößen in den Bereichen Kosten, Qualität, Service und Zeit.<sup>18</sup>

Kernstück des Business Reengineering ist diskontinuierliches Denken, das überkommene Regeln und fundamentale Annahmen erkennt, die den heutigen betrieblichen Abläufen bzw. Geschäftsprozessen zugrunde liegen, und sich von ihnen abwendet<sup>19</sup>. Das Reengineering bzw. die Neukonzeption von Geschäftsprozessen wird durch folgende Methoden erreicht:

- Zusammenfassen mehrerer Aufgaben bzw. Aktivitäten,
- Delegation der Entscheidungsbefugnisse auf die Mitarbeiterebene bzw. die Ebene der Prozessausführenden,
- Gestalten der Prozesse im Hinblick auf Ereignisse und Ergebnisse,
- Gestaltung von Prozessvarianten,
- Durchführen der Aktivitäten am Ort ihres Auftretens,
- Reduktion des Überwachungs- und Kontrollaufwandes und damit Konzentration auf die wertschöpfenden Aktivitäten,
- Reduktion des Abstimmungsaufwandes durch Limitierung der Prozessschnittstellen,
- Umfassende organisatorische Kundenorientierung und
- Integration der Vorteile der Dezentralisierung und der Zentralisierung in den Prozessen.<sup>20,21</sup>

Übertragen auf den Einsatz von mobilen Apps ist das folgende Vorgehensmodell adäquat:

1. Ist-Analyse und Modellierung der Geschäftsprozesse (z. B. mit der Business Process Modelling Notation)
2. Erkennen der Optimierungspotenziale und Eingrenzung auf die Potenziale für den Einsatz von Apps

---

<sup>18</sup> Vgl. Hammer und Champy 1994, S. 48.

<sup>19</sup> Vgl. Hammer und Champy 1994, S. 13.

<sup>20</sup> Vgl. Hammer und Champy 1994, S. 71–89.

<sup>21</sup> Siehe zu BPR auch Aichele 2012a, S. 24–35.

- 
3. Konzeption der Apps
  4. Realisierung der Apps
  5. Organisatorische Integration der Apps in die Geschäftsprozesse

Ggf. sind durch dieses Vorgehen nur Optimierungen im Sinne von Kaizen oder Continuous Process Improvement möglich. Besser sind vorgeschaltet oder parallel die Schritte:

- Analyse der Unternehmensstrategie, Unternehmenszielsetzung und des Geschäftsmodells (z. B. mit Business Object Management; BOM)<sup>22</sup>
- Erkennen der App-Potenziale im Umfeld des eigenen Geschäftsmodells (hier ist der Einsatz von Kreativitätstechniken wie Brainstorming, Mindmaps oder das Vorgehensmodell des Design Thinking sinnvoll)
- Konzeption der App-basierten Geschäftsprozesse

Eingebunden in das oben angeführte Vorgehensmodell gestaltet sich die adaptierte Version wie folgt:

1. Analyse der Unternehmensstrategie, Unternehmenszielsetzung und des Geschäftsmodells
2. Erkennen der App-Potenziale im Umfeld des eigenen Geschäftsmodells
3. Istanalyse und Modellierung der Geschäftsprozesse (z. B. mit der Business Process Modelling Notation)
4. Erkennen der Optimierungspotenziale und Eingrenzung auf die Potenziale für den Einsatz von Apps
5. Konzeption der App-basierten Geschäftsprozesse
6. Konzeption der Apps
7. Realisierung der Apps
8. Organisatorische Integration der Apps in die Geschäftsprozesse

In der Branche Facility-Management bieten spezielle Apps die Möglichkeit der dezentralen Fernsteuerung von elektronischen Geräten über Akteure. So kann z. B. ein Facility-Manager die Heizungstemperaturen für alle von ihm zu betreuenden Liegenschaften einfach über sein Smartphone regeln. Diese Remotesteuerungsmöglichkeit ist auch für Fahrstühle, Licht, Klimaanlagen, Jalousien und Rollläden, Türent- und Türverriegelungen, Alarmanlagen, Kühlgeräte und das An- und Abschalten elektronischer Verbraucher vorhanden. Der Facility-Manager empfängt über in die elektronischen Verbrauchern integrierten Sensoren Zustandsmeldungen der kritischen elektronischen Geräte. Die Kunden des Facility-Management-Unternehmens, z. B. die Liegenschaftsverwaltungen und auch deren Kunden, z. B. die Mieter von Wohnungen oder Büroeinheiten, können über

---

<sup>22</sup> Siehe zu BOM auch Aichele 2006, S. 201–215.

Kunden-Apps mit dem Facility-Manager kommunizieren bzw. Service- und Instandhaltungsmeldungen absetzen. Dies kann durch die Nutzung der eigenen Smartphones jederzeit und direkt vor Ort erfolgen.

Unternehmen aus dem Bereich Facility-Management, die ihren Kunden und deren Kunden diese Möglichkeiten offerieren, bieten echten Mehrwert im Energie- und Gerätemanagement und im Komfort gegenüber ihren Mitbewerbern an.

---

### 3.3 App-Projekte und Projektmanagement

Die Gestaltung der Projekte und des Projektmanagements zur Entwicklung der mobilen Applikationen sind abhängig von dem verfolgten Geschäftsmodell (B2B, B2C, Marketplaces u. a.).

Projekte zeichnen sich durch bestimmte Merkmale aus und sind mit gewissen Risiken und Erfolgsfaktoren verbunden. Sie sind einmalige Vorhaben, die von der Tagesroutine abheben und sich nicht ständig wiederholen. Außerdem ist ein Projekt zeitlich begrenzt durch definierte Start- und Endtermine. Ein Projekt hat klar definierte Ziele. Es muss genau festgelegt sein, was erreicht werden soll und wie dies zu geschehen hat. Des Weiteren handelt es sich bei Projekten um komplexe Vorhaben die unterschiedliche Techniken und Methoden zur erfolgreichen Durchführung erfordern. Es sind zum Teil neuartige und unbekannte Probleme zu lösen und Projekte haben daher ein besonderes Risiko. Ferner steht Projekten ein bestimmtes Budget zur Verfügung, das nicht überschritten werden darf.<sup>23</sup>

App-Entwicklungsprojekte sind eher kurzfristige, weniger strukturerfordernde Softwaregenerierungsvorhaben und von immaterieller Natur. Insofern sind weniger die strukturierten Softwareentwicklung-Vorgehensmodelle wie das Wasserfallmodell, das Spiralmodell oder das V-Modell geeignet, sondern eher aktivitätsorientierte und kurzfristig adaptierbare Vorgehensmodelle wie SCRUM oder Extreme Programming. Diese führen auch zur schnellen Erstellung von Prototypen und der iterativen Weiterentwicklung.

Software wird nie ganz fertig. Es bleibt immer noch etwas zu ändern, zu verbessern. Und wenn Änderungen nicht zwingend notwendig sind, bleiben sie oft trotzdem wünschenswert – so lange, bis die Software wieder aus dem Verkehr gezogen wird. Eine Vielzahl von Änderungen bewirkt natürlich, dass sich das Programm immer mehr vom ursprünglichen Konzept entfernt. Gerade wenn die Applikation erfolgreich ist und deswegen ständig weiterentwickelt wird, besteht diese Gefahr.

Aus diesen Gründen ist es sinnvoll, sich nach geeigneten Methoden umzusehen, die die Komplexität beherrschbar machen, den Zerfallsprozess verzögern und trotz strukturzerstörender Änderungen und Weiterentwicklungen dabei helfen, die Qualität und Zuverlässigkeit der Software aufrechtzuerhalten.<sup>24</sup>

---

<sup>23</sup> Vgl. Aichele 2006, S. 25.

<sup>24</sup> Vgl. Oestreich 2001, S. 17.

Projekte zum Bau eines fossil befeuerten Kraftwerks und zur Realisierung eines Softwaresystems haben vieles gemeinsam. Beide erfordern Input, benötigen Menschen, die das Projekt realisieren und haben ein definiertes Ziel. Beide werden in spezifizierten, diskreten Schritten, die teilweise parallel und oft auch sequentiell durchgeführt werden müssen, realisiert. Die Methoden und Tools des Projektmanagements sind grundsätzlich die Gleichen. Der wesentliche Unterschied liegt in der Einmaligkeit und Immateriellität des Softwareprojektes. Während Kraftwerke in gleicher oder ähnlicher Form schon seit Langem gebaut und konstruiert werden, der Fortschritt visuell und auch sensorisch von der Haptik nachvollzogen werden kann, entzieht sich die Software weitgehend einer einfachen Kontrollierbarkeit. Insofern müssen die Methoden zum erfolgreichen Projektmanagement von IT Projekten mehr Parameter und Einflüsse berücksichtigen. Ohne weiteres können diese Methoden aber für Projekte in anderen Bereichen eingesetzt werden. Umgekehrt gilt das in der Regel nicht.<sup>25</sup>

► **Definition der DIN 69901** Projekte sind Vorhaben, die im Wesentlichen durch Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet sind, wie z. B. Zielvorgabe, zeitliche, personelle oder andere Begrenzungen, Abgrenzung gegenüber anderen Vorhaben und eine projektspezifische Organisation.<sup>26</sup>

Die Eigenschaften von Projekten sind:<sup>27</sup>

- Besondere Bedeutung
- Quantitative und qualitative Zielvereinbarung
- Komplexität
- Umfang
- Interdisziplinarität, bereichsübergreifende Zusammenarbeit
- Einmaligkeit
- Endlichkeit, zeitliche Befristung
- Abgrenzung gegenüber anderen Vorhaben
- Neuartigkeit
- Unsicherheit
- Begrenzte Ressourcen
- Projektspezifische Organisation
- Risiko.

Zusammenfassend kann ein Projekt als ein Vorhaben bezeichnet werden, dessen Ablauf (zumindest weitgehend) einmalig ist, dessen Struktur eine gewisse Komplexität aufweist und dessen festgelegte Zielsetzung in vorgegebener Zeit und in einer definierten Qualität mit gegebenen Ressourcen zu erreichen ist.<sup>28</sup>

<sup>25</sup> Vgl. Aichele 2006, S. 23.

<sup>26</sup> Siehe DIN 69901, Projektmanagement.

<sup>27</sup> Der Begriff Projekt ist aus dem lateinischen „pro-jacere = nach vorne“ abgeleitet.

<sup>28</sup> Vgl. Aichele 2006, S. 29–30.

Der Begriff Management besitzt viele unterschiedliche Definitionen.<sup>29</sup> Im Wesentlichen beinhaltet der Begriff jedoch einen Realisierungsprozess, der über die Phasen

- Planung,
- Organisation,
- Durchführung,
- Verfolgung und Steuerung

mit dem Einsatz von Menschen (institutionalisierte Führung) zur Formulierung und Erreichung von Zielen führt.

Eine adäquate Projektorganisation und ein performantes Projektmanagement sind für App-Entwicklungsprojekte von immenser Bedeutung. Dabei reicht es nicht aus die Methoden, Techniken und Tools für das Projektmanagement anwenden zu können. Wesentlich bedeutender ist ein intelligentes Projektmanagement, d. h. die Projektziele kommunizieren zu können, die Unternehmens- und Projektmitarbeiter überzeugen und führen zu können, rechtzeitig und prospektiv Entwicklungstendenzen des Projekts zu erkennen, diese Tendenzen hinsichtlich der Projektziele permanent anzupassen, empathisch mit dem Projektsponsor und den Projektmitarbeitern umgehen zu können und natürlich die Projektziele unter den gegebenen Rahmenbedingungen zu erreichen.

Das Projektmanagement ist ein Leitungs- und Organisationskonzept, mit dem die vielen sich teilweise gegenseitig beeinflussenden Projektelemente und das Projektgeschehen nicht dem Zufall oder der Genialität einzelner Personen überlassen, sondern gezielt zu einem festen Zeitpunkt herbeigeführt werden (etwas entwerfen und geplant nach vorne bringen, etwas unternehmen, um zielgerichtet ein Vorhaben zu erreichen). Es beinhaltet die Gesamtheit von Führungsaufgaben, Organisation, Techniken und Mitteln für die Abwicklung eines Projektes. Hierzu wird im Regelfall eine temporäre Institution zur Wahrnehmung der Aufgaben implementiert (siehe Abb. 3.30).<sup>30</sup>

Die Projektleitung und das Projektmanagement sind die entscheidenden Faktoren für den Projekterfolg („Decisive Factors“). Das Projektmanagement muss neben dem erforderlichen Wissen über Vorgehensweisen, Methoden und Tools im Bereich mobile Applikationen auch über die richtige Kombination von menschlichen Eigenschaften verfügen. Das Projektmanagement sollte motivierend und integrativ sein, ausgleichend und antreibend, kommunikativ, aber informativ, freundlich, aber bestimmt, demokratisch, aber mit der für den Projekterfolg notwendigen Autorität sein. Die Projektleitung muss die richtigen Mitarbeiter mit den notwendigen Qualifikationen und menschlichen Eigenschaften auswählen und die richtigen Maßnahmen für die Teambildung ergreifen. Das Projektmanagement muss die Mitarbeiter führen können, aber auch Arbeiten de-

---

<sup>29</sup> Der Begriff Management ist aus dem italienischen „menaggiare = handhaben, bewerkstelligen“ abgeleitet.

<sup>30</sup> Vgl. Aichele 2006, S. 30–31.



**Abb. 3.30** Projektmanagement. (vgl. Aichele, 2006, S. 31)

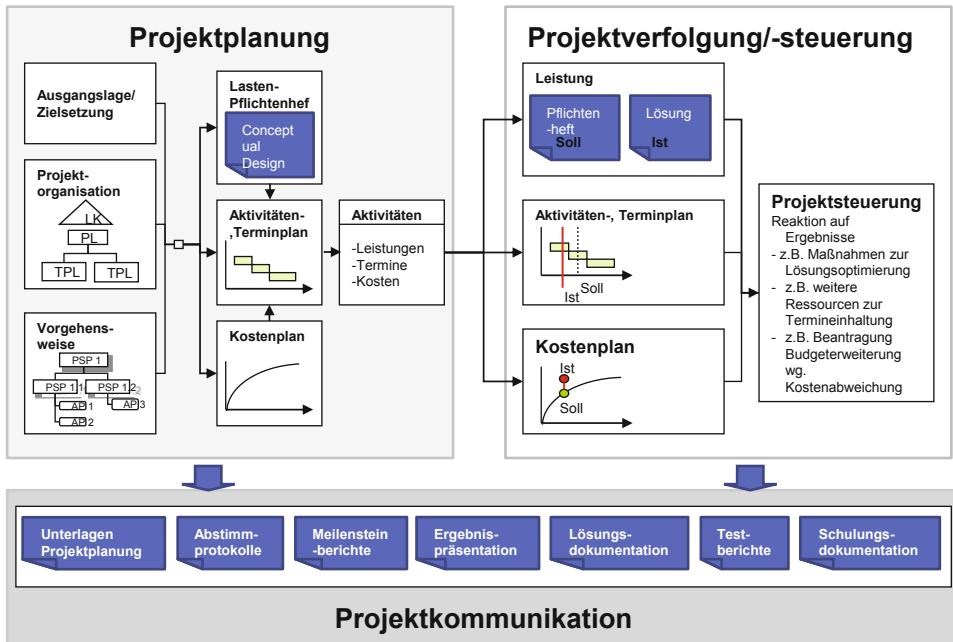
legieren können und damit den Mitarbeitern die entsprechend notwendigen Freiräume zugestehen.<sup>31</sup>

Projekte müssen organisiert werden, damit die Komplexität in realisierbare Teilprojekte bzw. Aufgabenpakete zerlegt wird (siehe Abb. 3.31). Zur Projektorganisation gehören die folgenden Aufgaben:

- Definition des Projekts (verbale Beschreibung)
- Benennung des Projektziels
- Qualifizierung des Projektziels (eventuell in Form eines Businessplans)
- Quantifizierung des Projektziels und der Projektunterziele (Business Case und/oder Kennzahlen)
- Definition der Projektvorgaben
- Strukturierung der Projektaufbauorganisation (Sponsoren, Lenkungsausschuss, Projektleitung, Teilprojektleiter, Projektmitarbeiter)
- Erstellung der Projektplanung, d.h. Planung der einzelnen Aufgabenpakete und Aktivitäten und Zuweisung der einsetzbaren Ressourcen wie Personal, Sachmittel, Finanzmittel (Projektstruktur- und Netzplanung)
- Vorgabe der Projektmethoden, -techniken und -tools
- Definition der Projektdokumentation
- Formalisierung und Regelung der Projektkommunikation<sup>32</sup>

<sup>31</sup> Vgl. Aichele 2006, S. 144.

<sup>32</sup> Vgl. Aichele 2006, S. 31–32.



**Abb. 3.31** Projektorganisation. (vgl. Aichele, 2006, S. 32)

Neben diesen allgemeingültigen Prinzipien ergeben sich für App-Entwicklungsprojekte einige Spezifika, die detailliert in Kap. 5 zum Vorgehens- und Validierungsmodell für die App-Entwicklung dargestellt werden.

In der Phase der **Initiierung** muss der Projektansatz definiert werden. Die Ausgangssituation (IT-Environment, Kundenstruktur, vorhandene eigene Ressourcen, Erfahrungen) muss realistisch und objektiv bewertet werden. In dieser Phase ist ggf. die Einbeziehung von externen Erfahrungsträgern notwendig. Die unterschiedlichen Akteure müssen ihre Rollen spezifizieren. Ein Businessplan mit integriertem Business Case sollte als Entscheidungsgrundlage erstellt werden.

Die Phase der **Planung** beinhaltet neben den Projektmanagementstandards wie die Projektstrukturplanung (PSP), Netzplanung, Ressourcen-, Kosten- und Budgetplanungen auch die Definition der technischen Spezifikationen (Geräte, Hardware, Systeme, Kommunikation) und die Planung der IT/TK-Infrastruktur.

In der Phase der **Umsetzung bzw. Realisierung** muss neben dem Projektcontrolling ein technisches Störfallmanagement (Incident Management) vorhanden sein.

## Literatur

- Aichele, C.: Intelligentes Projektmanagement. Kohlhammer, Stuttgart (2006)
- Aichele, C.: Kennzahlenbasierte Geschäftsprozessoptimierung, 2. Aufl. Gabler, Wiesbaden (2012a)
- Aichele, C.: Smart Energy, Von der reaktiven Kundenverwaltung zum proaktiven Kundenmanagement. Springer Vieweg, Wiesbaden (2012b)
- Aichele, C., Doleski, O.: Smart Meter Rollout. Springer Vieweg, Wiesbaden (2013)
- Allweyer, T.: Business Process Model and Notation: Einführung in den Standard für die Geschäftsprozessmodellierung, 2. Aufl., Norderstedt (2009)
- Gesetz über die Elektrizitäts- und Gasversorgung (Energiewirtschaftsgesetz – EnWG) vom 7. Juli 2005 (BGBl. I S. 1970, 3621), das durch Artikel 2 Absatz 66 des Gesetzes vom 22. Dezember 2011 (BGBl. I S. 3044) geändert worden ist, 2011
- Hammer, M., Champy, J.: Business Reengineering. Die Radikalkur für das Unternehmen. Campus, Frankfurt a. M. (1994)
- Hasso-Plattner-Institut, Universität Potsdam: Design thinking. [http://www.hpi.uni-potsdam.de/d\\_school/designthinking](http://www.hpi.uni-potsdam.de/d_school/designthinking) (2013). Zugegriffen 2. Juli 2013
- Laudon, K.C., Laudon, J.P., Schoder, D.: Wirtschaftsinformatik. Eine Einführung, 2. Aufl. Addison-Wesley, München (2009)
- Motsch, W.: Dynamische Tarife zur Kundeninteraktion mit einem Smart Grid. In: Aichele, C. (Hrsg.) Smart Energy, Von der reaktiven Kundenverwaltung zum proaktiven Kundenmanagement, S. 229–258. Springer Vieweg, Wiesbaden (2012)
- Oestereich, B.: Objektorientierte Softwareentwicklung. Analyse und Design mit der Unified Modeling Language, 5. Aufl. Oldenbourg Wissenschaftsverlag, München, (2001)
- Stevens, P., Pooley, R. UML – Softwareentwicklung mit Objekten und Komponenten. Pearson Studium, München (2001)
- Wikipedia: Programmablaufpläne, Programmablaufpläne nach DIN 66001. <http://de.wikipedia.org/wiki/Programmablaufplan> (2013). Zugegriffen 18. Sept. 2013

---

# Der professionelle Einstieg in die erfolgreiche App-Entwicklung

4

Marius Schönberger

*Fachliche und technologische Grundlagen der App-Entwicklung*

---

## Zusammenfassung

Mobile Endgeräte haben sich in kürzester Zeit als primäres Medium für den Internetzugang über alle Altersgruppen hinweg etabliert. In einer Welt voller intelligenter Smartphones, Tablet-PCs, Notebooks und weiterer internetfähiger Mobilgeräte sowie in Hinblick auf den Ausbau der Mobilfunknetze, schnellerer Breitbandverbindungen und WiFi-Netzwerke, wird die Nachfrage nach mobilen Anwendungen nur noch stärker. Das vorliegende Kapitel soll die fachlichen und technologischen Grundlagen der mobilen App-Entwicklung aufzeigen und damit einen Einstieg in die Thematik ermöglichen. Hierzu wird zunächst auf die Softwareentwicklung im Allgemeinen eingegangen, bevor im Anschluss Arten und Eigenschaften mobiler Endgeräte und Anwendungen aufgezeigt werden. Das Kapitel endet mit der Vorstellung wesentlicher Instrumente und Werkzeuge zur mobilen Anwendungsentwicklung.

---

## 4.1 Softwareentwicklung und Softwarebegriff

In der zweiten Hälfte des 20. Jahrhunderts hat sich die Softwareentwicklung stufenweise zu einer Ingenieurdisziplin entwickelt und wurde erstmals 1968 unter dem Begriff „Software Engineering“ geprägt.<sup>1</sup> Im Vergleich zu klassischen Ingenieurdisziplinen wie bspw. Bau-

---

<sup>1</sup> Vgl. Naur und Randell 1969, S. 138–155.

---

M. Schönberger (✉)  
Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: marius.schoenberger@fh-kl.de

ingenieurwesen, Maschinenbau oder Elektrotechnik stellt die Softwareentwicklung damit eine sehr junge Ingenieurwissenschaft dar.<sup>2</sup>

Eine erste Definition des Begriffs Software Engineering findet sich im Standardglossar der Softwaretechnik-Terminologie wieder, welches durch das Institute of Electrical and Electronics Engineers (IEEE) im Jahr 1990 veröffentlicht wurde.<sup>3</sup>

► „**software engineering.** 1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. 2) The study of approaches as in 1).“

Eine Erweiterung und Spezialisierung dieser Bestimmung des Begriffs Software Engineering erfolgte durch die Autoren Hesse, Keutgen, Luft und Rombach.<sup>4</sup>

► **Software Engineering** „Fachgebiet der Informatik, das sich mit der Bereitstellung und systematischen Verwendung von Methoden und Werkzeugen für die Herstellung und Anwendung von Software beschäftigt.“

Der Autor Helmut Balzert stützt sich mit seiner Begriffsbestimmung auf die zuvor genannten Definitionen und erweitert diese um das Einsatzgebiet des Software Engineering:<sup>5</sup>

► „**Software Engineering** beschreibt die zielorientierte Bereitstellung und systematische Verwendung von Prinzipien, Methoden und Werkzeugen für die arbeitsteilige, ingenieurmäßige Entwicklung und Anwendung von umfangreichen Softwaresystemen.“

Aus den bereits genannten Definitionen kann der Begriff Software Engineering folgendermaßen zusammengefasst werden:

► „**Software Engineering** beschäftigt sich mit der Entstehung und Entwicklung von Software sowie dem Betrieb von Softwaresystemen unter Verwendung allgemeingültiger Methoden, Verfahren und Werkzeuge der Softwareentwicklung.“

Die Bezeichnung „Software“ ist in der heutigen Zeit zu einem alltäglichen Begriff geworden, der hauptsächlich mit Text- und Bildbearbeitungsprogrammen oder Spielen für den privaten Computer in Verbindung gebracht wird. Ungeachtet dieser Tatsache werden Softwareprodukte immer häufiger zentraler Bestandteil komplexer elektronischer Geräte, die technische oder betriebswirtschaftliche Prozesse steuern oder unterstützen. Software ist

---

<sup>2</sup> Vgl. Aßmann et al. 2006, S. 105.

<sup>3</sup> IEEE 1990, S. 67.

<sup>4</sup> Hesse et al. 1984, S. 200–213.

<sup>5</sup> Balzert 2009, S. 17.

somit zu einem wichtigen Wirtschaftsfaktor geworden und nimmt nicht nur in Wirtschaft, Wissenschaft und Technik eine wichtige Stellung ein, sondern auch im Dienstleistungsbereich, Gesundheitswesen sowie an Schulen und Universitäten.<sup>6</sup> Nachfolgend wird für eine adäquate Auseinandersetzung mit dem Softwarebegriff zunächst die Definition von Software angegeben und Eigenschaften von Software aufgezeigt. Des Weiteren erfolgt eine detaillierte Klassifizierung des Softwarebegriffs.

In der Brockhaus Enzyklopädie ist der Begriff „Software“ wie folgt aufgelistet:<sup>7</sup>

► **Software** [dt. „weiche Ware“] nach DIN 44 300 die Gesamtheit oder Teil der Programme für Rechensysteme, wobei die Programme zusammen mit den Eigenschaften der Rechensysteme den Betrieb der Rechensysteme, die Nutzung der Rechensysteme zur Lösung gestellter Aufgaben oder zusätzliche Betriebs- und Anwendungsarten der Rechensysteme ermöglichen.“

An der Definition nach Brockhaus ist jedoch kritisch anzumerken, dass nicht nur ausschließlich Software für den Betrieb eines Rechensystems notwendig ist. Dies wird von Brockhaus jedoch suggeriert. Eine bessere Definition wird durch das IEEE gegeben. Software umfasst demnach Programme, Abläufe, Regeln sowie Dokumentationen und Daten, welche mit dem Betrieb eines Rechnersystems zu tun haben:<sup>8</sup>

► „**software**. Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.“

Eine allgemein fundierte und aktuelle betriebswirtschaftliche Abgrenzung des Begriffs Software wird von den Autoren Abts und Mülder gegeben:<sup>9</sup>

► „**Software** umfasst hierbei alle Produkte und Dienstleistungen, die eine sinnvolle Nutzung der Hardware überhaupt erst ermöglichen, also neben den Programmen z. B. die Anwendungsberatung, die Installationshilfe, die Dokumentation, die Schulung der Benutzer und die Wartung“

Historisch betrachtet wurde der Begriff Software erstmals 1957 als Kunstwort dem damals schon bestehenden Begriff Hardware gegenübergestellt.<sup>10</sup> Als Hardware werden Bauteile und Komponenten eines Rechensystems bezeichnet, welche eine physische Materialität

---

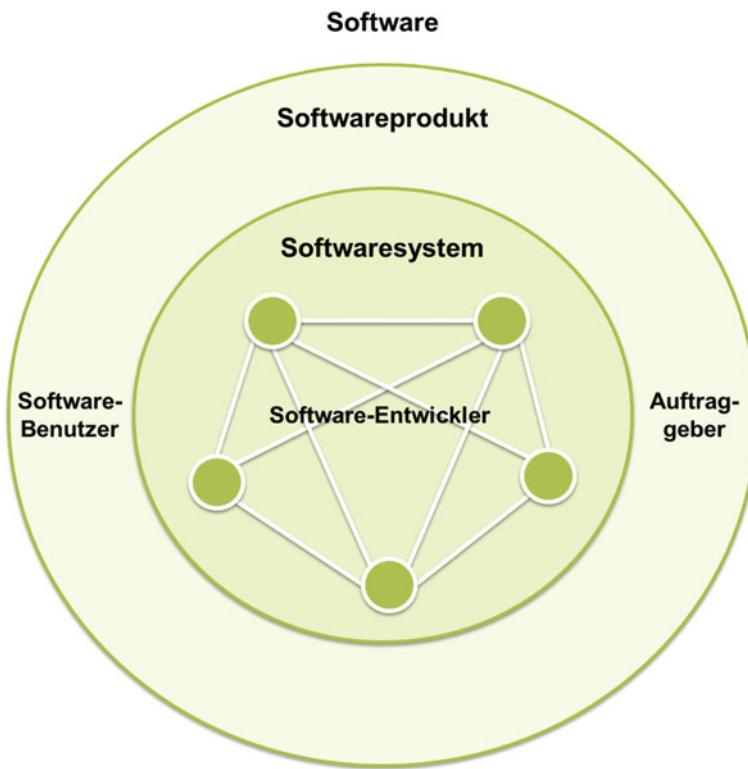
<sup>6</sup> Vgl. Pomberger und Pree 2004, S. 1.

<sup>7</sup> Brockhaus Enzyklopädie 2003, S. 818 f.

<sup>8</sup> IEEE 1990, S. 66.

<sup>9</sup> Abts und Mülder 2010, S. 57 f.

<sup>10</sup> Vgl. Ludewig und Licher 2010, S. 35.



**Abb. 4.1** Zusammenhang zwischen Software, -produkt und -system. (eigene Erstellung, in Anlehnung an Balzert 2009, S. 4)

besitzen. Software stellt dagegen jede Art digitaler Daten und Informationen dar, die mit Hardwaresystemen oder -komponenten interagieren können.<sup>11</sup>

In Zusammenhang mit dem Begriff Software werden weiterhin die Begriffe Softwaresystem sowie Softwareprodukt genannt.<sup>12</sup> Unter einem Softwaresystem werden Systeme verstanden, deren Systemelemente und -komponenten aus Software bestehen. Ein Softwaresystem mit Produktcharakter, welches für einen Auftraggeber ein in sich abgeschlossenes, mit Erfolg durchgeführtes Projekt darstellt, wird als Softwareprodukt bezeichnet.<sup>13</sup> Abbildung. 4.1 stellt den Zusammenhang zwischen den drei genannten Begriffen dar.

Aus den gegebenen Begriffsdefinitionen gehen Eigenschaften, welche sich auf die Entwicklung, Vermarktung und Wartung von Software beziehen, nicht oder nur be-

<sup>11</sup> Vgl. Hansen und Neumann 2009, S. 28.

<sup>12</sup> Vgl. Pomberger und Pree 2004, S. 3.

<sup>13</sup> Vgl. Balzert 2009, S. 3.

dingt hervor. Daher wird im Folgenden kurz auf weitere wichtige Softwaremerkmale eingegangen:<sup>14</sup>

- Software stellt ein immaterielles Produkt dar.
- Software unterliegt im Gegensatz zur Hardware keinem Verschleiß.
- Software altern.
- Software kann ohne Qualitätsverluste dupliziert oder vervielfältigt werden.
- Software ist leichter abänderbar als vergleichsweise ein technisches Produkt.
- Software ist schwer zu vermessen.

Je nach Untersuchungszweck erfolgt eine unterschiedliche Sichtweise auf den Begriff „Software“. Im Bereich der Wirtschaftsinformatik wird der Begriff, je nach der Nähe zur Hardware bzw. zur Anwendung, in Systemsoftware und Anwendungssoftware differenziert.<sup>15</sup> Die Systemsoftware, oder auch Basissoftware genannt, stellt maschinen- und hardwareorientierte Programme zur Verfügung und bildet somit die Schnittstelle zur Hardware. Die Systemsoftware wird hauptsächlich für eine spezielle Hardware oder Hardwarefamilie entwickelt und zielt darauf ab, diese zu steuern, zu verwalten und zu unterstützen sowie Anwendungen auszuführen.<sup>16</sup> Betriebssysteme, Übersetzungs- und Dienstprogramme sowie Protokolle und Treiber werden der Systemsoftware zugeordnet.<sup>17</sup>

Aufbauend auf der Hardware und der Systemsoftware stellt die Anwendungssoftware die Schnittstelle zum Benutzer dar. Auch als Applikationssoftware bezeichnet, zielt die Anwendungssoftware darauf ab, problem- und aufgabenorientierte Programme zur Verfügung zu stellen. Durch Verarbeitung und Bereitstellung relevanter Daten und Informationen erfolgt durch die Anwendungssoftware die Lösung unterschiedlicher Aufgaben des Nutzers.<sup>18</sup>

Je nach Aufgabenbereich erfolgt weiterhin eine differenzierte Sichtweise auf die Anwendungssoftware. Müssen Anwender gleiche oder identische Aufgabenbereiche in unterschiedlichen zeitlichen Abständen wiederholt durchführen, so handelt es sich hierbei um standardisierte Aufgaben, und die hierfür verwendete Anwendungssoftware wird als Standardsoftware bezeichnet. Zu diesen standardisierten Lösungen zählen unter anderem Anwendungen, wie z. B. Webbrowser, Standardbürosoftware, bspw. Text- und Bildbearbeitungsprogramme, sowie funktionsorientierte Standardsoftware, wie z. B. Kassen- und Bezahlsysteme.<sup>19</sup>

Erfolgt die Entwicklung einer Anwendungssoftware speziell auf den Bedarf und das Einsatzgebiet eines Benutzers, wird diese als Individualsoftware definiert. Individualsoftware

---

<sup>14</sup> Vgl. Lanninger 2009, S. 95; Balzert 2009, S. 9.

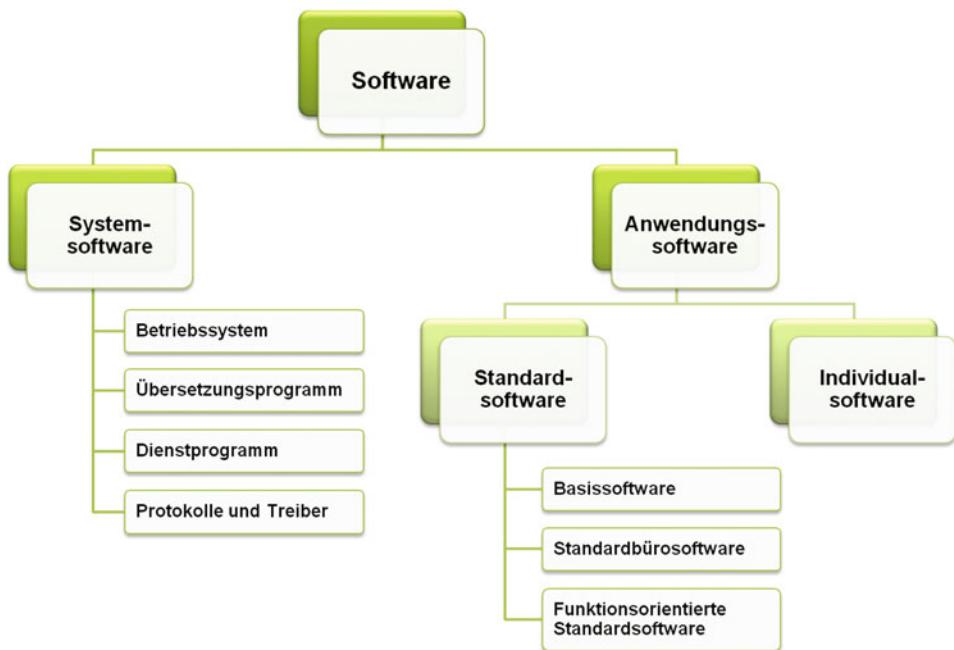
<sup>15</sup> Vgl. Kurbel 2013.

<sup>16</sup> Vgl. Lanninger 2009, S. 99.

<sup>17</sup> Vgl. Mertens et al. 2005, S. 21.

<sup>18</sup> Vgl. Diederichs 2004, S. 90; Balzert 2009, S. 5.

<sup>19</sup> Vgl. Mertens et al. 2005, S. 22.



**Abb. 4.2** Klassifizierung von Software. (eigene Erstellung, in Anlehnung an Mertens et al. 2005, S. 21)

kann entweder selbst durch eine eigene IT- oder Fachabteilung oder durch externe Softwareentwickler hergestellt werden und wird hauptsächlich auf einer eigenen Hardware- und Softwareumgebung angewendet.<sup>20</sup> Die Autoren Mertens et al., Lanninger und Balzert sprechen jeweils unabhängig voneinander von einem zunehmenden Trend hin zu Standardsoftware.<sup>21</sup> Balzert spricht explizit von einem Rückgang der Individualsoftware von bis zu fünf Prozent.<sup>22</sup> Grund hierfür sind u. a. hohe Entwicklungskosten sowie die Dynamik der Märkte, welche zu einem erheblichen Wartungsaufwand bestehender individueller Softwareprodukte führen.<sup>23</sup> Eine Studie der Bitkom zeigt jedoch, dass innerhalb der letzten Jahre, vor allem im Mobile-Device-Bereich, ein rasanter Anstieg bei der Entwicklung individueller Anwendungen zu verzeichnen ist.<sup>24</sup> Eine Übersicht über die Klassifizierung von Software wird durch Abb. 4.2 gegeben.

Die vorliegenden Begriffsdefinitionen, die Beschreibungen von Eigenschaften sowie die Klassifizierung von Software in System-, Anwendungs-, Standard- und Individualsoftware

<sup>20</sup> Vgl. Lanninger 2009, S. 109.

<sup>21</sup> Vgl. Mertens et al. 2005, S. 33; Lanninger 2009, S. 110; Balzert 2009, S. 14.

<sup>22</sup> Vgl. Balzert 2009, S. 14.

<sup>23</sup> Vgl. Mertens et al. 2005, S. 33; Lanninger 2009, S. 110.

<sup>24</sup> Vgl. Bitkom 2011.

---

zeigen, dass Software einen umfassenderen Begriff darstellt als bspw. das Computerprogramm. Auf eine detaillierte Beschreibung des Begriffs „Systemsoftware“ wird im weiteren Verlauf dieses Kapitels verzichtet. Ebenso erfolgt keine nähere Erläuterung der Komponenten von Standardsoftware.

---

## 4.2 Mobile Endgeräte

Mobile informations- und telekommunikationstechnische Systemlösungen (ITK-Lösungen) für den alltäglichen Einsatz werden durch die Miniaturisierung von Computertechnologie, eine weiträumige drahtlose Vernetzung und durch eine effektive mobile Stromversorgung ermöglicht und getrieben.<sup>25</sup> Im Wettbewerb um Kunden, Aufträge und Marktanteile bieten die mobile Kommunikation und Datenerfassung die notwendige Flexibilität, schnell und zuverlässig auf Kundenwünsche und sich ändernde Marktsituationen zu reagieren.<sup>26</sup> Mobile Endgeräte stellen für die Ausführung mobiler Kommunikation eine Plattform zur Verfügung und damit auch für die Ausführung mobiler Applikationen.<sup>27</sup>

Begünstigt durch den Rückgang an Festnetzanschlüssen sowie die rückläufige Anzahl an Festnetzminuten in Deutschland, ist aktuell eine Zunahme an Mobilfunkkunden zu verzeichnen.<sup>28</sup> Dieser Trend ist mitunter durch ein Angebot attraktiver Mobilfunktarife und -verträge zu erklären. Analog zu der steigenden Anzahl an Mobilfunkanschlüssen wächst die Nachfrage und Nutzung mobiler Endgeräte.<sup>29</sup> Gegenwärtige portable Endgeräte verfügen über hohe Leistungs- und Verwendungsmöglichkeiten und entwickeln sich immer mehr zu Allzweckgeräten, bei denen die Möglichkeit der Telefonie fast zweitranzig erscheint. Damit werden mobile Endgeräte auch für den Einsatz in Unternehmen interessant.<sup>30</sup>

Mobile Endgeräte unterliegen raschen Innovationszyklen und werden in großer Produktvielfalt auf dem Markt angeboten.<sup>31</sup> Daher ist es schwer eine einheitliche und zeitgemäße Begriffsdefinition anzugeben. Aufgrund dieser Tatsache sind in der Literatur häufig Begriffsbestimmungen aus software- und hardwaretechnischer Sichtweise anzutreffen. Innerhalb dieser Ausführung erfolgen zum einen der Bezug auf unterschiedliche Arten mobiler Endgeräte, wie bspw. Smartphones oder Tablet-PCs<sup>32</sup>, und zum anderen auf

---

<sup>25</sup> Vgl. Rügge 2007, S. 1.

<sup>26</sup> Vgl. Abts und Mülder 2010, S. 96.

<sup>27</sup> Vgl. Maske 2012, S. 207.

<sup>28</sup> Vgl. Bitkom 2012a.

<sup>29</sup> Vgl. Bitkom 2012b.

<sup>30</sup> Vgl. Euler et al. 2012, S. 108.

<sup>31</sup> Vgl. BSI 2012.

<sup>32</sup> Vgl. BSI 2006, S. 5; Pree 2006, S. 1135; Fuchs 2009, S. 18 f.; Roth 2005, S. 5; Rügge 2007, S. 18 f.

vorhandene mobile Anwendungen, wie bspw. Betriebssysteme oder Anwendungssoftware.<sup>33</sup> Für die weitere Betrachtung mobiler Endgeräte wird dem Begriff folgende Definition zugewiesen:<sup>34</sup>

► „Ein **mobiles Endgerät** ist ein singuläres mit Prozessen ausgestattetes elektronisches Gerät, das a) drahtlos und mittels Batterie(n) an jeden beliebigen Ort transportiert werden kann, b) während des Transports (ohne zusätzliche Stützfläche) benutzt werden kann, c) über integrierte Ein- und Ausgabemodalitäten (z. B. Bildschirm, Tastatur etc.) verfügt und d) alle Komponenten in einem Gehäuse vereint.“

Im Sinne dieser Definition wird im Folgenden zunächst der Mobilitätsbegriff allgemein beleuchtet. Im Anschluss daran erfolgt die Klassifizierung unterschiedlicher mobiler Endgeräte. Auf Grundlage dieser Einordnung werden Eigenschaften und Funktionen mobiler Endgeräte beschrieben. Das Kapitel schließt mit der Vorstellung aktueller Markt- und Absatzzahlen ab.

#### 4.2.1 Mobilitätsbegriff

Im Rahmen der in diesem Kapitel behandelten Thematik von mobilen Endgeräten sowie Applikationen soll an dieser Stelle der Terminus Mobilität für die weitere Verwendung dieser Begrifflichkeiten näher betrachtet werden. Der Begriff Mobilität stammt aus dem lateinischen Wort „mobilitas“ ab und kann als Synonym für Beweglichkeit betrachtet werden.<sup>35</sup> Die Autoren Reichenwald, Meier und Freimuth definieren Mobilität „als den Wechsel eines Gegenstands zwischen den definierten Einheiten eines Systems“.<sup>36</sup> Als Gegenstände werden neben materiellen Objekten auch Personen oder immaterielle Gegenstände klassifiziert. Das bereits angesprochene System der Mobilität besteht aus den Dimensionen Zeit und Raum und lässt sich weiter in physische und informationelle Mobilität unterscheiden. Im Sinne der informationellen Mobilität kann dem Begriff „Mobilität“ das Bedürfnis nach umfassender Informationsversorgung zu jeder Zeit und an jedem Ort zugewiesen werden. Bei dem Gegenstand Mobilität handelt es sich hierbei um Daten und Informationen. Aus dieser Denkweise heraus geht Mobilität über die bloße Tragbarkeit eines entsprechenden Endgerätes hinaus und wird durch die Eigenschaften und Funktion des Gerätes sowie durch verfügbare Kommunikationsmöglichkeiten bestimmt.<sup>37</sup> Hauptaugenmerk der informationellen Sichtweise besteht in der Signalübertragung, welche

---

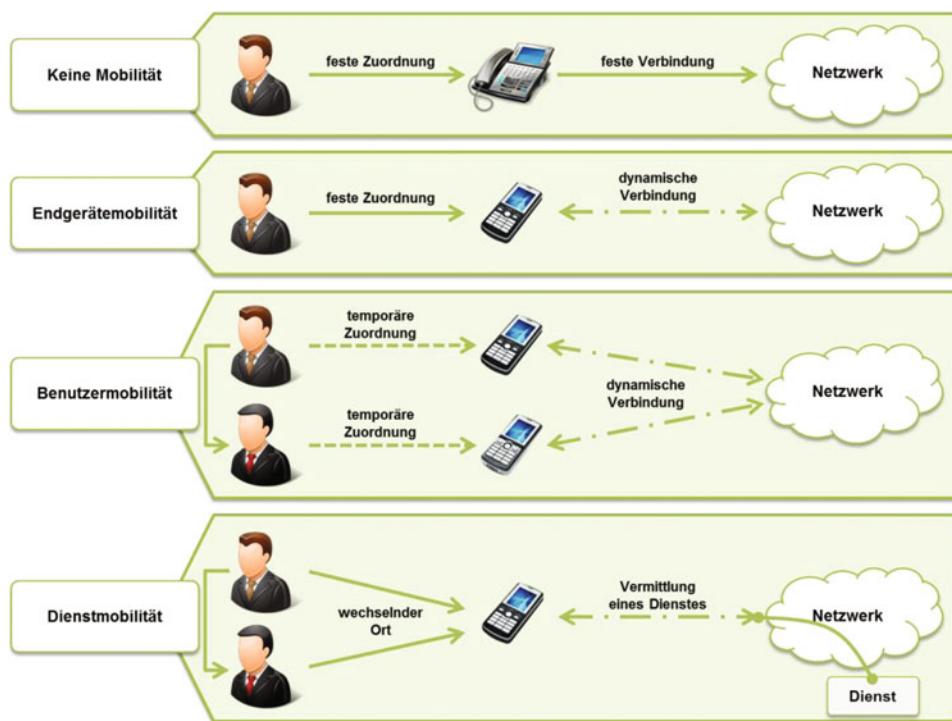
<sup>33</sup> Vgl. Kuassi und Bischel 2012, S. 130; Maske 2012, S. 207; Christmann und Hagenhoff 2009, S. 16; BSI 2006, S. 5.

<sup>34</sup> Krannich 2010, S. 37.

<sup>35</sup> Vgl. Scholze-Stubenknecht et al. 2000, S. 659.

<sup>36</sup> Reichenwald et al. 2002, S. 7.

<sup>37</sup> Reichenwald et al. 2002, S. 7.



**Abb. 4.3** Arten von Mobilität. (eigene Erstellung, in Anlehnung an Roth 2005, S. 8)

entweder leistungsgebunden über Kabelverbindungen oder drahtlos, bspw. über WLAN oder Funkverbindungen, stattfinden kann.<sup>38</sup>

Neben der informationellen Sichtweise erfolgt weiterhin die Betrachtung der physischen Mobilität, welcher Gegenstände oder Personen zugeordnet werden.<sup>39</sup> Aus der physischen Sichtweise heraus unterscheidet Roth im Allgemeinen zwischen Endgeräte-, Benutzer- und Dienstmobilität (vgl. Abb. 4.3):<sup>40</sup>

- Die Benutzermobilität ist dadurch gekennzeichnet, dass Personen unpersönliche Endgeräte in ihrer Umgebung zur Nutzung bestimmter Dienste in Anspruch nehmen können. Der Fokus liegt somit auf einem mobilen Benutzer, der an ein stationäres Gerät an ausgezeichneten Standorten gebunden ist.<sup>41</sup> Ein Beispiel hierfür sind EC-Kartenautomaten oder Ticketsysteme aus dem öffentlichen Nah- und Fernverkehr.

<sup>38</sup> Vgl. Meier 2002, S. 49.

<sup>39</sup> Vgl. Meier 2002, S. 47.

<sup>40</sup> Vgl. Roth 2005, S. 7 f.

<sup>41</sup> Vgl. Lonthoff 2004, S. 453

- Im Gegensatz zur Benutzermobilität garantiert die Endgerätemobilität die ununterbrochene Vernetzung eines Endgerätes bei räumlicher Veränderung.<sup>42</sup> Benutzern ortsunabhängiger Endgeräte wird es somit ermöglicht, mobile Dienste, wie bspw. Sprach-, Video- oder Bildübertragung, zu nutzen. Zur Gewährleistung der Endgerätemobilität werden drahtlose Netze benötigt.<sup>43</sup>
- Im Mittelpunkt der Dienstmobilität, auch als Dienstkonnektivität bezeichnet, stehen Kommunikationsdienste, unabhängig vom Standort des Benutzers. Diese Form der Mobilität bezeichnet weiterhin Dienste, welche sich von Gerät zu Gerät übernehmen lassen oder von überall aus erreicht werden können.<sup>44</sup> Ein Benutzer kann bspw. zu jeder Zeit und von jedem Standort aus seine E-Mails abrufen. Ebenfalls werden zur Realisierung der Dienstmobilität drahtlose Netze benötigt.<sup>45</sup>

Innerhalb der nachfolgenden Abschnitte werden mobile Endgeräte unter dem Prinzip der Endgerätemobilität charakterisiert. Für die erfolgreiche Entwicklung von mobilen Applikationen müssen weiterhin Endgeräte klassifiziert sowie deren Eigenschaften und Funktionen betrachtet werden. Hierzu wird im folgenden Abschnitt versucht, mobile Endgeräte in einer einheitlichen Systematisierung anzutragen.

#### **4.2.2 Typologisierung mobiler Endgeräte**

Mobile Endgeräte umfassen ein sehr breites Spektrum von tragbaren Computern, die sich deutlich von den herkömmlichen Personal Computern (PC) unterscheiden. Die Autoren Turowski und Poussotchi geben eine detaillierte Übersicht über vorhandene Endgeräte. Sie zählen Mobiltelefone, Smartphones, Personal Digital Assistants (PDAs), Tablet-PCs sowie Notebooks und Laptops zu der Gruppe mobiler Endgeräte.<sup>46</sup> Neben dieser Einordnung nach definierten Kriterien erfolgt durch Roth die Klassifizierung mobiler Endgeräte nach der Nutzungsart. In diesem Zusammenhang differenziert Roth in universelle und spezielle Endgeräte. Zu den universellen Geräten zählen alle Geräte, die durch den Hersteller für keinen festgelegten Zweck bestimmt sind und somit eine Installation beliebiger Anwendungen ermöglichen. Spezialgeräte hingegen sind für die Bewältigung bestimmter Aufgaben konstruiert, bei denen eine Änderung der Anwendungen nicht vorgesehen ist.<sup>47</sup> Frohberg unterscheidet mobile Endgeräte weiterhin nach dem Grad ihrer Portabilität. Eine Unterscheidung in stationäre, portable und mobile Geräte ist hierbei sinnvoll:<sup>48</sup>

---

<sup>42</sup> Vgl. Roth 2005, S. 7.

<sup>43</sup> Vgl. Roth 2005, S. 7.

<sup>44</sup> Vgl. Book et al. 2005, S. 123.

<sup>45</sup> Vgl. Roth 2005, S. 7.

<sup>46</sup> Vgl. Turowski und Poussotchi 2004, S. 57.

<sup>47</sup> Vgl. Roth 2005, S. 387 f.

<sup>48</sup> Vgl. Frohberg 2008, S. 17 f.

- **Stationär**

Als stationär werden alle Rechner bezeichnet, welche fix verkabelt an einem bestimmten Ort stehen und typischerweise diesen Standort längerfristig beibehalten. Diese Rechner zeichnen sich dadurch aus, dass sie schwer und unhandlich, aber leistungsstark sind. Für den erfolgreichen Betrieb von stationären Rechnern werden meistens externe Hardwarekomponenten, wie z. B. Maus, Tastatur und Bildschirme, benötigt.

- **Portabel**

Laptops, Note- und Netbooks werden üblicherweise als portable Geräte bezeichnet. Bei diesen Computergeräten sind bereits Tastatur, Bildschirm und Maus sowie diverse Speichermedien und Laufwerke in das Gehäuse integriert. Diese Geräte sind dafür ausgelegt, dass sie von Standort zu Standort transportiert und allgegenwärtig genutzt werden können.

- **Mobil**

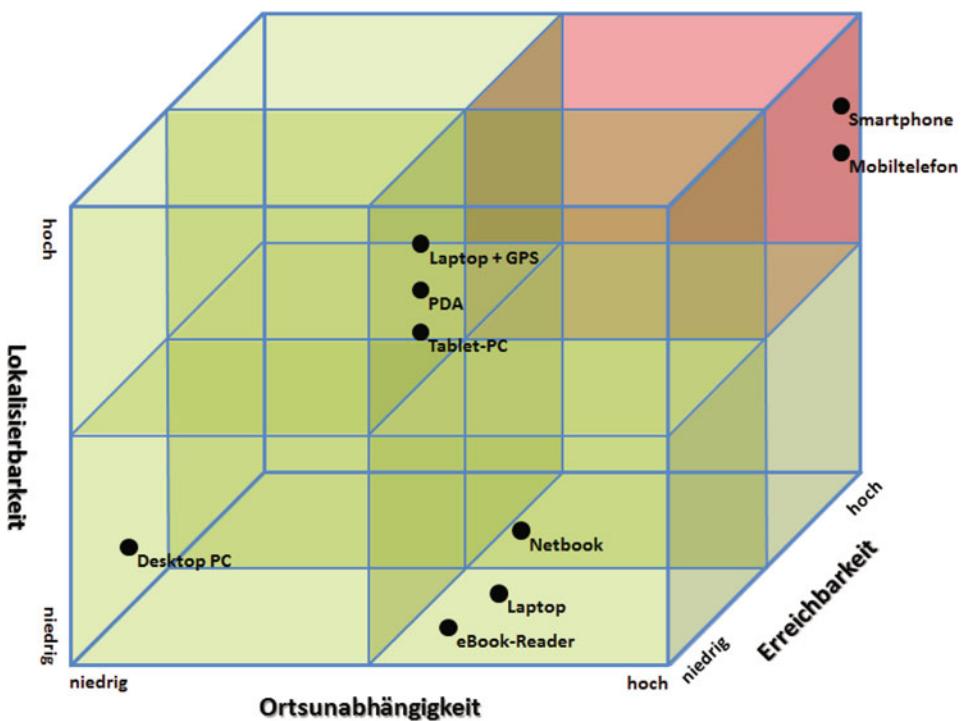
Ebenso wie portable Geräte können mobile Geräte bewegt werden. Im Unterschied zu portablen Geräten sind mobile Geräte eher als persönliche Accessoires anzusehen, welche immer am Körper mitgeführt werden können. Vorrangig werden mobile Geräte für kurzfristige Aktivitäten, wie z. B. für das Schreiben einer SMS oder das Aufnehmen eines Videos, benutzt. Mobile Geräte zeichnen sich vor allem dadurch aus, dass sie ohne eine Unterlage entweder innerhalb einer Bewegung oder stehend ausgeführt werden können.

Das Durlacher Institut hat bereits in einer Studie aus dem Jahr 1999 insgesamt sieben Eigenschaften benannt, die Endgeräte für die mobile Kommunikation mit sich bringen müssen. Tschersich stützt sich auf die Ergebnisse der Studie und bezieht sich bei seinem Ansatz zur Klassifizierung mobiler Endgeräte auf die drei Dimensionen Lokalisierbarkeit, Erreichbarkeit und Ortsunabhängigkeit. Die Betrachtung dieser drei Attribute innerhalb einer Acht-Quadranten-Matrix (vgl. Abb. 4.4) ermöglicht die Systematisierung aller auf dem Mobilfunksektor bekannten Endgeräte. Gerätetypen, bei denen alle drei Dimensionen besonders hoch ausgeprägt sind, erfüllen nach Tschersich alle Kriterien, um als mobiles Endgerät bezeichnet werden zu können.<sup>49</sup>

Die genaue Betrachtung der drei Dimensionen lässt Parallelen zu den Ausprägungen physischer Mobilität erkennen. Die Lokalisierbarkeit mobiler Endgeräte wird durch Technologien wie GPS ermöglicht, welche zur genauen Ortung von Endgeräten eingesetzt werden und dadurch die Möglichkeit offerieren, Dienstleistungen, je nach Standort des Nutzers, anbieten zu können. Die Erreichbarkeit zeichnet sich dadurch aus, dass Benutzer mobiler Endgeräte zu jeder Zeit und an jedem Ort erreichbar sind. Ortsunabhängigkeit bedeutet, dass unabhängig vom Standort des Benutzers Informationen und Daten über mobile Endgeräte abgerufen oder versendet werden können. Die gerade beschriebenen drei Attribute richten sich somit an den Grundgedanken der physischen Mobilität im Sinne der Benutzer-, Endgeräte- und Dienstmobilität.

---

<sup>49</sup> Vgl. Tschersich 2010.



**Abb. 4.4** Durlacher Umfeldanalyse von mobilen Endgeräten. (eigene Erstellung, in Anlehnung an Tschersich 2010)

Im folgenden Abschnitt werden gegenwärtig am Markt vorhandene mobile Betriebssysteme betrachtet, welche in Anlehnung an die in Abb. 4.4 dargestellte Matrix hauptsächlich auf Smartphones und Tablet-PCs operieren. Hierzu zählen die Betriebssysteme iOS von Apple, Android von Google und Windows Phone von Microsoft.

#### 4.2.3 Mobile Betriebssysteme für Smartphones und Tablet-PCs

Die Norm ISO/IEC 2382 differenziert Software in Anwendungs-, System- und Unterstützungssoftware. Einen wichtigen Bestandteil der Systemsoftware bildet das Betriebssystem (auch Operating System genannt), welches zum einen die Grundlage für die möglichen Betriebsarten des Computers darstellt und zum anderen die Ausführung der Anwendungsprogramme steuert und überwacht.<sup>50</sup> Nachfolgend werden die Betriebssysteme der zuvor vorgestellten mobilen Endgeräte kurz beschrieben (vgl. Abb. 4.5):

<sup>50</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 66.



Apple iOS 7

Google Android 4.2

Windows Phone 8

**Abb. 4.5** Übersicht aktueller mobiler Betriebssysteme. (eigene Erstellung, in Anlehnung an: Apple 2013; Google 2013; Microsoft 2013a)

- **Apple iOS7<sup>51</sup>**

Das speziell für Apples mobile Endgeräte (iPod, iPhone, iPad) angepasste Betriebssystem basiert auf der Grundlage des Betriebssystems Mac OS X und wurde erstmals im Jahr 2007 auf dem ersten iPhone vorgestellt. Im Laufe der Zeit wurde das Betriebssystem und der damit verbundene Funktionsumfang sukzessive dem technologischen Fortschritt angepasst und befindet sich nun in der Version 7.0.2. In dieser Version beinhaltet das Betriebssystem einen E-Mail-Client, ein Programm zur SMS-Verwaltung, eine Möglichkeit, schnurlos zu drucken, einen Browser, ein Tool zur Sprachsteuerung sowie eine Multimediaanwendung, um Videos und Bilder sowohl selbst zu erstellen als auch abzuspielen.

- **Google Android 4.2<sup>52</sup>**

Das von Google mitentwickelte Betriebssystem Android basiert auf einem Linux-Kernel der Version 2.6.<sup>53</sup> Ein Großteil der Android-Plattform ist unter die Open-Source-Lizenz gestellt worden. Die Verwendung von Android ist somit kostenfrei und bietet für Entwickler neben der Bereitstellung von Programmierschnittstellen auch eine umfassende Quellcodesammlung an. Der Funktionsumfang des Betriebssystems liefert Programme zum Senden und Empfangen von E-Mails, SMS oder MMS, zum Surfen im Internet sowie zum Aufnehmen und Abspielen von Fotos und Videos. Die Nutzer von Android

<sup>51</sup> Vgl. Apple 2013.

<sup>52</sup> Vgl. Google 2013.

<sup>53</sup> Vgl. Maske 2012, S. 403.

können zudem ihren eigenen individuellen Startbildschirm einstellen. Dies ermöglichen sogenannte Miniprogramme, welche z. B. aktuelle Wetter- oder Kalenderdaten direkt auf dem Startbildschirm anzeigen.

- **Microsoft Windows Phone 8<sup>54</sup>**

Das von Microsoft entwickelte Betriebssystem Windows Phone 8 ist seit Juni 2012 erhältlich und basiert auf dem Vorgängersystem Windows Phone 7.5. Microsoft setzt seit dem Erscheinen des Betriebssystems Windows Phone auf ein Kachelkonzept. Der Benutzer hat hierbei direkten Zugriff auf die Telefonfunktion, die SMS-Verwaltung, Kalendereinträge sowie auf soziale Netzwerke. Im Gegensatz zu den anderen Betriebssystemen arbeitet Windows Phone 8 nach dem Prinzip des Cloud-Computing, d. h. internetbezogene Dienste, wie z. B. Facebook, werden integriert und mit den Funktionen des Betriebssystems verknüpft.

Die in Abb. 4.5 dargestellten Betriebssysteme besitzen neben den bereits aufgelisteten speziellen Funktionen allgemeine Eigenschaften, wie bspw. Multitasking, Datenbankverwaltung sowie umfangreiche Personalisierungsmöglichkeiten. Im Gegensatz zu den allgemeinen Eigenschaften und Funktionen mobiler Endgeräte werden im Folgenden spezifische Eigenschaften mobiler Betriebssysteme betrachtet.

#### 4.2.4 Spezifische Eigenschaften mobiler Endgeräte

Nach dem Einstieg in das begriffliche Umfeld sowie der Klassifizierung mobiler Geräte stellt sich die Frage nach deren Eigenschaften und Funktionen. Diese sind an die Technik mobiler Endgeräte gekoppelt, die von Hersteller zu Hersteller unterschiedlich ausgeprägt ist. Trotz des rapiden technologischen Fortschrittes sind Unternehmen bei der Herstellung mobiler Endgeräte, im Vergleich zu Desktopanwendungen, an verschiedene Einschränkungen gebunden:<sup>55</sup>

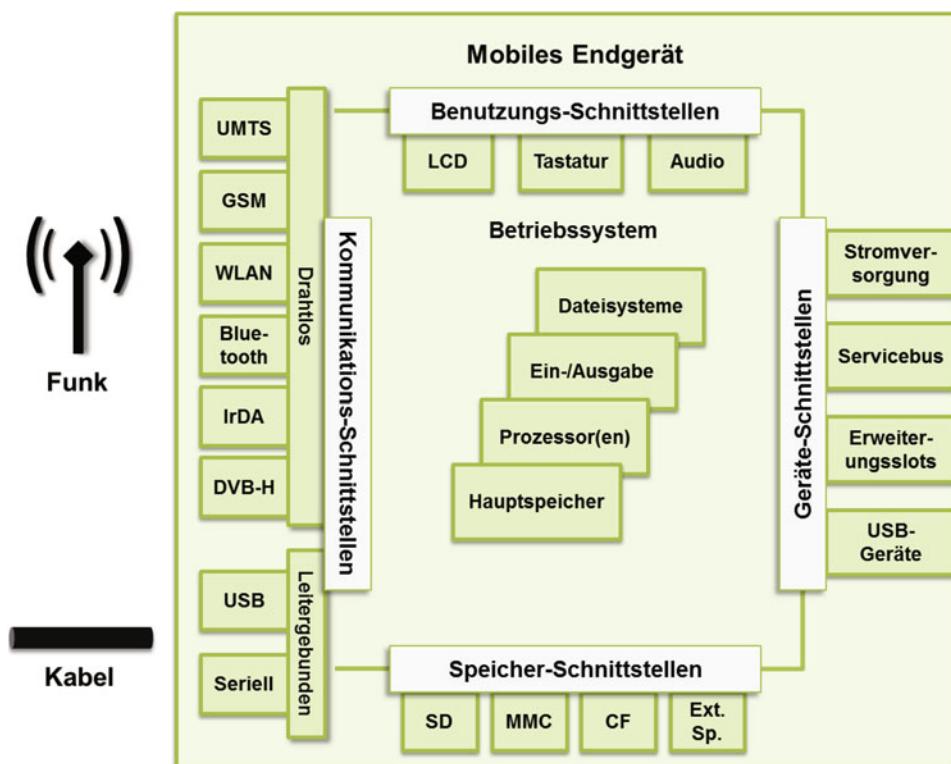
- Geringe Größe und Auflösung des Displays.
- Geringe Batterielebenszeit.
- Geringe Prozessorleistung.
- Geringe Speicherkapazität.
- Schlechte Dateneingabe und -ausgabe.
- Reduzierte Anzahl an Benutzerschnittstellen.

Die genannten Restriktionen wirken sich auf die Funktionalität sowie die Konzeption der Geräte aus und sind weiterhin ausschlaggebend für den mobilen Charakter. Im Gegensatz zu stationären Computern ist die Hardware mobiler Endgeräte nur mit hohem Aufwand

---

<sup>54</sup> Vgl. Microsoft 2013a.

<sup>55</sup> Vgl. Müller-Wilken 2002, S. 19; Roth 2002, S. 5 f.; Meier und Stormer 2008, S. 212.



**Abb. 4.6** Blockbild eines mobilen Endgerätes. (eigene Erstellung, in Anlehnung an BSI 2012, S. 6)

veränderbar. Oftmals ist der Austausch oder die Erweiterung integrierter Hardwarekomponenten nicht vorgesehen.<sup>56</sup> Abbildung 4.6 zeigt den allgemeinen Aufbau eines mobilen Endgerätes unter einem hardwareorientierten Blickwinkel.

Die Architektur mobiler Endgeräte setzt sich aus hardwarebezogenen Benutzungs-, Geräte-, Speicher- und Kommunikationsschnittstellen sowie aus softwarebasierenden Betriebssystemen zusammen. Zu den Benutzerschnittstellen gehören u. a. LCD-Displays, Tastaturen und Audiokomponenten, welche zur Steuerung des Endgerätes sowie zur Ein- und Ausgabe von Informationen vorgesehen sind. Die Geräteschnittstellen mobiler Endgeräte dienen primär zur Wiederaufladung der internen Stromversorgung. In weiteren Fällen dienen sie zur Erweiterung des Funktionsumfangs, bspw. durch die Anbindung zusätzlicher USB-Geräte. Die Speicherschnittstellen ermöglichen zum einen die Erweiterung des internen Speichers durch SD-, MMC- oder Compact-Flash-Speicherkarten (CF) und zum anderen den einfachen Austausch von Informationen und Daten. Kommunikations-schnittstellen bilden die Verbindung zu Datendiensten verschiedener Mobilfunkbetreiber oder anderer mobiler sowie stationärer Endgeräte. Für die Anbindung an Mobilfunknetze

<sup>56</sup> Vgl. BSI 2012, S. 5 f.

werden hierfür drahtlose Schnittstellen, bspw. UMTS oder GSM, verwendet. Für den Informationsaustausch mit anderen Endgeräten können drahtlose Verbindungen, wie bspw. Bluetooth oder WLAN, als auch leitergebundene Verbindungen, bspw. USB- oder serielle Kabel, benutzt werden. Der Übergang zwischen den durch Software bereitgestellten Anwendungen und den in Hardware realisierten Komponenten ist fließend. Das Blockbild aus Abb. 4.6 kann nicht als allgemeingültig betrachtet werden. Je nach Hersteller sind die Komponenten mobiler Endgeräte unterschiedlich ausgeprägt. Beispielsweise verfügen das iPhone sowie das iPad nicht über alle angegebenen Geräte- und Speicherschnittstellen.

Neben den genannten Beschränkungen mobiler Endgeräte aus hardwaretechnischer Sicht ergeben sich weitere inhärente Einschränkungen für mobile Endgeräte. Aufgrund der Portabilität mobiler Geräte sind diese anfälliger für Verlust oder Diebstahl. Nach einer aktuellen Studie wird alle 53 s in Europa ein Laptop gestohlen.<sup>57</sup> Aufbauend auf diesem Tatbestand müssen somit geeignete Schutzmaßnahmen überlegt werden. Eine weitere Einschränkung besteht in der Unzuverlässigkeit drahtloser Netze. Da drahtlose Netze noch nicht flächendeckend vorhanden sind, ist eine optimale Anbindung an Kommunikationsnetzwerke nicht immer gegeben. Im Vergleich zu leitergebundenen Übertragungswegen werden, je nach Standard des drahtlosen Netzes, nur geringe Datenraten zur Verfügung gestellt.<sup>58</sup>

---

### 4.3 Mobile Applikationen

Während sich der Nutzenschwerpunkt des mobilen Internets anfangs noch auf die Informationssuche und Nachrichtenübermittlung fokussiert hatte, rückte seit den letzten Jahren immer mehr das Bedürfnis nach Interaktion und Mitgestaltung in den Vordergrund. Diese Entwicklung ist nicht zuletzt auf einfach zu bedienende Geräte sowie intuitiv steuerbare Applikationen zurückzuführen. Immer mehr Unternehmen erkennen den daraus resultierenden Mehrwert und ermöglichen den Zugriff auf Dienstleistungen über mobile Applikationen. Dabei suchen sie nach neuen innovativen Wegen, um den Kontakt mit Kunden auf- sowie auszubauen, um dadurch einen beiderseitigen Nutzen zu schaffen.<sup>59</sup>

Das Aufkommen mobiler Applikationen hat in kürzester Zeit zu einer neuen Sichtweise auf die Lösung alltäglicher Probleme und Aufgaben geführt. Damit verbunden sind neue Dienstleistungsangebote im privaten als auch im geschäftlichen Bereich. Aus produktorientierter Sichtweise lassen sich für mobile Anwendungen vier Klassen identifizieren:<sup>60</sup>

---

<sup>57</sup> Vgl. Kallus 2013.

<sup>58</sup> Vgl. Satyanarayanan 1996, S. 1 f.

<sup>59</sup> Vgl. Berger und Lehner 2002, S. 85.

<sup>60</sup> Vgl. Schuhmann 2002, S. 7.

- Informationsorientierte Dienste, wie bspw. Reiseinformationen, Börseninformationen oder Nachrichten.
- Applikationsorientierte Dienste, wie bspw. Computerspiele, Übersetzungsdienste oder Währungsrechner.
- Transaktionsorientierte Dienste, wie bspw. Bezahlstellen, Reservierungsdienste oder Tauschbörsen.
- Kommunikationsorientierte Dienste, wie bspw. E-Mail, Chat oder soziale Netzwerke.

Demnach sind mobile Applikationen durch ihre Eigenschaft, Dienste über das Internet oder sonstige private Netzwerke nutzen sowie bereitstellen zu können, gekennzeichnet. In diesem Zusammenhang werden gegenwärtig immer mehr Cloud-Computing-Dienste entwickelt, welche konsistente Datenbestände auf unterschiedlichen mobilen und stationären Endgeräten zur Verfügung stellen.<sup>61</sup> Nach Angaben von Schuhmann, Eckert und Schneider liegt das Potenzial mobiler Applikationen somit in der Mensch-zu-Mensch- sowie in der Mensch-zu-Maschine-Kommunikation. Im Vordergrund dieser Annahmen steht der Austausch von Informationen über digitale Dienste sowie über ein oder mehrere unterschiedliche Endgeräte im Sinne der Dienstmobilität (vgl. Abb. 4.3).

Zur Veranschaulichung der Architektur und Zugriffsmöglichkeiten mobiler Systeme müssen zunächst die Begriffe „mobiler Dienst“ und „mobile Applikation“ deutlich voneinander abgegrenzt werden. Im Vergleich zu mobilen Applikationen, bei denen es sich um softwaretechnische Implementierungen handelt, stellen mobile Dienste den Endbenutzern Dienstleistungen in mobilen Anwendungsbereichen zur Verfügung. Komponenten für den Zugriff auf mobile Systeme sind hierbei mobile Endgeräte, Kommunikationstechnologien sowie unterschiedliche Backendlösungen, wie bspw. Server- oder Cloud-Computing-Systeme.<sup>62</sup> Abbildung 4.7 veranschaulicht den Zusammenhang zwischen mobilen Diensten, Applikationen, Endgeräten und Systemen.

Abbildung 4.7 zeigt, dass die Implementierung von Applikationen auf mobilen Endgeräten zur Realisierung mobiler Dienste notwendig ist. Zur weiteren Betrachtung mobiler Applikationen werden im Folgenden zunächst unterschiedliche Definitionen des Begriffs Applikation aufgezeigt und eine allgemeingültige Begriffsbestimmung entwickelt. Daran anknüpfend erfolgt, in Bezug auf unterschiedliche Entwicklungsansätze bei der Applikationsentwicklung und Distributionstypologien, die Klassifizierung mobiler Anwendungen.

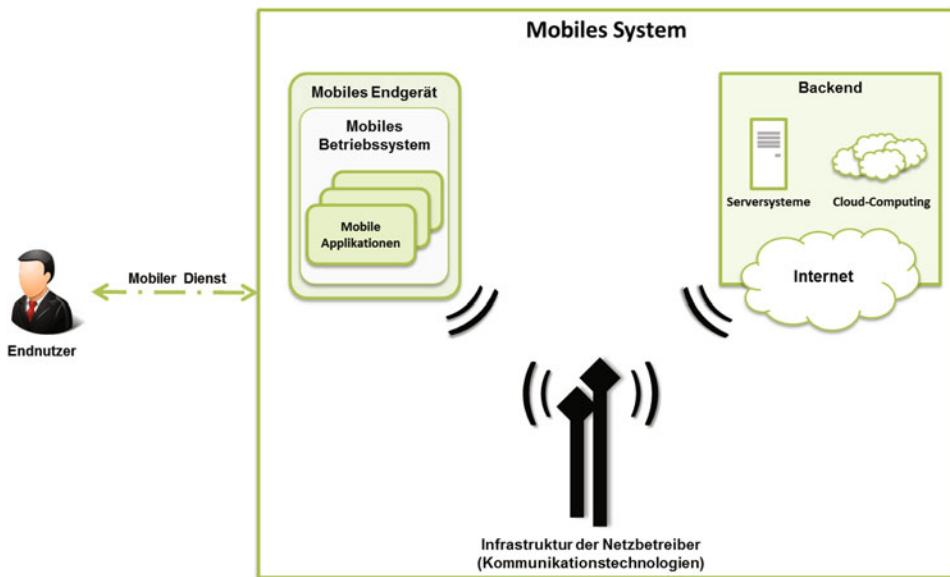
### 4.3.1 Applikationsbegriff

Eine erste Betrachtung und Bestimmung des Begriffs Applikation findet sich im Standardglossar der Softwaretechnik-Terminologie wieder:<sup>63</sup>

<sup>61</sup> Vgl. Eckert und Schneider 2012, S. 194.

<sup>62</sup> Vgl. Kraus 2012, S. 24.

<sup>63</sup> IEEE 1990, S. 10.



**Abb. 4.7** Akteure und Komponenten eines mobilen Systems. (eigene Erstellung; in Anlehnung an Kraus 2012, S. 24)

► „**application software**. Software designed to fulfill specific needs of a user; for example, software for navigation, payroll, or process control.“

Diese Definition beschreibt die Aufgabe sowie den Gegenstand von Applikationen analog zur Anwendungssoftware. Im Laufe der Zeit hat sich der Begriff Applikation in Bezug auf die mobile Nutzung und Anwendung jedoch mehr als eigenständiger Begriff etabliert und wird in Abgrenzung zur stationären Anwendungssoftware in unterschiedlichen Literaturquellen definiert. Nach Golding besitzen mobile Applikationen zwei grundlegende Eigenschaften, welche eine eindeutige Unterscheidung zu stationären Anwendungen ermöglicht:<sup>64</sup>

1. „[...] applications that have some kind of user interface on the handset [...]“
2. „[...] they utilise the wireless communications capability of the device they run on [...]“

Voraussetzungen für mobile Applikationen nach Golding sind somit zum einen vorhandene Schnittstellen für Benutzereingaben und zum anderen die Nutzung kabelloser Übertragungstechniken. In ähnlicher Weise definiert der Autor Lehner den Begriff „mobile Applikation“. Er bezeichnet diese als abstrahierte Eigenschaft eines computergestützten Systems, welches drahtlos mit anderen Computersystemen über das Internet kommuniziert.

<sup>64</sup> Golding 2008, S. 13.

nisiert.<sup>65</sup> Nach dieser Definition ist weiterhin festzuhalten, dass neben den bereits von Golding angesprochenen kabellosen Übertragungstechniken, Betriebssysteme zur Ausführung und Nutzung mobiler Applikationen vorausgesetzt werden. Eine aktuelle Definition wird durch den Autor Maske gegeben, der bei seinen Ausführungen mobile Endgeräte sowie die Eigenschaften dieser berücksichtigt.<sup>66</sup>

► „Eine **mobile Applikation** ist eine spezielle Applikationsform, die zur Ausführung auf mobilen Endgeräten konzipiert ist. [...] Applikationen, die auf mobilen Endgeräten ausgeführt werden, müssen die besonderen Eigenschaften dieser Gerätekategorien im Vergleich zu klassischen Desktop-PCs berücksichtigen.“

Für die vorliegende Arbeit lassen sich obige Definitionen zu einer allgemeingültigen Begriffsbeschreibung wie folgt zusammenfassen:

► „Eine **mobile Applikation** stellt eine spezifische Anwendungssoftware dar, die zur Anwendung auf einem Betriebssystem sowie zur Ausführung auf mobilen Endgeräten entwickelt wird und neben der Berücksichtigung besonderer Endgeräteeigenschaften, die Nutzung kabelloser Übertragungstechniken voraussetzt.“

Die Entwicklung von mobilen Applikationen hat sich innerhalb der letzten Jahre aus einem Nischenbereich heraus zu einem der wichtigsten Wachstumsmärkte entwickelt. Nach Aussagen einer IBM-Studie aus dem Jahr 2010 waren 55 % der befragten IT-Experten der Meinung, dass die Entwicklung von mobilen Applikationen im Jahre 2015 wichtiger sein wird, als die bisher übliche Anwendungsentwicklung.<sup>67</sup> Nachfolgend werden aktuelle Applikationstypen vorgestellt und Unterschiede hervorgehoben.

### 4.3.2 Applikationstypen und Entwicklungsstrategien

Zu Beginn der Planungsphase stehen Softwareentwickler vor der Auswahl geeigneter Entwicklungsstrategien zur Realisierung des Projektvorhabens. In diesem Sinne muss auf strategischer Ebene zunächst entschieden werden, auf welchen Plattformen die zu erstellende Applikation implementiert werden soll. Die Auswahl hängt in erster Linie vom Geschäftsmodell des mobilen Dienstes sowie von der fokussierten Zielgruppe der Applikation ab. Sofern die Applikation für nur eine spezielle Endgerätegruppe oder für ein spezielles Betriebssystem erfolgen soll, bietet sich eine plattformspezifische Entwicklung an. Für die Verbreitung der Applikation auf unterschiedlichen Betriebssystemen und Endgeräteklassen müssen plattformunabhängige Entwicklungsstrategien eingesetzt werden.<sup>68</sup>

<sup>65</sup> Vgl. Lehner 2002, S. 5.

<sup>66</sup> Vgl. Maske 2012, S. 106 f.

<sup>67</sup> Vgl. IBM 2010.

<sup>68</sup> Vgl. Kraus 2012, S. 26.

Die beiden Varianten der plattformabhängigen und -unabhängigen Entwicklung werden im Folgenden näher betrachtet.

Zur plattformabhängigen Entwicklungsstrategie zählt die Programmierung von mobilen Applikationen in einem nativen Code. Native Applikationen beschreiben Anwendungen, die nur auf einem bestimmten Endgerätetyp und dessen zugehörigem Betriebssystem lauffähig sind.<sup>69</sup> Hierzu wird die Installation auf dem jeweiligen Endgerät vorausgesetzt. Die Entwicklung nativer Anwendungen ist dann sinnvoll, wenn die Applikation auf gegebene Hardware- und Softwareressourcen eines bestimmten Endgerätes zugreifen soll, bspw. auf die interne Kamera, Bewegungssensoren oder GPS. Gleichzeitig bringt dies jedoch einige Nachteile mit sich, da für die Portierung der Applikation auf weitere Endgeräte und Betriebssysteme hohe Entwicklungskosten entstehen. Ein weiterer Nachteil besteht in der Verbreitung nativer Applikationen, die hauptsächlich über plattformspezifische Marktplätze erfolgen kann.<sup>70</sup>

Im Gegensatz hierzu werden webbasierte Applikationen über das Internet angeboten und umgehen so die Restriktionen und Rahmenbedingungen plattformspezifischer Marktplätze. Webanwendungen können von jedem beliebigen mobilen Endgerät mit vorhandenem Internetbrowser aufgerufen und genutzt werden. Aus diesem Grund zählen Webanwendungen zu den plattformunabhängigen Entwicklungsstrategien.<sup>71</sup> Webbasierte Anwendungen werden auf der Grundlage von HTML, CSS oder JavaScript entwickelt und über das Internet zur Verfügung gestellt. Eine Installation der Anwendung wird somit nicht vorausgesetzt. Ebenfalls müssen bei der Entwicklung keine technischen Restriktionen des jeweiligen Endgerätes berücksichtigt werden. Für die Nutzung webbasierter Anwendungen ist somit eine aktive Internetverbindung notwendig. Bisher weisen Webanwendungen noch einen eingeschränkten Zugriff auf die Hardwareressourcen des jeweiligen Endgerätes auf. Grund hierfür sind fehlende Schnittstellen, welche Zugriffe auf die Hardware mobiler Endgeräte über webbasierte Programmiersprachen ermöglichen.<sup>72</sup>

Gegenwärtig versuchen sogenannte hybride Applikationen, die Vorteile beider bisher genannten Entwicklungsstrategien zu vereinen. Hybride Anwendungen basieren zum einen auf einem nativen Kern, der die Verbindung zu den Hardwareressourcen ermöglicht, und zum anderen auf webbasierten, plattformübergreifenden Funktionen. Hierfür wurden spezielle Frameworks entwickelt, die plattformunabhängige Anwendungen in native Anwendungen umwandeln.<sup>73</sup> Die Zukunft dieser Entwicklungsstrategie ist jedoch fraglich, da viele Hersteller, wie Apple oder Google, ihre Marktdominanz ausnutzen, um hybride Technologien und Frameworks zu verhindern. Weiterhin stehen hybride Applikationen aufgrund mangelnder Sicherheit und fehlender Offenheit in der Kritik vieler

---

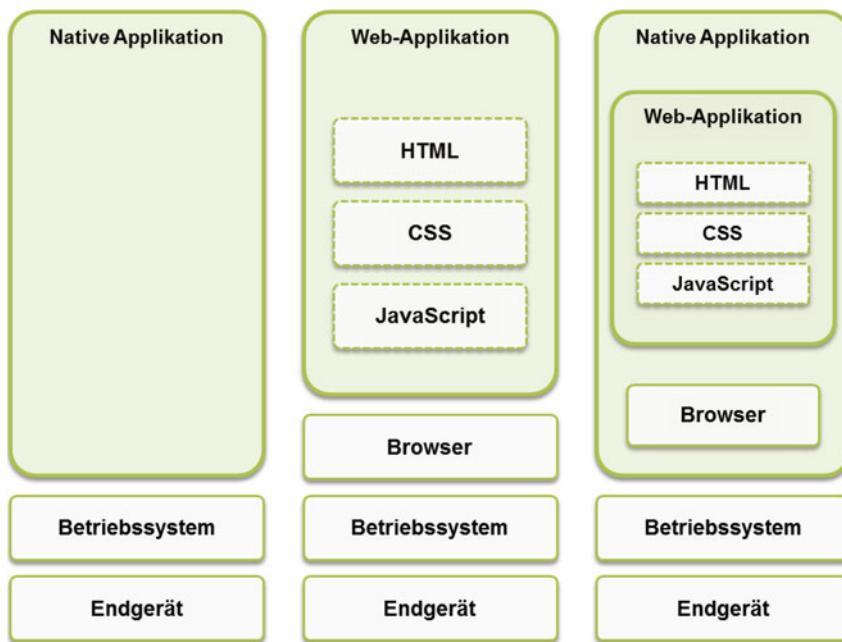
<sup>69</sup> Vgl. Christmann et al. 2010, S. 32.

<sup>70</sup> Vgl. Scheller 2011, S. 23.

<sup>71</sup> Vgl. Christmann et al. 2010, S. 32.

<sup>72</sup> Vgl. Scheller 2011, S. 23 f.

<sup>73</sup> Vgl. Christmann et al. 2010, S. 33.



**Abb. 4.8** Nativer, webbasierter und hybrider Ansatz zur mobilen Applikationsentwicklung

Entwickler.<sup>74</sup> Abbildung 4.8 zeigt native, webbasierte sowie hybride Ansätze zur mobilen Applikationsentwicklung.

#### 4.4 Instrumente und Werkzeuge der mobilen Anwendungsentwicklung

Nach aktuellem Stand der Technik wird die Softwareentwicklung durch den überwiegenden Einsatz von Menschen geprägt, die zur Planung, Erstellung und Wartung von Software auf rechnergestützte Werkzeuge und Instrumente zurückgreifen. Softwareentwickler beschäftigen sich somit nicht nur mit der Abarbeitung technischer Prozesse der Softwareentwicklung, sondern auch mit Aktivitäten der Projektverwaltung sowie des Qualitätsmanagements. Im Sinne des Software Engineering besteht die Hauptaufgabe der Entwickler in der Auswahl und Anwendung passender Methoden zur systematischen und effektiven Herstellung qualitativ hochwertiger Software.<sup>75</sup>

Instrumente und Werkzeuge der Softwareentwicklung unterstützen Entwickler über alle Tätigkeiten innerhalb des Softwareentwicklungsprozesses hinweg. Nach den Autoren

<sup>74</sup> Vgl. Jobs 2010.

<sup>75</sup> Vgl. Sommerville 2001, S. 22.

Ludewig und Lichter dienen Werkzeuge im Sinne der Softwareentwicklung zur Ausführung einer Arbeit.<sup>76</sup> Typische Arbeiten zur Softwarerealisierung werden nachfolgend kurz aufgelistet:<sup>77</sup>

- Analyse
- Spezifikation der Anforderungen
- Architekturentwurf
- Codierung und Modultest
- Integration, Test und Abnahme
- Betrieb und Wartung
- Auslauf und Ersetzung

Im Software Engineering muss der Begriff „Werkzeug“ von den Begriffen „Methode“ und „Notation“ differenziert werden. Während Werkzeuge hauptsächlich zur Transformation von Informationen eingesetzt werden, beschreiben Methoden Handlungsanweisungen, die den Entwicklern bei der Auswahl und Anwendung geeigneter Werkzeuge unterstützen. Im Gegensatz zu den Werkzeugen bleiben Werkstoffe im Softwareprodukt enthalten. Hierzu zählt die Notation einer Software, welche eine allgemeingültige Syntax und Semantik für die Erstellung des Programms vorgibt. Werkzeuge, Methoden und Notation können durch geeignete Softwarekonzepte zu einem Gesamtsystem miteinander verbunden werden.<sup>78</sup>

Der Erfolg mobiler Anwendungen und Endgeräte inspiriert Softwareentwickler neue anspruchsvolle Applikationen zu entwickeln oder bereits bestehende Anwendungen auf mobile Betriebssysteme zu portieren. Herausforderungen bestehen hierbei in der Entwicklung funktionsfähiger, sicherer und benutzerfreundlicher Applikationen. Im Vergleich zu Desktopanwendungen bestehen bei der Entwicklung mobiler Applikationen komplexere Rahmenbedingungen, wie bspw. geringere Speichergrößen, schwankende Netzabdeckung sowie die Verarbeitung sensibler Daten. Diese Anforderungen werden bei der Planung und Konzeption mobiler Anwendungen nicht immer ausreichend berücksichtigt und der Stellenwert der nötigen Qualitätssicherung oftmals unterschätzt.<sup>79</sup> Aus diesem Grund werden innerhalb des vorliegenden Kapitels Methoden, Werkzeuge und Programmiersprachen zur mobilen Anwendungsentwicklung näher betrachtet.

#### **4.4.1 Grundlagen der Programmierung**

Die Anfänge der Programmierung reichen bis in das 19. Jahrhundert zurück und wurden erstmals auf maschinell lesbaren Informationsträgern für den Einsatz in Webstühlen

---

<sup>76</sup> Vgl. Ludewig und Lichter 2010, S. 41.

<sup>77</sup> Vgl. Ludewig und Lichter 2010, S. 39 f.

<sup>78</sup> Vgl. Ludewig und Lichter 2010, S. 42.

<sup>79</sup> Vgl. Heidemann und Zumbruch 2012, S. 241 f.

entwickelt. Die Speicher bestanden aus kleinen Holzplatten, in denen der Code für ein bestimmtes Webmuster in einer vorgegebenen Lochkombination angebracht war. Die sogenannte Lochkarte wurde in den Ablesemechanismus des Webstuhls eingeführt und sorgte für die Einstellung der senkrechten Schnüre oder Litzen. Dieses Prinzip wurde später für weitere Mechanismen, wie bspw. zum Antrieb von Musikorgeln, verwendet.<sup>80</sup>

Mit der Herstellung digitaler Rechenautomaten befasste sich erstmals Charles Babbage, welcher mit seiner Idee der analytischen Maschine das menschliche Rechnen nachahmen wollte. Hierzu entwarf er auf hölzerner Basis ein Rechen- und Steuerwerk, einen Zahlen- und Zwischenspeicher sowie Ein- und Ausgabemöglichkeiten. Der Rechenautomat besaß somit alle Komponenten, welche auch heute in einem modernen Rechnersystem verbaut und eingesetzt werden. Aufgrund der damaligen unpräzisen Herstellungsverfahren sowie der mechanischen Komplexität konnte die analytische Maschine nie fertiggestellt werden.<sup>81</sup>

Durch die ständige Verbesserung der Lochkarten sowie die Einführung von Zahl-, Register- und Sortiermaschinen konnte letztendlich die Entwicklung elektronischer und moderner Rechner angetrieben werden. 1934 wurde durch Konrad Zuse der erste elektronische Rechner auf dem Grundkonzept von Babbage entwickelt. Der Rechner ermöglichte es, Rechenoperationen mit Dualzahlen durchzuführen und konnte zudem logische Operationen berücksichtigen. Zeitgleich mit dem Bau des ersten elektromechanischen Computers erfolgte die formale Definition von Programmiersprachen.<sup>82</sup>

Der gerade dargestellte historische Rückblick über die Entstehung elektronischer Computersysteme zeigt, dass der Hintergrund der Entwicklung von Rechenmaschinen ein vorliegendes Optimierungsproblem war, das maschinell gelöst werden sollte, bspw. die Herstellung eines Teppichs mit unterschiedlichen Webmustern oder die Berechnung mathematischer Problemstellungen. Übertragen auf den Bereich der Informatik beschreibt der Vorgang des Programmierens die Formulierung computergestützter Verfahren zur Lösung bestimmter Aufgaben. Zur Unterstützung dieses Vorgangs werden unter dem Paradigma der imperativen Programmierung Algorithmen und Datenstrukturen verwendet, welche die Wechselwirkungen sowie die Gesamtheit eines Programms abbilden.<sup>83</sup>

In diesem Abschnitt werden zunächst die Grundbegriffe der Programmierung, wie z. B. „Algorithmus“, „Variable“ und „Datentyp“, definiert. Darauf aufbauend erfolgt abschließend die Betrachtung von Kontrollstrukturen.

#### 4.4.1.1 Algorithmen

Einen zentralen Betrachtungsschwerpunkt der Informatik bilden Algorithmen sowie die damit verbundenen Datenstrukturen und benötigten Rechnerarchitekturen. Unter einem Algorithmus ist hierbei ein Schema zu verstehen, nach dem eine gestellte Aufgabe sys-

<sup>80</sup> Vgl. Levi und Rembold 2003, S. 21 f.

<sup>81</sup> Vgl. Levi und Rembold 2003, S. 22.

<sup>82</sup> Vgl. Levi und Rembold 2003, S. 23.

<sup>83</sup> Vgl. Pomberger und Pree 2004, S. 67.

matisch bearbeitet und beendet werden kann. Nach Pomberger und Pree definiert sich der Begriff Algorithmus wie folgt:<sup>84</sup>

► „Ein **Algorithmus** ist eine präzise, das heißt in einer festgelegten Sprache abgefasste endliche Beschreibung eines schrittweisen Problemlösungsverfahrens zur Ermittlung gesuchter Größen.“

Eine ähnliche Definition liefern die Autoren Hering, Gutekunst und Dyllong.<sup>85</sup>

► „Ein **Algorithmus** ist ein Verfahren zur Lösung von Problemen. Dazu wird angegeben, welche elementaren Schritte in welcher Reihenfolge zu erledigen sind.“

Ein Algorithmus kann weiterhin zwei Zustände annehmen. Wird in Anlehnung an die Begriffsbestimmung von Pomberger und Pree eine gesuchte Größe innerhalb einer endlichen Anzahl an Schritten ermittelt, so wird ein Algorithmus als „terminierend“ bezeichnet. Analog hierzu wird ein Algorithmus „nicht terminierend“ genannt, wenn der Wert einer gesuchten Größe nicht in endlich vielen Schritten ermittelt werden kann.<sup>86</sup>

Zur Umsetzung und Beurteilung von Algorithmen sind nachfolgend wichtige Eigenschaften aufgeführt, welche einer genaueren Erörterung bedürfen:<sup>87</sup>

### 1. Korrektheit

Ein Algorithmus muss die seiner Entwicklung zugrunde liegende Spezifikation erfüllen und ein korrektes Ergebnis gemäß der vorgegebenen Aufgabenstellung liefern.

### 2. Vollständigkeit

Ein Algorithmus darf bei der Ausführung keine Schritte oder Aktionen auslassen, wenn diese für die erfolgreiche Lösung der Aufgabe relevant sind. Weiterhin müssen zugewiesene Datenstrukturen für die Ausführung des Algorithmus zu jeder Zeit in korrekter und vollständiger Form vorliegen.

### 3. Eindeutigkeit

Jede Aktion innerhalb des Algorithmus muss eindeutig interpretierbar und ausführbar sein, d. h., dass die Architektur und Codierung des Algorithmus eindeutig definiert ist. Ebenso muss bei wiederholter Ausführung des Algorithmus unter gleicher Problemstellung immer dieselbe Lösung ermittelt werden.

### 4. Endlichkeit

Unabhängig von der Sprache oder Notation muss der Umfang eines Algorithmus endlich sein und darf für die Speicherung in Form einer Datei nur eine bestimmte Speicherkapazität erfordern.

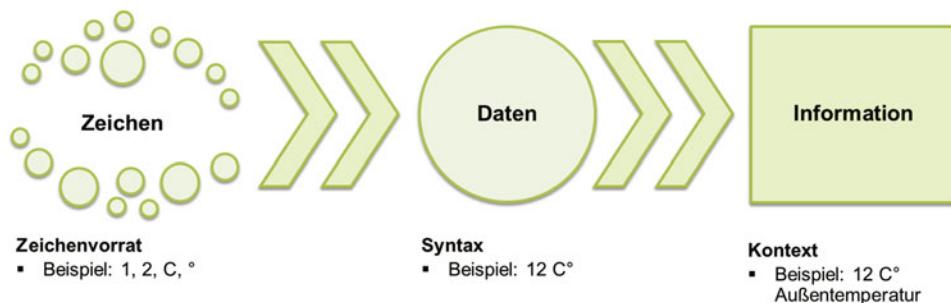
---

<sup>84</sup> Pomberger und Pree 2004, S. 67.

<sup>85</sup> Vgl. Hering et al. 2000, S. 7.

<sup>86</sup> Vgl. Pomberger und Pree 2004, S. 67.

<sup>87</sup> Vgl. Pomberger und Dobler 2008, S. 33 f.



**Abb. 4.9** Zusammenhänge zwischen Zeichen, Daten und Information. (eigene Erstellung, in Anlehnung an Krcmar 2010, S. 14)

Für die vorliegende Arbeit kann die Thematik von Algorithmen wie folgt zusammengefasst werden:

► „Ein **Algorithmus** ist eine allgemeingültige Rechenvorschrift, die aus mehreren elementaren Anweisungen besteht, die in einer systematischen Reihenfolge ausgeführt werden müssen und nach einer endlichen Anzahl an Iterationen zu einem korrekten und eindeutigen Ergebnis führt.“

Wie bereits beschrieben werden Algorithmen mithilfe entsprechender Softwareprogramme umgesetzt, sodass ein Computersystem die darin enthaltenen Anweisungen einlesen und abarbeiten kann. Für die Umsetzung verwendet ein Programm entsprechende Daten- und Programmstrukturen, auf die im Folgenden näher eingegangen werden.

#### 4.4.1.2 Daten, Datenstrukturen und Operationen

Durch die Nutzung von ITK zur Unterstützung der Daten- und Informationsverarbeitung erfolgt die Betrachtung von Zeichen, Daten, Informationen und Wissen. Hierbei entsteht der Eindruck, dass die jeweiligen Begriffe jeweils gleichen Bedeutungen entsprechen. Obwohl Beziehungen zwischen den Begrifflichkeiten bestehen, können sie keineswegs synonym zueinander verwendet werden. Abbildung 4.9 stellt die Beziehungen zwischen den Begriffen hierarchisch dar.

Auf der untersten Ebene befindet sich eine unbestimmte Anzahl an verschiedenen Zeichen, welche zunächst in einer unsystematischen Reihenfolge zueinanderstehen. Die Entstehung von Daten oder Datenbeständen ist auf die Zuordnung von Zeichen zu einem eindeutigen Alphabet zurückzuführen. Die Anreicherung der Daten mit zusätzlichem Kontext verschafft den Daten Bedeutung, sodass daraus nützliche Informationen entstehen. Die Vernetzung von Informationen führt zur Entstehung von Wissen.<sup>88</sup> Nachfolgende Ausführungen beziehen sich hauptsächlich auf die Betrachtung von Daten sowie Da-

<sup>88</sup> Vgl. Krcmar 2010, S. 14 f.

Datentyp	Wertebereich	Typische Werte	Operationen
Boolean	0 (False) und -1 (True)	false, true	and, or, not
Integer	-32.768 bis 32.767	-1, 0, 1, 10000, ...	+, -, *, /
Character	Je nach Zeichensatz	"+", "A", "a", "?", ...	and, or, not
String	0 bis ca. 2 Milliarden	"Wort", "Das ist ein Text"	length

**Abb. 4.10** Ausgewählte Datentypen zur Programmierung. (eigene Erstellung, in Anlehnung an Microsoft 2007)

tenstrukturen. Zwei unterschiedliche Ausprägungen von Datenobjekten können hierbei betrachtet werden:<sup>89</sup>

- Variablen, deren Werte während der Ausführung eines Algorithmus verändert werden.
- Konstanten, deren Wert nicht verändert werden kann.

Damit Datenobjekte in einen Algorithmus integriert werden können, müssen diese zunächst bestimmten Datentypen zugewiesen werden. Ein Datentyp definiert die Menge von Werten aus der ein Datenobjekt einen Wert annehmen kann sowie eine Menge von Operationen, die auf das entsprechende Datenobjekt erlaubt sind.<sup>90</sup> Datentypen können dadurch die Objekte der Anwendungswelt möglichst gut darstellen und weiterhin für die Bearbeitung in Rechnern realisiert werden. Abbildung 4.10 stellt eine Auswahl an Datentypen dar, welche in Bezug auf die vorliegende Arbeit notwendig sind.

Zur Deklaration von Datenobjekten muss neben der Festlegung der Art des Datenobjekts eine geeignete Bezeichnung angegeben werden, welche das Datenobjekt eindeutig beschreibt. Üblicherweise werden in der Programmierung hierfür kurze Bezeichner ausgewählt, die meist nur aus Buchstaben bestehen. Zusammengefasst besteht die Architektur eines Datenobjekts aus einem eindeutigen Bezeichner, einem Datentyp sowie einem Wert aus dem zugelassenen Wertebereich.<sup>91</sup> Innerhalb des nächsten Abschnitts erfolgt die Betrachtung und Erläuterung der Zustandsänderung von Datenobjekten.

#### 4.4.1.3 Wertzuweisungen und Kontrollstrukturen

Die imperitative Programmierung beschäftigt sich hauptsächlich mit der Zuweisung von Werten und Anweisungen zur Ermittlung von Werten. Zum besseren Verständnis des Vorgangs der Wertzuweisung soll folgende Schreibweise dienen:

► **Wertzuweisung:** Datenobjekt {Variable | Konstante} := Ausdruck {Operator | Operand}

<sup>89</sup> Vgl. Pomberger und Dobler 2008, S. 33.

<sup>90</sup> Vgl. Pomberger und Dobler 2008, S. 36.

<sup>91</sup> Vgl. Pomberger und Dobler 2008, S. 47.

Auf der linken Seite des Zuweisungssymbols (`:=`) wird der Bezeichner einer zuvor deklarierten Variable oder Konstante angegeben. Auf der rechten Seite wird der Ausdruck angegeben, welcher dem jeweiligen Datenobjekt zugewiesen werden soll. Ein Ausdruck definiert eine Vorschrift zur Berechnung eines neuen Wertes aus bereits ermittelten Werten von Datenobjekten. Ein Ausdruck kann weiterhin aus Variablen oder Konstanten bestehen, welche als Operanden bezeichnet werden, oder aus Operatoren, welche arithmetische Rechenvorschriften darstellen. Die Reihenfolge der Auswertung eines Ausdrucks erfolgt wie durch die Mathematik definiert und kann durch das Setzen von Klammern manipuliert werden. Grundsätzlich erfolgt zunächst die Auswertung des Ausdrucks auf der rechten Seite. In einem zweiten Schritt erfolgt die Zuweisung des ermittelten Wertes an das jeweilige Datenobjekt.<sup>92</sup> Nachfolgendes Beispiel soll die Funktionsweise einer Wertzuordnung aufzeigen und deren Auswirkungen erläutern:

### Wertzuweisung

1. `i := 25`
2. // Position 1
3. `i := i + 5`
4. // Position 2

Für das oben aufgeführte Beispiel wird angenommen, dass die Variable „`i`“ vom Datentyp „Integer“ ist. Die erste Anweisung in der ersten Zeile bewirkt, dass die Variable an der mit „Position 1“ gekennzeichneten Stelle den Wert „25“ besitzt. Die Auswertung der zweiten Anweisung führt dazu, dass zur Variablen „`i`“ die Zahl „5“ aufaddiert wird. Das Ergebnis der Auswertung wird der Variable „`i`“ übergeben. Der zuvor zugewiesene Wert wird durch den neuen Wert überschrieben, sodass an der mit „Position 2“ markierten Stelle die Variable „`i`“ den Wert „30“ aufweist.

Zur Lösung von Aufgaben mittels Algorithmen ist es oftmals notwendig, abhängig von der Erfüllung bzw. Nichterfüllung einer Bedingung, unterschiedliche Aktionen zu veranlassen. Zur Steuerung des Programmablaufs können hierfür bedingte oder wiederholende Verzweigungen implementiert werden. Letztere werden innerhalb der Softwareentwicklung auch als Schleifen bezeichnet. Eine Verzweigungsanweisung kann in folgender Schreibweise angegeben werden:

### Verzweigungsanweisung

```
if {Bedingung} then {Zuweisung 1} else {Zuweisung 2} end if
```

Zur besseren Lesbarkeit und zum Verständnis von Verzweigungen erfolgt die Aufteilung innerhalb von Algorithmen über mehrere Zeilen hinweg und nicht wie oben angegeben innerhalb einer Zeile. Die Ausführung einer Verzweigung beginnt mit der Auswertung

<sup>92</sup> Vgl. Pomberger und Dobler 2008, S. 48.

der Bedingung. Eine Bedingung muss zur korrekten Auswertung so formuliert sein, dass als Ergebnis ein Wahrheitswert (true oder false) geliefert wird. Je nach Ergebnis erfolgt die Ausführung einer Aktion (Zuweisung 1 oder Zuweisung 2). Die Verzweigung wird durch den Ausdruck „end if“ beendet. Der in obiger Schreibweise angegebene „else“-Teil ist optional anzusehen. Eine Verzweigung muss mindestens die Komponenten „if“, „then“ und „end if“ aufweisen.<sup>93</sup> Nachfolgend wird beispielhaft eine einfache Verzweigung dargestellt und erläutert:

### Verzweigung

1. if  $x < y$  then
2.  $x := x + 5$
3. else
4.  $x := x - 5$
5. end if

Zunächst muss die in der ersten Zeile aufgeführte Bedingung der Verzweigung ausgewertet werden. Hierzu wird geprüft, ob gegenwärtig die Variable „x“ kleiner als die Variable „y“ ist. Ist dies der Fall, die Bedingung somit wahr, wird der Wert der Variablen „x“ um die Zahl „5“ erhöht. Ist die Bedingung falsch, wird der Wert der Variablen „x“ um die Zahl „5“ vermindert.

Für die Lösung mehrerer aufeinanderfolgender Aufgaben bietet sich der Einsatz von Schleifen an. Schleifen dienen, abhängig von der Erfüllung bzw. Nichterfüllung einer Bedingung, zur wiederholten Ausführung festgelegter Anweisungsfolgen. Hierzu können kopfgesteuerte Schleifen, welche eine Bedingung zum Schleifenanfang auswerten, und fußgesteuerte Schleifen, welche eine Bedingung zum Schleifenende auswerten, unterschieden werden. Die einfachste Schleifenform ist die While-Schleife, für die folgende Schreibweise angegeben werden kann:

### Schleifenarten

- **Kopfgesteuerte Schleife:** while {Bedingung} do {Zuweisung 1, . . . , n} end while
- **Fußgesteuerte Schleife:** do {Zuweisung 1, . . . , n} while {Bedingung} end while

Im Zuge der kopfgesteuerten Schleife erfolgt zunächst die Überprüfung und Ausführung der Bedingung. Bei erfolgreicher Prüfung werden die nachfolgenden Zuweisungen ausgeführt. Diese werden solange wiederholt, bis die Bedingung zu einem Abbruch der Schleife führt. Da die Bedingungsprüfung zum Schleifenanfang durchgeführt wird, kann bereits ein Abbruch vor dem ersten Durchlaufen der Schleife angeordnet werden. Im Gegensatz hierzu wird die fußgesteuerte Schleife mindestens einmal durchlaufen bevor ein Abbruch

<sup>93</sup> Vgl. Pomberger und Dobler 2008, S. 50.

der Schleife erfolgen kann.<sup>94</sup> Nachfolgendes Beispiel soll zur Illustration der Anwendung einer Schleife dienen:

#### While-Schleife

1. summe := 0
2. n := 3
3. i := 1
4. while i <= n do
5. summe := summe + i
6. i := i + 1
7. end while

Für das obige Beispiel wird angenommen, dass die Variablen „summe“, „n“, und „i“ vom Datentyp „Integer“ sind. Die Bedingungsprüfung in der vierten Zeile ermöglicht das Durchlaufen der Schleife nur, wenn die Variable „i“ kleiner oder gleich der Variablen „n“ ist. Ist die Bedingung wahr, erfolgt in der fünften Zeile die Erhöhung des Wertes der Variablen „summe“ um den Wert der Variablen „i“. In der nachfolgenden Zeile wird die Variable „i“ um eins erhöht. Diese Anweisung ist notwendig, damit die Schleife nach endlichen Schritten beendet werden kann und nicht endlos durchlaufen wird. Nach insgesamt „n“-Schritten wird die Schleife beendet, da die Bedingung erfüllt ist. „Ergebnis der oben aufgeführten Summenberechnung ist der Wert „6“.

In den vorangegangenen Abschnitten wurden einige grundlegende Konstrukte der Programmierung angegeben. Für die Entwicklung von Algorithmen zur Softwareherstellung müssen weiterhin Programmiersprachen berücksichtigt werden, welche für den Aufbau und die Beschreibung von Algorithmen ein einheitliches Alphabet zur Verfügung stellen.

### 4.4.2 Programmiersprachen zur mobilen Applikationsentwicklung

Die Heterogenität mobiler Plattformen stellt eine besondere Herausforderung bei der Entwicklung mobiler Applikationen dar. Kenntnisse über verschiedenartige Betriebssysteme, Entwicklungsumgebungen, Programmierschnittstellen und sprachen sind notwendig, um eine möglichst große Anzahl an Endgeräten zu unterstützen. Innerhalb des folgenden Kapitels werden Programmiersprachen zur Applikationsentwicklung behandelt und vorgestellt. Zuvor wird der Begriff Programmiersprache definiert und der typische Aufbau dargestellt.

Im Vergleich zur Kommunikation zwischen Menschen stellen Programmiersprachen künstliche Sprachen dar, welche zur Kommunikation zwischen Menschen und Computern sowie zur Informationsverarbeitung dienen. Die Autoren Hering, Gutekunst und Dyllong definieren den Begriff Programmiersprache wie folgt.<sup>95</sup>

<sup>94</sup> Vgl. Pomberger und Dobler 2008, S. 52.

<sup>95</sup> Hering et al. 2000, S. 302.

► „**Programmiersprachen** dienen dazu, Aufgaben mit Hilfe von Sprachelementen so zu formulieren, dass sie mit dem Rechner gelöst werden können.“

Eine weitere Definition wird von Kannengiesser gegeben:<sup>96</sup>

► „Eine **Programmiersprache** ist eine formale Sprache, die zur Erstellung von Verarbeitungsanweisungen für Rechnersysteme verwendet wird, und richtet sich in Form und Funktion als Sprache an die Struktur und Bedeutung von Information.“

Aus diesen Definitionen lässt sich ableiten, dass Softwareentwickler mithilfe einer Programmiersprache ein für den Computer verständliches Programm formulieren. In diesem Zusammenhang lässt sich der Begriff Programmierung folgendermaßen definieren:<sup>97</sup>

► „Der Begriff der **Programmierung** beschreibt die Umsetzung der funktionalen Beschreibung eines Softwaresystems in den Quelltext einer bestimmten Programmiersprache.“

Programmiersprachen ermöglichen die Formulierung von Algorithmen in einen für den Computer verständlichen Code. Syntax und Semantik einer Programmiersprache müssen für eine erfolgreiche Umsetzung eingehalten werden. Die Syntax einer Sprache formuliert gültige Folgen von Zeichenketten mithilfe kontextfreier Grammatiken und definiert somit alle zulässigen Wörter, die durch eine Programmiersprache formuliert werden können. Die Semantik beschreibt, welche Bedeutung die einzelnen Wörter der Syntaxdefinition besitzen. Die Beschreibung der Semantik erfolgt für die meisten Programmiersprachen auf textueller Grundlage.<sup>98</sup>

Programmiersprachen lassen sich weiterhin, je nach Abstraktionsgrad und Struktur, in folgende Bereiche einteilen:<sup>99</sup>

- **Maschinensprachen** beschreiben eine Reihenfolge an einzelnen Befehlen, welche durch die Hardware eines Prozessors festgelegt ist.
- **Assemblersprachen** unterscheiden sich von Maschinensprachen durch die Anwendung von Befehlwörtern mit zugeordneten Parametern.
- **Höhere Programmiersprachen** beschreiben algorithmische Verfahren in einer rechnerunabhängigen Form.
- **Anwendungsorientierte Sprachen** enthalten Syntax und Semantik für einen eingeschränkten Anwendungsbereich.
- **Dokumentenbeschreibungssprachen** beschreiben die logische Struktur von Textdokumenten.

In Bezug auf den Umgang mit zu verarbeitenden Daten und Operationen werden in der Literatur weiterhin zwei unterschiedliche Programmierparadigmen betrachtet. Das im-

---

<sup>96</sup> Kannengiesser 2007, S. 22.

<sup>97</sup> Henning et al. 2007, S. 19.

<sup>98</sup> Vgl. Henning et al. 2007, S. 10.

<sup>99</sup> Vgl. Victor 2007, S. 199.

perative Programmierparadigma bezieht sich auf Sprachen, welche aus einer Folge von Anweisungen bestehen, die streng sequenziell abgearbeitet werden. Imperative Sprachen sind weiterhin durch die Verwendung von Funktionen und Prozeduren gekennzeichnet. Im Gegensatz zu diesen prozeduralen Sprachen entwickelten sich objektorientierte Sprachen, welche die Bearbeitung von Daten und Befehlen durch die Nutzung von Objekten realisieren.<sup>100</sup>

Programmiersprachen stellen in erster Linie Werkzeuge der Softwareentwickler dar. Neben der Erfahrung sowie methodischem Vorgehen bei der Softwareentwicklung, ist die Auswahl der richtigen Programmiersprache entscheidend für den Projekterfolg. Nachfolgend werden aktuelle Programmiersprachen zur Softwareentwicklung vorgestellt und näher beschrieben.

#### 4.4.2.1 Java zur Entwicklung von Android-Applikationen

Die Programmiersprache Java hat sich seit der Veröffentlichung im Jahr 1996 zu einer umfangreichen und leistungsstarken Softwaretechnologie entwickelt. In die Entwicklung von Java ist hauptsächlich die Firma Sun Microsystems involviert, welche bereits seit 1991 unter der Bezeichnung Green-Project erste Prototypen der Programmiersprache testete. Ursprüngliches Ziel der Firma war die Bereitstellung einer einfachen Programmiersprache mit grafischer Benutzeroberfläche für den Einsatz in Haushalts- und Unterhaltungselektronik. Mit dem Aufkommen des Internets änderte sich die Zielrichtung der Entwicklung auf die Unterstützung von Internetseiten durch grafikfähige und interaktive Programmelemente. Java weist eine Reihe von Eigenschaften auf, auf die nachfolgend kurz eingegangen wird:<sup>101</sup>

- **Einfachheit**

Java stellt eine einfache Programmiersprache dar, welche aus wenigen Sprachkonstrukten besteht und auf komplexe Konzepte, wie bspw. die Mehrfachvererbung, verzichtet.

- **Objektorientiert**

Java stellt eine vollständige, objektorientierte Sprache dar und verweigert den Aufruf von Unterprogrammen oder Prozeduren.

- **Verteilt**

Java ermöglicht die Realisierung von Anwendungen, welche innerhalb einer gemeinsamen Netzstruktur auf verteilte Ressourcen zugreifen können.

- **Sicherheit**

Für die Nutzung verteilter Anwendungen über das Internet stellt Java verschiedene Sicherheitsmechanismen für den Schutz vor unerlaubten Zugriffen zur Verfügung.

- **Architekturneutral**

Durch Java erstellte und übersetzte Programme können auf jeder Rechnerarchitektur ausgeführt werden, auf der ein Java-Laufzeitsystem vorhanden ist.

---

<sup>100</sup> Vgl. Kannengiesser 2007, S. 32.

<sup>101</sup> Vgl. Kröckertskothen 2005, S. 13 ff.

Zur Erstellung und Anwendung von Java-Programmen werden ein Editor zur Eingabe des Programmquelltextes, ein Compiler zum Übersetzen des Programms sowie ein Java-Laufzeitsystem zur Programmausführung benötigt. Zur Programmierung von mobilen Applikationen für das Betriebssystem Android müssen zusätzliche Java-Bibliotheken in die Laufzeitumgebung eingebunden werden. Diese Bibliotheken enthalten spezielle Klassen zur Kommunikation mit dem Android-Betriebssystem sowie mit der Hardware des jeweiligen Endgerätes.<sup>102</sup>

#### **4.4.2.2 C# zur Entwicklung von Windows-Phone-Applikationen**

Die Programmiersprache C ist eine von Dennis Richie und Brian Kernighan im Jahre 1978 veröffentlichte Sprache, die an den Bell Laboratories für die Systemprogrammierung des Betriebssystems Unix entwickelt wurde. Aufgabe der Sprache war zunächst die Bearbeitung nichtnumerischer Probleme für die Maschinenprogrammierung. C zählt zu den imperativen Programmiersprachen und besitzt ebenfalls, wie die Sprache Java, eine Funktionsbibliothek, die durch spezielle Bibliotheken beliebig modifiziert werden kann.<sup>103</sup>

Im Laufe der Zeit haben sich viele weitere Sprachen aus der ursprünglichen C-Sprache entwickelt, wie bspw. C++ oder C# (sprich: C sharp). Letztere stellt eine objektorientierte Version der C-Sprache dar und bietet eine Reihe an Eigenschaften, Methoden und Attributen zur Entwicklung komponentenorientierter Programme. Microsoft stellt über das sogenannte .NET Framework (sprich: dot Net) Klassenbibliotheken, Programmierschnittstellen und Dienstprogramme zur Entwicklung und Ausführung von Anwendungsprogrammen auf Basis von C# zur Verfügung. Das .NET Framework unterstützt weiterhin die Entwicklung mobiler Anwendungen für das Betriebssystem Windows Phone.<sup>104</sup>

Zur Unterstützung der objektorientierten Programmierung können folgende Eigenschaften von C# aufgezeigt werden:<sup>105</sup>

- **Datenabstraktion**

Durch C# erfolgt die Bildung von Klassen zur eindeutigen Beschreibung von Datenobjekten.

- **Datenkapselung**

C# ermöglicht durch die sogenannte Kapselung von Daten den kontrollierten Zugriff auf die Werte und Informationen von Datenobjekten.

- **Vererbung**

Durch die Bildung abgeleiteter Klassen in C# ist im Gegensatz zu Java die Durchführung einer mehrfachen Vererbung möglich.

---

<sup>102</sup> Vgl. Felker 2011, S. 47.

<sup>103</sup> Vgl. Erlenkötter 2005, S. 11 f.

<sup>104</sup> Vgl. Microsoft 2013b.

<sup>105</sup> Vgl. Louis 2010, S. 26 ff.

- **Polymorphie**

C# erlaubt die Implementierung von Anweisungen, die zu jeweils unterschiedlicher Laufzeit verschiedene Wirkungen haben können.

Weitere Möglichkeiten der C#-orientierten Anwendungsentwicklung bestehen in der effizienten und maschinennahen Programmierung, der Umsetzung universeller und modularer Programme sowie der Übertragbarkeit von Programmen auf verschiedene Endgeräte.<sup>106</sup>

#### 4.4.2.3 Objective-C zur Entwicklung von iOS-Applikationen

Ebenfalls wie C# stellt Objective-C eine Erweiterung der Programmiersprache C dar und wurde von Brad Cox und Tom Love zu Beginn der 80er-Jahre entwickelt. Objective-C ist eine strikte Obermenge der Programmiersprache C und erweitert den Sprachumfang um objektorientierte Elemente und erlaubt weiterhin die nahtlose Vermischung von C- und Objective-C-Syntax. Objective-C stellt somit eine hybride Programmiersprache dar, welche die Verwendung von imperativen und objektorientierten Programmierparadigmen erlaubt. Objective-C wurde für den Einsatz auf dem Betriebssystem NextStep vorgesehen und liefert heute die Grundlage für Apples Betriebssysteme Mac OS X und iOS.<sup>107</sup>

Die auch in Objective-C mögliche Klassenbildung erlaubt es dem Entwickler, die Repräsentation und das Verhalten von Objekten zu spezifizieren. Durch die Definition von Klassen erfolgt weiterhin die Festlegung von Zugriffs- und Manipulationsmöglichkeiten auf das Objekt bzw. seinen Zustand. Hierfür werden Instanzvariablen eingesetzt, denen weiterhin unterschiedliche Eigenschaften und Zugriffsrechte auf die Objekte innewohnen.<sup>108</sup> Aufgrund des hybriden Ansatzes weist Objective-C Eigenschaften aus den imperativen und objektorientierten Programmierparadigmen auf. Im Vergleich zu Java und C# können jedoch in Bezug auf die Implementierung mit Objective-C wesentliche Unterschiede festgestellt werden, bspw. das fehlende Speichermanagement für Datenobjekte sowie das Senden von Nachrichten für den Methodenaufruf.<sup>109</sup>

#### 4.4.2.4 HTML 5 zur Entwicklung webbasierter Applikationen

Durch die Entstehung des Internets war es erstmals möglich, Informationen über ein Netzwerk von Computern zu verteilen. Zu Beginn noch in Größe und Bandbreite begrenzt, ermöglichten Hochgeschwindigkeitsmodems dem damals noch so betitelten ARPAnet (Advanced Research Projects Agency Network of the Department of Defense) einen technischen Aufstieg. Erstmals konnten neben Rüstungsfirmen und akademischen Institutionen auch Einzelpersonen und Unternehmen im Netzwerk digital kommunizieren.

<sup>106</sup> Vgl. Fuchs 2007, S. 139.

<sup>107</sup> Vgl. Gall et al. 1995, S. 32 f.

<sup>108</sup> Vgl. Gall et al. 1995, S. 32 f.

<sup>109</sup> Vgl. Mark et al. 2011, S. 8.

Dennoch fehlte es an standardisierten Funktionen zum Austausch von Dokumenten, Bildern oder Audiodateien. Diesem Problem nahmen sich Wissenschaftler der Europäischen Organisation für Kernforschung (CERN) an und entwickelten 1989 die Hypertext-Markup-Language (HTML) für den Austausch von Forschungsergebnissen zwischen den Standorten in Frankreich und der Schweiz.<sup>110</sup>

Das anfängliche Ziel von HTML beinhaltete somit den Austausch von Informationen über einen einfachen und strukturierten digitalen Weg. Aktuell erfolgreiche Webtechnologien verlangen jedoch mehr als diesen ursprünglichen Ansatz. Aus diesem Grund wurde die HTML-Sprache seit der Veröffentlichung ständig weiterentwickelt und befindet sich gegenwärtig in der fünften Version. Empfehlungen zur Weiterentwicklung von HTML werden durch das World Wide Web Consortium (W3C) an die Browserhersteller herausgegeben. HTML stellt als sogenannte Auszeichnungssprache eine Besonderheit unter den Programmiersprachen dar, da sie als solche nicht programmiert sondern geschrieben wird.<sup>111</sup>

Aufbauend auf der Internettechnologie und der damit verbundenen Client-Server-Architektur beruhen webbasierte Applikationen auf HTML-gestützten Webseiten, die zur Ausführung und Nutzung einen Internetbrowser benötigen. Im Gegensatz zu herkömmlichen Webseiten unterscheiden sich mobile Web-Applikationen darin, dass sie ein ähnliches Aussehen und Verhalten wie bei einer nativen Anwendung erreichen. Die Entwicklung webbasierter Applikationen kann weiterhin von den folgenden Eigenschaften von HTML profitieren:<sup>112</sup>

- **Offenheit**

Die Weiterentwicklung von HTML wird ständig durch das W3C vorangetrieben. Zudem besteht die Möglichkeit, eigene Verbesserungsvorschläge und Ideen zur Entwicklung von HTML an die Browserhersteller zu kommunizieren.

- **Plattformunabhängigkeit**

Durch Nutzung der Internettechnologie ermöglicht HTML eine plattformunabhängige Anwendungsentwicklung. Somit können webbasierte Applikationen auf verschiedenen Betriebssystemen zum Einsatz kommen.

- **Flexibilität**

Durch eine getrennte Sicht auf Inhalt, Logik und Design können durch den Einsatz von HTML große Teile der entwickelten Anwendungen auf verschiedenen Endgeräten verwendet werden.

Webbasierte Applikationen unterliegen jedoch Einschränkungen, wie bspw. dem begrenzten Zugang zu den Hardwareressourcen mobiler Geräte sowie der geringen Performance bei multimedialen Anwendungen.<sup>113</sup>

---

<sup>110</sup> Vgl. Muscaino und Kennedy 2003, S. 3.

<sup>111</sup> Vgl. Albert und Stiller 2012, S. 150.

<sup>112</sup> Vgl. Albert und Stiller 2012, S. 150.

<sup>113</sup> Vgl. Albert und Stiller 2012, S. 158.

Innerhalb des vorliegenden Kapitels wurden Programmiersprachen zur Umsetzung nativer, hybrider oder webbasierter Applikationen vorgestellt und beschrieben. Für die Umsetzung dieser Sprachen zu einer lauffähigen Anwendung erfolgt der Einsatz adäquater Entwicklungsumgebungen. Eine Auswahl und Erklärung gegenwärtiger Entwicklungsumgebungen wird nachfolgend gegeben.

#### 4.4.3 Entwicklungsumgebungen für mobile Applikationen

Die Verwendung von Werkzeugen ist ein wesentliches Merkmal des ingenieurmäßigen Vorgehens und wird zur Steigerung der Effektivität und Effizienz von Entwicklern im Bereich des Software Engineering angestrebt. Das Ergebnis der Softwareentwicklung ist jedoch immer von der Anwendung und Nutzung des Werkzeugs abhängig.<sup>114</sup> Ein einfaches Werkzeug zur Entwicklung von Softwareanwendungen stellen sogenannte Editoren dar, welche die Entwickler bei der Umsetzung von Programmcode unterstützen. Für die Umsetzung in ein endgültiges Anwendungsprogramm werden neben dem Editor weiterhin ein Übersetzer sowie ein Binder benötigt. Diese Werkzeuge dienen der Transformierung und Zusammenführung von Programmcode in eine einheitliche und ausführbare Form.<sup>115</sup>

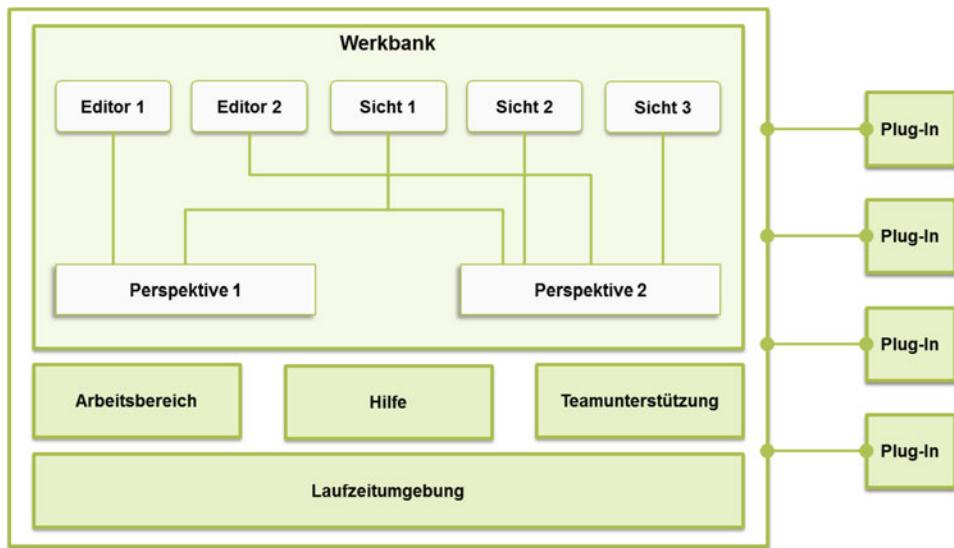
Integrierte Entwicklungsumgebungen vereinen Editoren, Übersetzer und Binder zu einem einheitlichen Werkzeug und unterstützen die Entwickler neben der reinen Programmieraktivität durch die Spezifikation, den Entwurf sowie den Test von Anwendungssoftware. Die einzelnen Werkzeuge können weiterhin durch den Entwickler aufgerufen und Einstellungen an diesen über eine grafische Benutzeroberfläche vorgenommen werden. Die Regelung und Steuerung des Kontroll- und Datenflusses zwischen den einzelnen Werkzeugen wird durch die Entwicklungsumgebungen vorgenommen.<sup>116</sup> Der allgemeingültige Aufbau von Entwicklungsumgebungen wird in Abb. 4.11 aufgezeigt und anschließend erläutert.

Der Arbeitsbereich einer Entwicklungsumgebung beinhaltet alle Projekte, an denen ein Entwickler beteiligt ist. Jedes vorhandene Projekt definiert ein oder mehrere Unterverzeichnisse sowie die darin enthaltenen Daten. Eine allgemeine Struktur für die Interaktion mit dem Benutzer der Entwicklungsumgebung wird durch die Werkbank in Form einer grafischen Benutzerschnittstelle zur Verfügung gestellt. Unabhängig vom jeweiligen Betriebssystem, stützt sich die Implementierung der Werkbank auf vorgegebene Programmobilotheken und setzt sich weiterhin aus mindestens einem Editor sowie mehreren Sichten zusammen. Der Inhalt einer Werkbank wird durch die jeweilige Perspektive bestimmt. Innerhalb der Entwicklungsumgebung ermöglicht die Teamunterstützung eine Anbindung an ein Versionskontrollsystem, welches Komponenten und Funktionen für

<sup>114</sup> Vgl. Balzert 2009, S. 59 f.

<sup>115</sup> Vgl. Balzert 2009, S. 60.

<sup>116</sup> Vgl. Balzert 2009, S. 76.



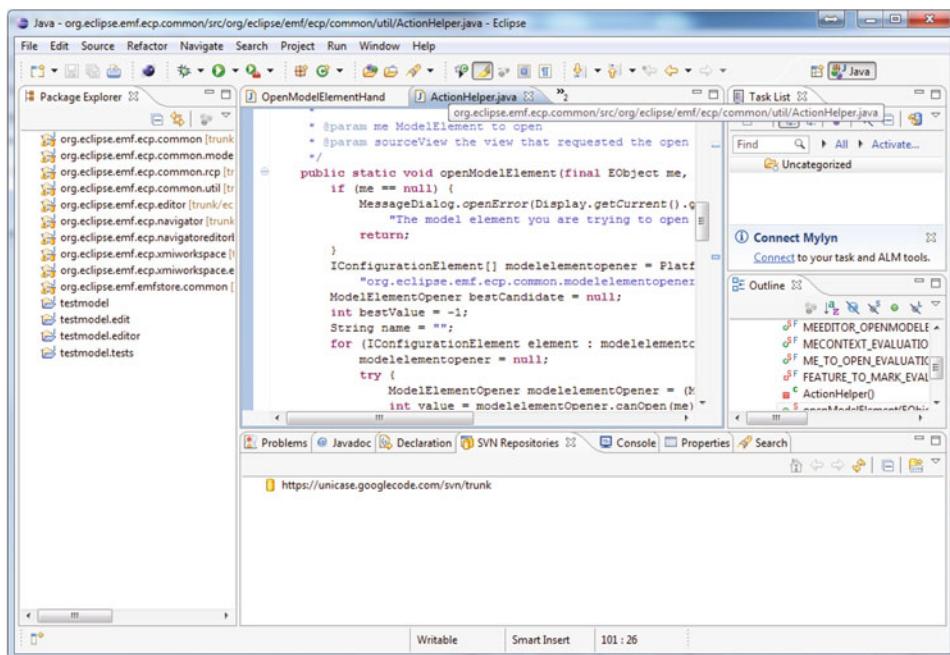
**Abb. 4.11** Aufbau einer Entwicklungsumgebung. (eigene Erstellung, in Anlehnung an Balzert 2009, S. 22)

die Versionierung und Zusammenarbeit in Teams ermöglicht. Weiterhin ist eine Hilfe-komponente in die Entwicklungsumgebung integriert, die Dokumentationen über die Anwendung von Werkzeugen bereitstellt. Die Entwicklungsumgebung kann durch mehrere Plug-ins erweitert werden, bspw. um Programmabibliotheken oder weitere Entwicklungs-werkzeuge. Die Laufzeitumgebung stellt die Basis einer Entwicklungsumgebung dar und ermöglicht die plattformunabhängige Ausführung erstellter Programme.<sup>117</sup> Nachfolgend werden Entwicklungsumgebungen für die mobile Applikationsentwicklung vorgestellt. Die Betrachtung der Entwicklungsumgebungen erfolgt, je nach Anwendungsbereich, zur Entwicklung von Android-, iOS- oder Windows-Phone-Applikationen.

#### 4.4.3.1 Entwicklungsumgebungen für Android-basierte Applikationen

Insbesondere für die Android-Entwicklung hat sich die Eclipse-Entwicklungsumgebung als Standard etabliert. Eclipse ist unter der Open-Source-Lizenz frei verfügbar und wurde ursprünglich als integrierte Entwicklungsumgebung für die Programmiersprache Java genutzt. Aufgrund der offenen Struktur von Eclipse ist seit der Version 3.0 die Einbindung von Plug-ins zur Erweiterung des Funktionsumfanges erlaubt. Mittlerweile existiert eine Vielzahl an quelloffenen sowie kommerziellen Erweiterungen für Eclipse, sodass die Entwicklungsumgebung auch für viele weitere Programmiersprachen und

<sup>117</sup> Vgl. Balzert 2009, S. 77 f.



**Abb. 4.12** Grafische Benutzeroberfläche von Eclipse. (Behrens et al. 2011)

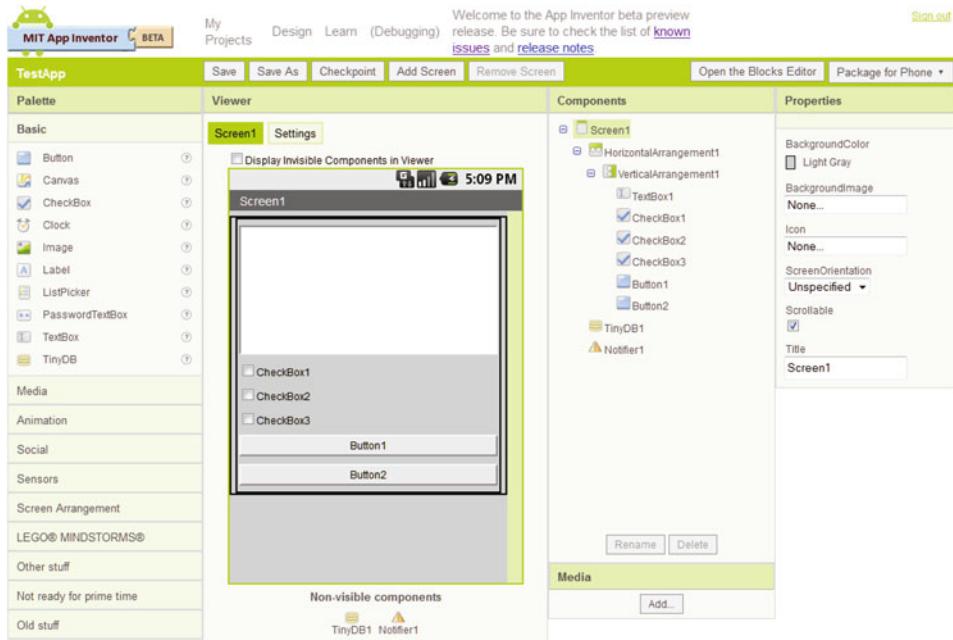
Entwicklungsaufgaben eingesetzt werden kann. Eclipse wird von der Eclipse Foundation weiterentwickelt und aktuell in der Version 4.2 angeboten (Abb. 4.12).<sup>118</sup>

Zur Entwicklung mobiler Applikationen für das Betriebssystem Android bietet der Hersteller Google eine eigene webbasierte Entwicklungsumgebung namens App Inventor an. Hintergrund der Entwicklung von App Inventor war die Herstellung eines experimentellen Lehr- und Lernwerkzeuges zur Erstellung lauffähiger mobiler Anwendungen für einen ausgewählten Kreis amerikanischer Hochschulen. App Inventor zielte somit darauf ab, Schüler und Studenten einen leichten Einstieg in das Programmieren von Anwendungen für mobile Endgeräte zu ermöglichen. Nachdem die Entwicklungsumgebung unter realistischen Bedingungen getestet, optimiert und letztendlich auf eine stabile Basisversion gesetzt wurde, kündigte Google im Dezember 2010 die Öffnung der Entwicklungsplattform für jeden interessierten Benutzer ohne jegliche Zulassungsbeschränkung an.<sup>119</sup>

Für die Erstellung mobiler Applikationen mittels App Inventor sind zwei geteilte Arbeitsschritte notwendig. Im ersten Schritt erfolgt durch den Entwickler die Gestaltung der Applikation. Hierzu wird der sogenannte Designer verwendet, welcher über eine Browseranwendung gestartet wird. Zur Gestaltung der Applikation müssen aus einer integrierten Funktionsbibliothek gewünschte Steuerelemente, bspw. Textfelder, Schaltflächen

<sup>118</sup> Vgl. Behrens et al. 2011, S. 63.

<sup>119</sup> Vgl. Kloss 2011, S. 23 ff.



**Abb. 4.13** Designansicht des App Inventors

oder Beschriftungen auf die zukünftige Programmoberfläche der Applikation gezogen, benannt und positioniert werden.<sup>120</sup> Der zweite Schritt besteht in der Bearbeitung der Steuerelemente durch die Zuweisung von Methoden und Funktionen. Für die Herstellung lauffähiger Algorithmen müssen hierfür in einem separaten Editor funktionale Blöcke untereinander verbunden werden, die, ähnlich dem Befehlssatz einer klassischen Programmiersprache, die Syntax der Entwicklungssprache des App Inventors darstellen. Die Entwicklung mobiler Applikationen beruht somit auf dem systematischen Zusammenwirken der Steuerelemente aus dem Designer sowie der funktionalen Blöcke aus dem sogenannten Blocks-Editor.<sup>121</sup> Abbildung 4.13 bildet den Designer des App Inventors ab.

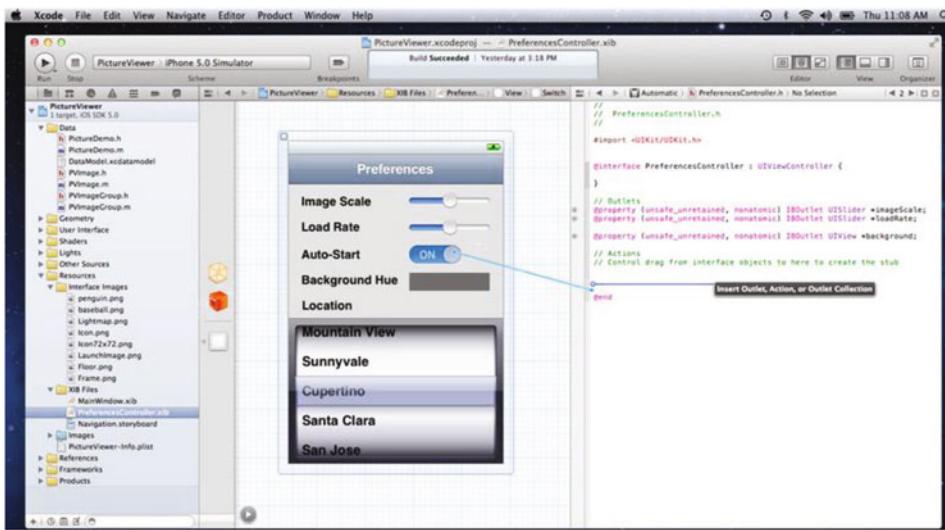
Der App Inventor stellt somit eine Alternative zur herkömmlichen Java-Programmierung über Eclipse dar. Auf die Vor- und Nachteile beider Entwicklungsumgebungen wird innerhalb des Kapitel 7 nochmals eingegangen. Nachfolgend werden Entwicklungsumgebungen für iOS basierte Applikationen aufgezeigt.

#### 4.4.3.2 Entwicklungsumgebungen für iOS-basierte Applikationen

Die Programmierung von mobilen Applikationen für Apple-Betriebssysteme setzt den Einsatz der Entwicklungsumgebung Xcode voraus. Hinter dieser Bezeichnung verbirgt

<sup>120</sup> Vgl. Kloss 2011, S. 59 f.

<sup>121</sup> Vgl. Kloss 2011, S. 75 f.



**Abb. 4.14** Designansicht von Xcode. (Apple 2012)

sich eine integrierte Entwicklungsumgebung, die von Apple kostenlos zur Verfügung gestellt wird. Ebenso wie Eclipse ist Xcode eine Kombination aus Projektverwaltung, visuellem Gestaltungstool, Editor und Übersetzer. Weiterhin können Entwickler auf zusätzliche Tools zur Analyse des Programmcodes oder zur Dokumentation der Applikation zurückgreifen.<sup>122</sup>

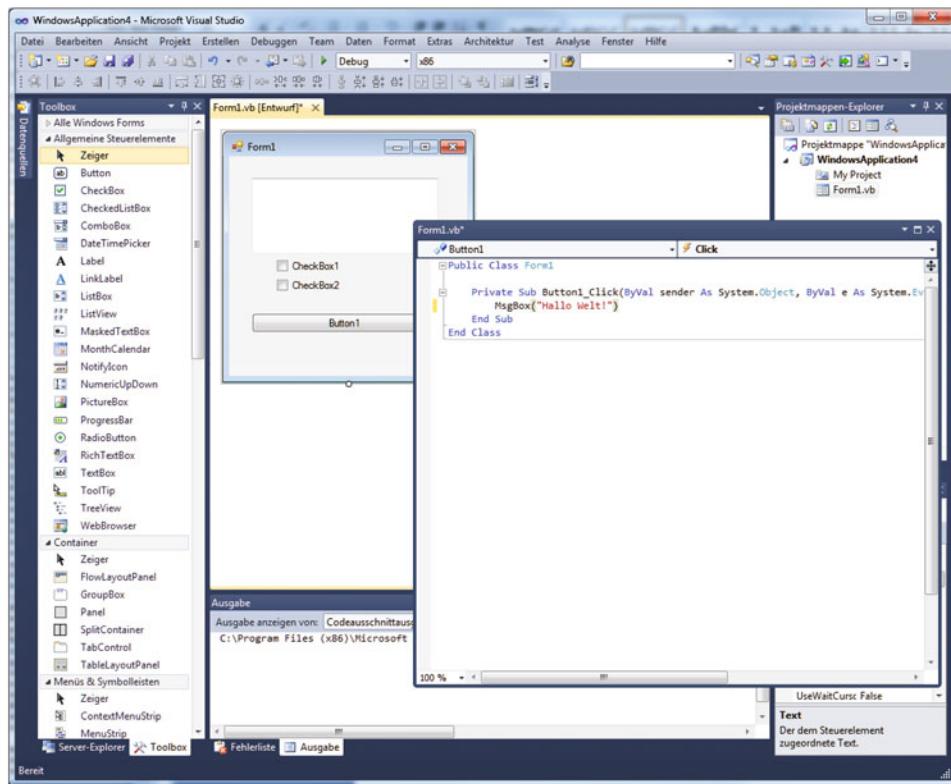
Der Aufbau der Xcode-Entwicklungsumgebung ähnelt dem Prinzip des App Inventors. Die Gestaltung der Benutzeroberfläche wird über den in Xcode integrierten Interface-Builder realisiert, welcher ebenfalls das Drag-and-drop-Verfahren unterstützt. Die Eingabe von Quellcode erfolgt in einem separaten Texteditor. Zur Programmierung von mobilen Applikationen mittels Xcode wird hauptsächlich die Programmiersprache Objective-C verwendet.<sup>123</sup> Die Erstellung des Layouts sowie die Programmierung des Codes waren ursprünglich auf unterschiedlichen Ebenen vorgesehen. Aktuell befindet sich Xcode in der Version 4.0, welche erstmals die beiden Sichten zu einem gemeinsamen Arbeitsbereich zusammenfügt.<sup>124</sup> Abbildung 4.14 zeigt die Entwicklungsumgebung Xcode.

Ein Nachteil in der Verwendung von Xcode ist, dass die Entwicklungsumgebung nur auf einem Apple-Rechner installiert und genutzt werden kann. Dies bedeutet, dass neben den üblichen Entwicklungskosten zusätzliche Kosten für die Anschaffung von notwendiger Hardware anfallen. Dies ist ebenso bei der Verwendung von Visual Studio der Fall, welche bei der Entwicklung von Windows-Phone-Applikationen zum Einsatz kommt. Für Visual

<sup>122</sup> Vgl. Hinzberg 2011, S. 31.

<sup>123</sup> Vgl. Böhme 2012, S. 24.

<sup>124</sup> Vgl. Apple 2012.



**Abb. 4.15** Designansicht von Visual Studio 2010

Studio wird das Betriebssystem Windows von Microsoft vorausgesetzt. Nachfolgend wird die Entwicklungsumgebung näher beschrieben.

#### 4.4.3.3 Entwicklungsumgebungen für Windows-Phone-basierte Applikationen

Visual Studio 2010 stellt ebenso wie Xcode und Eclipse eine integrierte Entwicklungsumgebung zur Programmierung von Anwendungssoftware zur Verfügung. Programmierer können mittels Visual Basic klassische Windows-Programme sowie dynamische Webseiten oder mobile Anwendungen unter Verwendung der Programmiersprachen C, C++ oder C# entwickeln. Für die individuelle Anpassung an die Arbeitsbedürfnisse können Entwickler benutzerdefinierte Modifikationen und Ergänzungen in die Entwicklungsumgebung integrieren oder die bereits vorhandenen Funktionen für das Dokumentieren, Analysieren und Testen der Software nutzen.<sup>125</sup>

<sup>125</sup> Vgl. Microsoft 2013c.

Gegenwärtig werden vier unterschiedliche Versionen von Visual Studio ausgeliefert: Visual Studio Express, Professional, Premium und Ultimate. Die Express-Versionen der Entwicklungsumgebung sind kostenlos und im Funktionsumfang reduzierte Varianten. Sie sind zudem auf nur eine Programmiersprache beschränkt, bspw. Visual C# Express oder Visual C++ Express. Die Express-Versionen werden häufig zu Werbezwecken oder für den Einsatz in der Lehre verbreitet. Im Gegensatz hierzu können Entwickler mit Visual Studio Professional mehrere Programmiersprachen verwenden. Weiterhin besteht die Möglichkeit, Anwendungen für mobile Applikationen zu entwickeln. Die Premium und Ultimate-Varianten der Entwicklungsumgebung enthalten zusätzliche Funktionen und Methoden für die teambasierte Anwendungsentwicklung sowie zur Verwaltung des gesamten Lebenszyklus einer Applikation.<sup>126</sup> Anhand des einfachen Programms „Hallo Welt“ soll in Abb. 4.15 das Verständnis für die Eingabe des Quellcodes über den Texteditor sowie die Entwurfansicht zur Darstellung der Benutzeroberfläche von Visual Studio 2010 vereinfacht dargestellt werden.

Visual Studio 2010 kann neben der Installation auf einem stationären Rechner ebenfalls über eine Client-Server-Architektur betrieben werden und ermöglicht dadurch den Aufbau einer leistungsfähigen Infrastruktur zur Verwaltung und Bearbeitung von Softwareprojekten in verteilten Teams.<sup>127</sup>

---

## Literatur

- Abts, D., Mülner, W.: Grundkurs Wirtschaftsinformatik. Eine kompakte und praxisorientierte Einführung, 7. Aufl. Vieweg + Teubner, Wiesbaden (2010)
- Albert, K., Stiller, M.: Der Browser als mobile Plattform der Zukunft – Die Möglichkeiten von HTML5-Apps. Chancen und Grenzen der Entwicklung mobiler Anwendungen mit Hilfe von Web-standards. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 147–160. Springer, Berlin (2012)
- Apple Inc.: What's new in Xcode 4. <https://developer.apple.com/technologies/tools/whats-new.html> (2012). Zugegriffen 24. Sept. 2013
- Apple Inc.: iOS 7. Das mobile OS aus einer ganz neuen Perspektive. <http://www.apple.com/de/ios/> (2013). Zugegriffen 13. Okt. 2013
- Aßmann, U., Demuth, B., Hartmann, F.: Risiken in der Softwareentwicklung. Wissenschaftliche Zeitschrift der Technischen Universität Dresden 55(3–4), S. 105–109 (2006)
- Balzert, H.: Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering, 3. Aufl. Spektrum Akademischer, Heidelberg (2009)
- Behrens, G., Kuz, V., Behrens, R.: Softwareentwicklung von Telematikdiensten. Konzepte, Entwicklung und zukünftige Trends. Springer, Berlin (2011)
- Berger, S., Lehner, F.: Mobile B2B-Anwendungen. In: Hampe, J.F., Schwabe, G. (Hrsg.) Mobile and Collaborative Business 2002, Proceedings zu Teilkonferenz der Multikonferenz Wirtschaftsinformatik, S. 85–94. Nürnberg (2002)

---

<sup>126</sup> Vgl. Kotz 2011, S. 33.

<sup>127</sup> Vgl. Microsoft 2013c.

- Böhme, I.: iPhone- und iPad-Programmierung für Einsteiger. iOS-Apps entwickeln von Anfang an. Markt + Technik, München (2012)
- Book, M., Gruhn, V., Hülder, M., Schäfer, C.: Der Einfluss verschiedener Mobilitätsgrade auf die Architektur von Informationssystemen. In: Hampe, J.F., Lehner, F., Pousttchi, K., et al. (Hrsg.) Mobile Business – Process, Platforms, Payment, LNI-Proceedings zur 5. Konferenz Mobile Commerce Technologien und Anwendungen (MCTA), S. 117–130. Augsburg (2005)
- Brockhaus in der Wissenmedia: Brockhaus – Computer und Informationstechnologie, 1. Aufl. Bibliographisches Institut & F. A. Brockhaus AG, München (2003)
- Bundesamt für Sicherheit in der Informationstechnik (BSI): Mobile Endgeräte und mobile Applikationen: Sicherheitsgefährdungen und Schutzmaßnahmen, Bonn. [https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Broschueren/MobilEndgeraete/mobile\\_endgeraete\\_pdf.pdf?blob=publicationFile](https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Broschueren/MobilEndgeraete/mobile_endgeraete_pdf.pdf?blob=publicationFile) (2006). Zugegriffen 23. Sept. 2013
- Bundesamt für Sicherheit in der Informationstechnik (BSI): Mobile Endgeräte, Bonn. [https://www.bsi.bund.de/DE/Themen/weitereThemen/MobileSecurity/MobileEndgeraete/mobileendgeraete\\_node.html](https://www.bsi.bund.de/DE/Themen/weitereThemen/MobileSecurity/MobileEndgeraete/mobileendgeraete_node.html) (2012). Zugegriffen 23. Sept. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM): Apps-Boom geht weiter, Berlin. [http://www.bitkom.org/de/themen/64022\\_69195.aspx](http://www.bitkom.org/de/themen/64022_69195.aspx) (2011). Zugegriffen 23. Sept. 2013.
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM): Sprachtelefonie verlagert sich vom Festnetz ins Mobilnetz, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_71274.aspx](http://www.bitkom.org/de/markt_statistik/64046_71274.aspx) (2012a) Zugegriffen 23. Sept. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM): Neuer Rekord bei Mobilfunkanschlüssen in Deutschland, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_71315.aspx](http://www.bitkom.org/de/markt_statistik/64046_71315.aspx) (2012b). Zugegriffen 23. Sept. 2013
- Christmann, S., Hagenhoff, S.: Mobiles Internet im Business-to-Business-Bereich – Eine Fallstudienuntersuchung, Arbeitsbericht Nr. 4/2009. Institut für Wirtschaftsinformatik, Göttingen (2009)
- Christmann, S., Hagenhoff, S., Thorsten, C.: Webbasierte Anwendungen als Lösungsansatz für die Heterogenität im mobilen Internet, Arbeitsbericht Nr. 3/2010. Institut für Wirtschaftsinformatik, Göttingen (2010)
- Diederichs, H.: Komplexitätsreduktion in der Softwareentwicklung. Ein systemtheoretischer Ansatz. Books On Demand, Norderstedt (2004)
- Eckert, C., Schneider, C.: Smart Mobile Apps: Enabler oder Risiko? In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 193–209. Springer, Berlin (2012)
- Erlenkötter, H.: C Programmieren vom Anfang an, 10. Aufl. Rowohlt Taschenbuch, Hamburg (2005)
- Euler, M., Hacke, M., Hatherz, C., Steiner, S., Vercales, S.: Herausforderungen bei der Mobilisierung von Business Applikationen und erste Lösungsansätze. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 107–125. Springer, Berlin (2012)
- Felker, D.: Android Apps Entwicklung. Wiley-VCH, Weinheim (2011)
- Frohberg, D.: Mobile Learning, Wirtschaftswissenschaftliche Fakultät. Universität Zürich, Zürich (2008)
- Fuchß, T.: C#. In: Henning, P.A., Vogelsang, H. (Hrsg.) Handbuch Programmiersprachen. Softwareentwicklung zum Lernen und Nachschlagen, S. 138–175. Carl Hanser, München (2007)
- Fuchß, T.: Mobile Computing: Grundlagen und Konzepte für mobile Anwendungen. Carl Hanser, München (2009)
- Gall, H., Hauswirth, M., Klösch, R.: Objektorientierte Konzepte in Smalltalk, C++, Objective-C, Eiffel und Modula-3. Informatik Spektrum 18(4), S. 195–202 (1995)

- Golding, P.: Next Generation Wireless Applications: Creating Mobile Applications in a Web 2.0 and Mobile 2.0 World, 2. Aufl. Wiley-VCH, Weinheim (2008)
- Google Android Developer Portal: Android 4.2 Jelly Bean. <http://developer.android.com/break-about/versions/jelly-bean.html> (2013). Zugegriffen 25. Sept. 2013
- Hansen, H.-R., Neumann, G.: Wirtschaftsinformatik I: Grundlagen und Anwendungen, 9. Aufl. Lucius & Lucius, Stuttgart (2009)
- Heidemann, W.-R., Zumbruch, I.: Zertifizierte Apps: mehr Funktionalität, Sicherheit und Bedienungsfreundlichkeit. In: Vercales, S.; Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 241–252. Springer, Berlin (2012)
- Henning, P.A., Hoffmann, D.W., Vogelsang, H.: Grundlagen der Programmiersprachen. In: Henning, P.A., Vogelsang, H. (Hrsg.) Handbuch Programmiersprachen. Softwareentwicklung zum Lernen und Nachschlagen, S. 9–59. Carl Hanser, München (2007)
- Hering, E., Gutekunst, J., Dyllong, U.: Handbuch der praktischen und technischen Informatik, 2. Aufl. Springer, Berlin (2000)
- Hesse, W., Keutgen, H., Luft, A.L., Rombach, H.D.: Ein Begriffssystem für die Softwaretechnik. Informatik-Spektrum 7, S. 200–213 (1984)
- Hinzeberg, H.: Objective-C und Cocoa Praxiseinstieg: Programmierung für Mac OS X und iPhone. MITP, Heidelberg (2011)
- IBM: IBM Survey: IT Professionals Predict Mobile and Cloud Technologies Will Dominate Enterprise Computing by 2015, Armonk. <http://www-03.ibm.com/press/us/en/pressrelease/32674.wss> (2010). Zugegriffen 23. Sept. 2013
- Institute of Electrical and Electronics Engineers (IEEE): IEEE standard glossary of software engineering terminology, IEEE Std 610.12 (1990)
- Jobs, S.: Thoughts on flash. <http://www.apple.com/hotnews/thoughts-on-flash/> (2010). Zugegriffen 23. Sept. 2013
- Kallus, M.: Laptop, Smartphone, Tablet. Frankfurt Hauptstadt der Mobile-Diebe. <http://www.cio.de/knowledgecenter/security/2882363/index2.html> (2013). Zugegriffen 23. Sept. 2013
- Kannengiesser, M.: Objektorientierte Programmierung mit PHP5. Franzis, Poing (2007)
- Kloss, J.H.: Android-Apps. Mobile Anwendungen entwickeln mit App Inventor. Markt + Technik, München (2011)
- Kotz, J.: Erfolgreich Visual Basic 2010 programmieren. Addison-Wesley, München (2011)
- Krannich, D.: Mobile System Design. Herausforderungen, Anforderungen und Lösungsansätze für Design, Implementierung und Usability-Testing Mobiler Systeme. Books on Demand, Norderstedt (2010)
- Kraus, C.: Mobile Software – Grundlagen und Erfolgsfaktoren für Apps im Mobile Business. Technologie und Konzeptstudie. 2kit consulting, Oberhausen (2012)
- Krcmar, H.: Informationsmanagement, 5. Aufl. Springer, Berlin (2010)
- Kröckertschothen, T.: Java 2. Grundlagen und Einführung, Regionales Rechenzentrum für Niedersachsen, 4. Aufl. Hannover (2005)
- Kuassi, L., Bischel, M.: Anwendungssicht mobiler Geschäftsanwendungen. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 125–147. Springer, Berlin (2012)
- Kurbel, K.: Software. In: Enzyklopädie der Wirtschaftsinformatik, Artikel: Software. <http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wienzyklo-paedie/lexikon/technologien-methoden/Software/software/?searchterm=Software> (2013). Zugegriffen 23. Sept. 2013
- Lanninger, V.: Prozessmodell zur Auswahl Betrieblicher Standardanwendungssoftware für KMU. Josef Eul, Lohmar-Köln (2009)

- Lehner, F.: Grundlagen und Entwicklung – Einführung und Motivation. In: Teichmann, R., Lehner, F. (Hrsg.) Mobile Commerce. Strategien, Geschäftsmodelle, Fallstudien. Springer, Berlin (2002)
- Levi, P., Rembold, U.: Einführung in die Informatik für Naturwissenschaftler und Ingenieure, 4. Aufl. Carl Hanser, München (2003)
- Lonthoff, J.: Projekt mobiCOMP. In: von Knop, J., Haverkamp, W., Jessen, E. (Hrsg.) E-Science und Grid, Ad-hoc-Netze, Medienintegration, LNI-Proceedings Volume P-55 zur 18. DFN-Arbeitstagung über Kommunikationsnetze, 2.-4.6.2004, Düsseldorf, S. 451–465 (2004)
- Louis, D.: Visual C# 2010. Das komplette Starterkit für den erfolgreichen Einstieg. Markt + Technik, München (2010)
- Ludewig, J., Lichter, H.: Software Engineering. Grundlagen, Menschen, Prozesse, Techniken. dpunkt, Heidelberg (2010)
- Mark, D., Nutting, J., LaMarche, J.: Beginning Iphone 4 Development: Exploring the IOS SDK. Apress, New York (2011)
- Maske, P.: Mobile Applikationen 1: Interdisziplinäre Entwicklung am Beispiel des Mobile Learning. Gabler, Wiesbaden (2012)
- Meier, A., Stormer, H.: eBusiness & eCommerce. Management der digitalen Wertschöpfungskette, 2. Aufl. Springer, Berlin (2008)
- Meier, R.: Generierung von Kundenwert durch mobile Dienste. Potenziale durch Kommunikation und Vernetzung. Deutscher Universitäts-Verlag, München (2002)
- Mertens, P., Bodendorf, F., König, W., et al.: Grundzüge der Wirtschaftsinformatik, 9. Aufl. Verlag, Berlin (2005)
- Microsoft Corp.: Datentyp: Zusammenfassung (Visual Basic). <http://msdn.microsoft.com/de-de/library/47zceaw7%28v=VS.90%29.aspx> (2007). Zugegriffen 23. Sept. 2013
- Microsoft Corp.: Microsoft windows phone 8 features. <http://www.windowsphone.com/de-de/features> (2013a). Zugegriffen 25. Sept. 2013
- Microsoft Corp.: .Net Framework 4. <http://msdn.microsoft.com/de-de/netframework/default.aspx> (2013b). Zugegriffen 24. Sept. 2013
- Microsoft Corp.: Microsoft visual studio 2010 professional testversion. <http://www.microsoft.com/de-de/download/details.aspx?id=16057> (2013c). Zugegriffen 24. Sept. 2013
- Müller-Wilken, S.: Mobile Geräte in verteilten Anwendungsumgebungen: ein Integrationsansatz zwischen Abstraktion und Migration, Fachbereich Informatik. Universität Hamburg, Hamburg (2002)
- Muscaino, C., Kennedy, B.: HTML & XHTML. Das umfassende Referenzwerk, 4. Aufl. O'Reilly, Köln (2003)
- Naur, P., Randell, B.: Software Engineering. Report on a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7th to 11th October 1968, Scientific Affairs Division, NATO, Brussels, S. 138–155 (1969)
- Pomberger, G., Dobler, H.: Algorithmen und Datenstrukturen. Eine systematische Einführung in die Programmierung. Addison-Wesley, München (2008)
- Pomberger, G., Pree, W.: Software Engineering. Architektur-Design und Prozessorientierung, 3. Aufl. Carl Hanser, München (2004)
- Pree, W.: Mobiles Rechnen. In: Rechenberg, P., Pomberger, G. (Hrsg.) Informatik-Handbuch, 4. Aufl., S. 1135–1149. Carl Hanser, München (2006)
- Reichenwald, R., Meier, R., Freimuth, N.: Die mobile Ökonomie – Definition und Spezifika. In: Reichenwald, R. (Hrsg.) Mobile Kommunikation. Wertschöpfung, Technologien, neue Dienste, S. 3–19. Gabler, Wiesbaden (2002)
- Roth, J.: Ein Anwendungsrahmenwerk für synchrone kollaborative Anwendungen in mobilen Umgebungen, Fern-Universität Hagen, Fachbereich Informatik, Informatik-Bericht Nr. 292, Hagen (2002)

- Roth, J.: Mobile Computing: Grundlagen, Technik, Konzepte, 2. Aufl. d.punkt-Verlag, Heidelberg (2005)
- Rügge, I.: Mobile Solutions: Einsatzpotenziale, Nutzungsprobleme und Lösungsansätze. Deutscher Universitäts-Verlag, Wiesbaden (2007)
- Satyanarayanan, M.: Fundamental challenges in mobile computing. In: ACM (Hrsg.) Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing, S. 1–7. ACM Press, USA (1996)
- Scheller, U.: Native- oder Web-Anwendungen, wohin geht die Reise? mobile Zeitgeist 1/11, 23–25 (2011) (Hamburg)
- Scholze-Stubenknecht, W., Wermke, M., Klosa, A., Drosdowski, G.: Duden – Die deutsche Rechtschreibung: auf der Grundlage der neuen amtlichen Rechtschreibregeln, 22. Aufl. Dudenverlag, Mannheim (2000)
- Schuhmann, M.: Betriebswirtschaftliche und technologische Grundlagen von E-Commerce und M-Commerce. In: Keuper, F. (Hrsg.) Electronic Business und Mobile Business. Konzepte, Ansätze und Geschäftsmodelle, S. 3–25. Gabler, Wiesbaden (2002)
- Sommerville, I.: Software Engineering, 6. Aufl. Pearson Studium, München (2001)
- Stahlknecht, P., Hasenkamp, U.: Einführung in die Wirtschaftsinformatik, 11. Aufl. Springer, Heidelberg (2005)
- Tschersich, M.: Was ist ein mobiles Endgerät? Hamburg. <http://www.mobile-zeitgeist.com/2010/03/09/was-ist-ein-mobiles-endgeraet/> (2010). Zugegriffen 23. Sept. 2013
- Turowski, K., Pousttchi, K.: Mobile Commerce. Grundlagen und Techniken, Springer, Berlin (2004)
- Victor, F.: Programmiersprachen. In: Schneider, U., Werner, D. (Hrsg.) Taschenbuch der Informatik, 6. Aufl., S. 197–220. Carl Hanser, München (2007)

---

# Mit Struktur und Methode in die projektindividuelle App-Entwicklung

5

Marius Schönberger und Christian Aichele

*Praktische Ansätze zur zielorientierten Anwendung von Software Engineering*

---

## Zusammenfassung

„Die App-Entwicklung ist eine gelungene Mischung aus grundsätzlichem Projektmanagement und kreativer Vorgehensweise“

Kaum ein modernes Unternehmen kommt aktuell ohne die Verbreitung zusätzlichen Mehrwertes zu einer langfristigen Sicherung des Erfolgs seiner Produkte. Im Trend liegt insbesondere die Bereitstellung zusätzlicher Softwareanwendungen für die mobilen Endgeräte der Kunden, sogenannte mobile Applikationen (Apps). Gegenwärtig kann über diverse Onlinemarkte auf eine Vielzahl verfügbarer mobiler Anwendungen zugegriffen werden, die zur Unterstützung fast jeder alltäglichen Situation verschiedene Lösungen anbieten. Dies stellt eine große Herausforderung an die Entwickler dar, da nicht nur die Verbesserung und Weiterentwicklung bestehender, sondern auch die Programmierung neuer individueller Applikationen von der breiten Masse gefordert wird. Die Entwicklung mobiler Anwendungen verlangt somit ein grundlegendes Verständnis über die Anwendung von Projektmanagementmethoden und Vorgehensweisen des

---

M. Schönberger (✉)  
Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: marius.schoenberger@fh-kl.de

C. Aichele  
Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: christian.aichele@fh-kl.de

Software Engineering. In Bezug auf die Generierung innovativer Applikationen wird weiterhin ein hohes Maß an Kreativität gefordert.

Im vorliegenden Kapitel werden für die erwähnten Herausforderungen bei der Entwicklung mobiler Anwendungen praktische Ansätze für die zielorientierte Anforderung des Software Engineering gegeben. Hierzu wird zunächst ein Vorgehensmodell für die Entwicklung und Vermarktung mobiler Applikationen aufgestellt sowie weitere in Bezug auf die Softwareherstellung etablierte Vorgehens- und Entwicklungsmodelle genannt. Die nachfolgenden Kapitel richten sich an dem zuvor angeführten Vorgehensmodells zur mobilen Anwendungsentwicklung aus und zeigen notwendige Aktivitäten innerhalb der jeweiligen Phasen des Vorgehensmodells auf. Darüber hinaus werden Werkzeuge und Methoden zur Unterstützung der jeweiligen Phasen aufgezeigt und deren Anwendung innerhalb des Softwareentwicklungsprozesses beschrieben.

An dieser Stelle möchten sich die Autoren bei Herrn William Motsch bedanken, der sie in Bezug auf die Aufstellung des Vorgehensmodells zur mobilen Anwendungsentwicklung sowie bei den Ausführungen zur Planungs- und Konzeptphase mit hilfreichen Diskussionen und Anregungen unterstützte.

---

## 5.1 Der Einsatz des Software Engineering in der mobilen Anwendungsentwicklung

In der Softwareentwicklung existieren eine Vielzahl von Modellen und Vorgehensweisen, die teils historisch betrachtet, teils aus wachsenden Anforderungen an die IT-Branche stetig weiterentwickelt und gestaltet wurden. Von dem klassischen Wasserfallmodell bis hin zu agilen Vorgehensmodellen, wie z. B. SCRUM, werden Modellansätze für die Softwareentwicklung vorgeschlagen, die zudem auch in Grundzügen Empfehlungen für ein entsprechendes Projektmanagement aussprechen. In Hinblick auf die Entwicklung mobiler Applikationen stellt sich grundsätzlich die Frage, welche Modelle und Entwicklungsformen für eine erfolgreiche Entwicklung zu empfehlen sind. Bezüglich der großen Ideenvielfalt und der Neuartigkeit vieler mobiler Applikationen sind im Generellen die Grundzüge des Projektmanagements zu berücksichtigen.

Aufgrund der Möglichkeiten, eigene Ideen schnell umsetzen zu können sowie auch Applikationen für spezielle Anwendungsfälle über Marktplätze im Internet bereitzustellen, wird eine schnelle Verbreitung mobiler Anwendungen gefördert. Als Besonderheit der mobilen Applikationsentwicklung können je nach Anwendungsfall eine relativ kurze Entwicklungszeit und eine dynamische Ressourcenplanung charakteristisch sein. So unterschiedlich die Ausgangssituation für die Entwicklung sowie die Idee und den Anwendungsfall mobiler Anwendungen aussieht, so verschieden können die Herangehensweisen und Entwicklungsmodelle gestaltet werden. Nachfolgend werden verschiedene Ansätze betrachtet, die insbesondere auf die sehr schnelle „Time-to-Market“ von mobilen Applikationen ausgerichtet sind.

**Abb. 5.1** Vorgehensmodell zur mobilen Anwendungsentwicklung

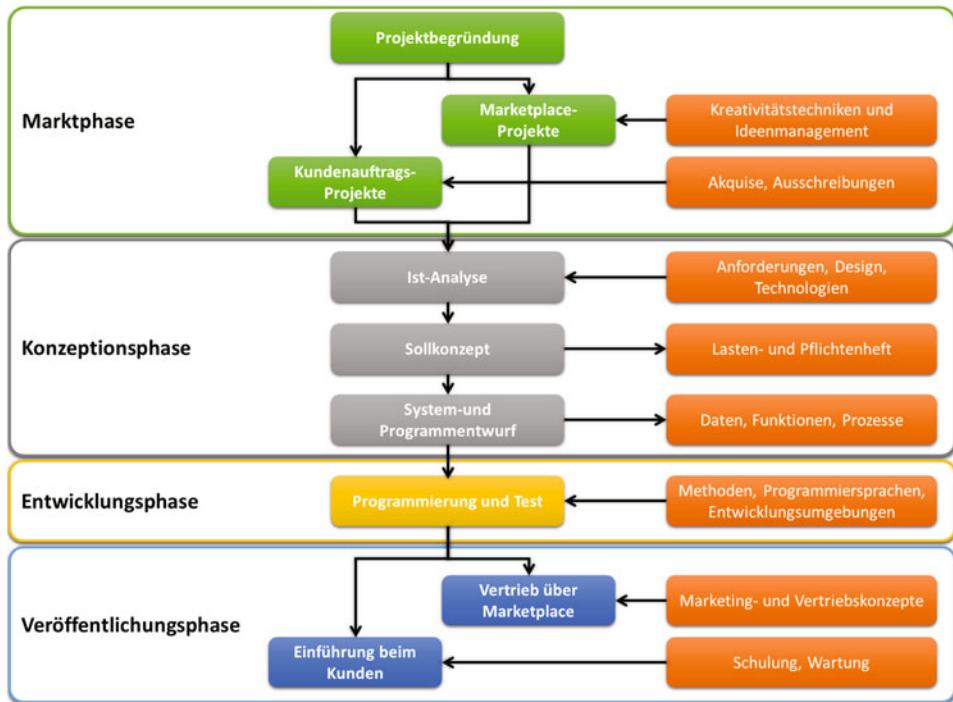


Aufgrund der in der Literatur vielfach sehr umfassenden und komplizierten Modelle zur Softwareentwicklung liegt der Fokus bei der Beschreibung der nachfolgenden Modelle und Phasen auf einer guten und unkomplizierten Durchdringung der themenbezogenen Inhalte anhand eines auf die mobile Anwendungsentwicklung ausgerichteten Phasenmodells.

Die thematischen Inhalte des Phasenmodells folgen aktuellen Methoden und Ansätzen im Software Engineering, sodass eine schrittweise Anwendung der einzelnen Themengebiete ermöglicht wird. Zeitgleich bleibt jedoch der Gesamtprozess weiterhin im Betrachtungsfokus des Phasenmodells.

Die in der Praxis auftretende modulare Spezialisierung von einzelnen Teilaufgaben, bspw. bei der Konzepterstellung oder der Programmierung, ermöglicht die Darstellung einzelner logischer Phasen im Modell sowie eine höhere Flexibilität in Hinblick auf die Durchführung und Einordnung im Gesamtprozess.

Die Entscheidung für eine geeignete Vorgehensweise zur Entwicklung von mobilen Anwendungen umfasst im Vorfeld die Betrachtung der zugrunde liegenden Eigenschaften der Ausgangssituation. Dies ist bereits im Vorfeld der eigentlichen Entwicklung von wesentlicher Bedeutung, da sich insbesondere in Bezug auf zeitliche und finanzielle Prämissen das Vorgehen einer auftragsbezogenen Kundenanforderung von einer ideengetriebenen Entwicklung unterscheiden kann. Die Formulierung konkreter Anforderungen spielt vor dem Beginn der Entwicklung ebenfalls eine zentrale Rolle, um Ziel und Zweck der mobilen Applikation zu definieren, die zu nutzende Technologie und das Betriebssystem festzulegen sowie Wünsche hinsichtlich des Designs aufzunehmen. Um die Anforderungen in den Entwicklungsprozess umzusetzen, ist es von hoher Bedeutung, dass die benötigten Ressourcen, in Form von geeignetem Personal und technischer Ausstattung, zur Verfügung stehen. Die Zusammenfassung dieser Aspekte sollte im Zuge der Analyse der



**Abb. 5.2** Aktivitäten und Ergebnisse des Vorgehensmodells zur App-Entwicklung. (Eigene Erstellung, in Anlehnung an Stahlknecht und Hasenkamp 2005, S. 218)

Ausgangssituation erfolgen, sodass marktbezogene und anforderungsspezifische Elemente sowie die vorhandenen und benötigten Ressourcen berücksichtigt werden. Die Auswahl von geeigneten Vorgehensmodellen resultiert aus der vorhergehenden Einschätzung der Ausgangssituation.

In Abb. 5.1 werden die einzelnen Projektphasen bei der App-Entwicklung in einem gemeinsamen Rahmen verdeutlicht. Nachfolgend werden die Aktivitäten innerhalb der einzelnen Projektphasen aufgezeigt sowie Initiierungsergebnisse und Ergebnisse der jeweiligen Phasen veranschaulicht (vgl. Abb. 5.2).

Die Aktivitäten zu den einzelnen Phasen im Vorgehensmodell werden in den nachfolgenden Kapiteln beschrieben. Hierbei erfolgt insbesondere die Fokussierung auf das Themengebiet der App-Entwicklung. Die Ergebnisse aus den vorhergehenden Phasen sind für die weitere Anwendung des Phasenmodells von hoher Bedeutung, sodass eine Empfehlung für ein Vorgehensmodell nur dann festgelegt werden sollte, wenn die vorhergehenden Phasen erfolgreich durchlaufen wurden. Je nach Ausgangssituation kann die Situation eintreten, dass eine andere Vorgehensweise zu wählen ist. Die Merkmalsausprägungen in den Elementen im Phasenmodell hängen eng zusammen, sodass eine grundlegende Analyse der Ausgangssituation für die Entscheidung eines geeigneten Modells für die Program-

mierung von mobilen Anwendungen zu Beginn zwingend notwendig ist. Ebenfalls ist festzuhalten, dass keine allgemeingültige Empfehlung ausgesprochen werden kann, die für jede Situation den größtmöglichen Nutzen bietet. Der modellbezogene Ansatz bietet eine gute Referenz, deren Spezifikation in der Praxis situativ ausgeprägt werden kann.

---

## 5.2 Vorgehens- und Entwicklungsmodelle

Für die Entwicklung von Software ist nicht nur ein allgemeines Ingenieur- oder Informatikwissen, sondern auch ein breites Projektmanagementwissen erforderlich. Wesentliches Element aus Sicht eines Projektleiters ist es, eine Software innerhalb des Kostenbudgets und der Termine sowie in der gewünschten Qualität zu liefern. In diesem Zusammenhang müssen Softwareentwickler grundlegende Erfahrungen mit der Erstellung von Zeitplänen oder der Auswahl geeigneter Vorgehensmodelle aufweisen. Besonders die Auswahl von Vorgehens- oder Entwicklungsmodellen ist zum einen entscheidend für den Erfolg der Software und zum anderen für das Einhalten terminlicher Vorgaben und Rahmenbedingungen.

In diesem Zuge stehen vielschichtige fachliche und menschliche Herausforderungen, wie z. B. das Verständnis von Stakeholdern untereinander, eine sehr große Anzahl von Arbeitspaketen mit verschachtelten Abhängigkeiten, aber auch die Kommunikation zwischen Auftraggeber und Auftragnehmer. Je nach der strategischen Bedeutung solcher Projekte sind die Risiken von Kosten und Terminüberschreitungen enorm hoch. Frustrierende Projektverläufe und Projektabbruch sind hierbei die Folge.<sup>1</sup>

Um die Erfahrungen solcher Projekte zu nutzen und somit eine Fehlervermeidung herbeizuführen, helfen Vorgehensmodelle dabei, die Erfahrungen anderer für eigene Projekte zu nutzen. Hierbei vereinfacht ein gemeinsames Prozessverständnis die Zusammenarbeit und Austauschbarkeit der Projektmitarbeiter. Die Zuständigkeiten für die zu liefernden Ergebnisse können durch die Anwendung von Vorgehensmodellen klar geregelt werden, sodass im Zuge der Planung wichtige Arbeitsschritte berücksichtigt werden können und das Projekt besser strukturiert werden kann. Anhand dieses Vorgehens können Projekte einfacher miteinander verglichen werden, sodass eine höhere Prozessqualität auch zu besseren Ergebnissen führt, wobei durch ein bestimmtes Vorgehensmodell nicht gleichermaßen alle Projekte abgedeckt werden können.<sup>2</sup>

Die Wahl des Vorgehensmodells hängt hierbei mit dem Gleichheitsgrad der Projekte zusammen, sodass bei einem hohen Gleichheitsgrad eine vorausschauende Planung mit einem wasserfallorientierten Vorgehen geeignet ist, während bei vielen Projektunbekannten oder Forschungsprojekten eine anpassbare Planung mit agilen Vorgehensmodellen

---

<sup>1</sup> Vgl. Tremp und Ruggiero 2011, S. 10.

<sup>2</sup> Vgl. Tremp und Ruggiero 2011, S. 10.

zielführender ist. Die Vorgehensmodelle zur Softwareentwicklung und die Methoden des Software Engineering sind anders als die allgemeinen Projektmanagementmethoden stark auf die Aspekte der Softwareentwicklung fokussiert, sodass mit diesen Modellen nicht immer auch generelle Projektmanagementempfehlungen ausgesprochen werden können.<sup>3</sup> In Hinblick auf die Bedeutung und die Bewertung einzelner Vorgehensmodelle für die Softwareentwicklung werden in der Literatur verschiedene Instrumentarien zur Verfügung gestellt. Je nach Anwendungsfall, Projektgröße und Anforderungen an die Flexibilität, können unterschiedliche Modelle genutzt werden.<sup>4</sup>

Die Besonderheiten der Vorgehensmodelle im Software Engineering sowie in Bezug auf die Entwicklung mobiler Anwendungen sollen in den nachfolgenden Kapiteln dargestellt werden. In einem ersten Schritt wird zunächst der Begriff Vorgehensmodell definiert und ein allgemeines Ordnungsschema von Vorgehensmodellen dargestellt. Daran anknüpfend werden einige Vorgehensmodelle genannt und kurz erläutert. Das Kapitel endet mit dem Aufzeigen von Bewertungsverfahren zur Modellauswahl.

### **5.2.1 Definition und Einordnung von Vorgehens- und Entwicklungsmodellen**

Vorgehens- und Entwicklungsmodelle stellen Werkzeuge der Softwareentwicklung dar, die zur Planung, Durchführung und Kontrolle von informationstechnischen Projekten (IT-Projekten) eingesetzt werden. Vorgehensmodelle bestehen aus mehreren aufbauenden Phasen, welche in einer festen Reihenfolge hintereinander geschaltet sind. Aus den zahlreichen Ausprägungen unterschiedlicher IT-Projekte resultiert, dass nicht nur ein Vorgehensmodell zur Bearbeitung verschiedener Aufgabenstellungen existieren kann. Im Bereich der Softwareentwicklung können je nach Softwaresystem unterschiedliche Vorgehens- und Entwicklungsmodelle zum Einsatz kommen.<sup>5</sup> In der Literatur finden sich mehrere Definitionen zu Vorgehens- und Entwicklungsmodellen. Die Autoren Stahlknecht und Hasenkamp geben folgende Begriffsdefinition:<sup>6</sup>

► **Vorgehensmodell** Allgemein beschreibt jedes Vorgehensmodell die Folge aller Aktivitäten, die zur Durchführung eines Projektes erforderlich sind. Vorgehensmodelle für die Systementwicklung von Anwendungssystemen geben an, wie die Prinzipien, Verfahren und Werkzeuge der Software-Entwicklung [...] einzusetzen sind.

---

<sup>3</sup> Vgl. Tremp und Ruggiero 2011, S. 10.

<sup>4</sup> Koord und Krauter 2009.

<sup>5</sup> Vgl. Wieczorek und Mertens 2011, S. 66.

<sup>6</sup> Stahlknecht und Hasenkamp 2005, S. 219.

Eine weitere Definition wird von den Autoren Horn, Kerner und Forbig gegeben, welche den Begriff Prozessmodell synonym zu dem Begriff Vorgehensmodell verwenden:<sup>7</sup>

- Ein **Vorgehensmodell** oder auch Prozessmodell regelt die gesamte Softwareentwicklung durch die einheitliche und verbindliche Vorgabe von Phasen, Aktivitäten und Produkten, die dabei und bei den begleitenden Tätigkeiten anfallen. Die Gesamtentwicklung unterteilt sich in Phasen, in denen verschiedene Aktivitäten ausgeführt werden müssen. [...].

Die Definitionen von Balzert<sup>8</sup> oder von Brandt-Pook und Kollmeier<sup>9</sup> weisen ähnliche Aussagen auf, sodass folgende allgemeingültige Begriffsbestimmung in Bezug auf die Entwicklung von Anwendungssoftware gegeben werden kann:

- Ein **Vorgehens- oder Entwicklungsmodell** stellt einen Entwicklungsplan zur Konzeption, Herstellung und Wartung von Softwareprodukten oder -systemen dar, welcher durch standardisierte Phasen, Aktivitäten und Werkzeuge das generelle Vorgehen der Softwareentwicklung festlegt.

Der Einsatz von Vorgehensmodellen findet unter anderem im Bereich der Politik, der Wirtschaft und des Militärs Anwendung, wird aber innerhalb der nachfolgenden Ausführungen nur in Bezug auf den Einsatz in der Softwareentwicklung näher betrachtet.

Aufgrund der steigenden Komplexität zu entwickelnder Software werden im Bereich der Softwareentwicklung immer häufiger spezialisierte Vorgehensmodelle gefordert, die Entwicklungsrisiken überschaubarer und den Entwicklungsstand transparenter gestalten sollen. Diese Forderung führte in den letzten Jahren zu einem starken und unübersichtlichen Wachstum an vorhandenen Vorgehensmodellen.<sup>10</sup> Die Fachgruppe „Vorgehensmodelle für die betriebliche Anwendungsentwicklung“ der Gesellschaft für Informatik hat zur Klärung der genannten Problematik ein allgemeines Schema zur Einordnung und Definition von Vorgehensmodellen vorgegeben.<sup>11</sup> Das Schema wird in Abb. 5.3 dargestellt.

Grundlegend ist jedes Vorgehensmodell identisch aufgebaut. Neben der Abgrenzung der einzelnen Phasen voneinander, durch bspw. definierte Meilensteine oder Entscheidungspunkte, erfolgt die Umwandlung eines vorläufigen Konzeptes in ein lauffähiges Softwareprodukt.<sup>12</sup> In der Literatur wird somit keine Unterscheidung der Vorgehensmodelle nach Ergebnissen oder internen Prozessabläufen vorgenommen. Vielmehr erfolgt

<sup>7</sup> Horn et al. 2003, S. 312.

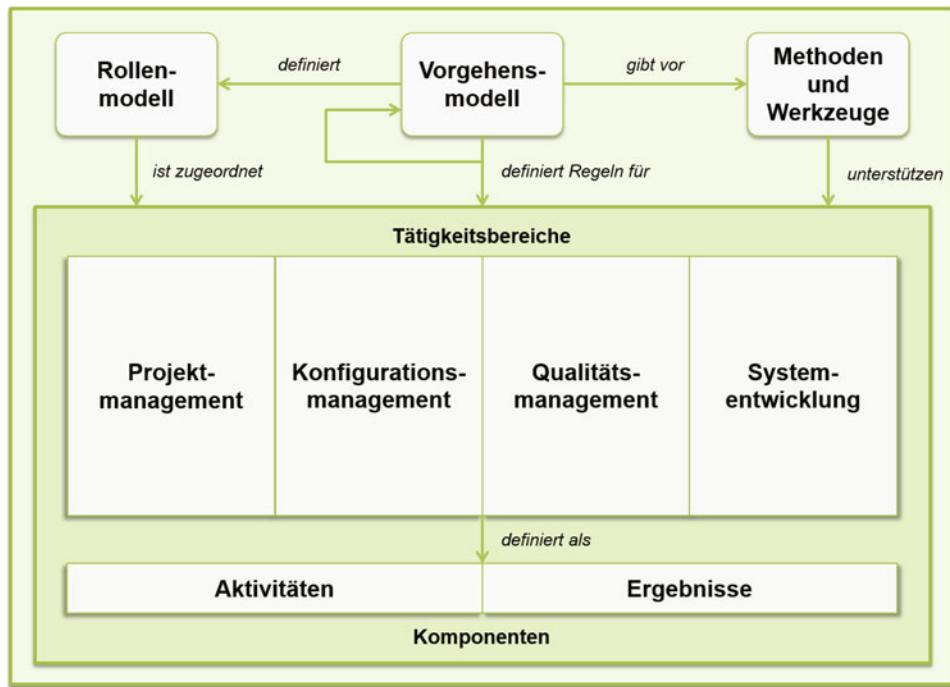
<sup>8</sup> Vgl. Balzert 2009, S. 598.

<sup>9</sup> Vgl. Brandt-Pook und Kollmeier 2008, S. 3.

<sup>10</sup> Vgl. Biskup und Fischer 2003, S. 3 f.

<sup>11</sup> Vgl. Biskup und Fischer 2003, S. 3.

<sup>12</sup> Vgl. Krcmar 2005, S. 148.



**Abb. 5.3** Ordnungsschema von Vorgehensmodellen. (Eigene Erstellung, in Anlehnung an Biskup und Fischer 2003, S. 3)

die Differenzierung nach klassischen, modernen und agilen Vorgehensmodellen, die nachfolgend näher erläutert werden.

### 5.2.2 Klassische Vorgehensmodelle

Zur Familie der klassischen Vorgehensmodelle, auch sequentielle Vorgehensmodelle oder Phasenmodelle genannt, werden Wasserfall- und Schleifenmodelle zusammengefasst. Jedes Modell dieser Familie gliedert das zu bearbeitende Projekt in sequenziell hintereinander ablaufende Phasen. Phasenmodelle in verschiedenen Variationen beschreiben im Wesentlichen folgende Phasen in den Softwareentwicklungsprozessen:<sup>13</sup>

- Problemanalyse und Grobplanung
- Systemspezifikation und Planung
- System- und Komponentenentwurf
- Implementierung und Komponententest

<sup>13</sup> Vgl. Pomberger und Pree 2004, S. 11.

- System- und Integrationstest
- Betrieb und Wartung

Die Vorgehensweise bei der phasenorientierten Softwareentwicklung basiert auf dem Top-down-Ansatz einzelner gekapselter Schritte im Vorgehensmodell. In diesem Kontext erfolgt eine schrittweise Konkretisierung. Als Grundlage für die einzelnen Phasen werden konkret definierte Produkte, häufig in Form von Dokumenten, benötigt. Diese werden in den phasenbezogenen Prozessen verarbeitet, in deren Rahmen bestimmte Methoden und Werkzeuge entsprechende Anwendung finden, sodass deren Resultate an die nächste Phase weitergeleitet werden. Die Voraussetzung zum Übergang in eine nächste Phase bildet der Abschluss der gegenwärtigen Phase. Die erfolgreiche Beendigung einer Phase wird durch die Erstellung von Dokumenten und Fortschrittsberichten, z. B. in Form von Anforderungs- oder Entwurfsdokumenten, gekennzeichnet. Rücksprünge in vorherige Phasen sind nur an zuvor definierten Zeitpunkten oder Projektabschnitten zulässig. Der Einsatz dieser streng sequenziellen Vorgehensweise verfolgt das Ziel, dass Softwareprojekte besser geplant, organisiert und kontrolliert werden können.<sup>14</sup>

Das Wasserfallmodell zählt zu einem der ersten Vorgehensmodelle und wurde im Laufe der Zeit an aktuelle Softwareanforderungen angepasst. Diese Spezialisierungen des Modells führen dazu, dass die Anzahl der inhärenten Phasen sowie deren Einteilung voneinander abweichen. Abbildung 5.4 zeigt das Wasserfallmodell nach Aichele, welches eine Weiterentwicklung des ursprünglichen Wasserfallmodells darstellt. Das Modell beinhaltet insgesamt acht Phasen, die jeweils mit einer Qualitätskontrolle abschließen.<sup>15</sup> Ist diese Qualitätsprüfung nicht zufriedenstellend, sieht das Modell entweder ein erneutes Durchlaufen der Phase oder einen Rücksprung in eine vorherige Phase vor. Werden durch die Validierung keine erheblichen Qualitätsmängeln festgestellt, erfolgt auf Basis der Arbeitsergebnisse der Übergang in die nächste Phase.<sup>16</sup>

Eine Verfeinerung des Wasserfallmodells stellt das Spiralmodell dar, welches erstmals im Jahre 1988 durch Barry W. Boehm vorgestellt wurde.<sup>17</sup> Beim Spiralmodell handelt es sich um ein evolutionäres Modell, dass einerseits den Gesamtaufwand des Projektes und andererseits den Projektfortschritt in den einzelnen Spiralzyklen darstellt.<sup>18</sup> Die Entwicklung eines Softwareproduktes mittels des Spiralmodells sieht das Durchlaufen der Phasen Entwicklung, Integration, Qualitätssicherung und Planung vor.<sup>19</sup> Jede Phase enthält mehrere Spiralwindungen, die weitere unterschiedliche Arbeitspakete enthalten. Ähnlich dem Wasserfallmodell ist jeder Zyklus innerhalb einer Phase mit einem Validierungsschritt versehen. Dadurch wird gewährleistet, dass die Entwickler frühzeitig Fehler entdecken

<sup>14</sup> Vgl. Biskup und Fischer 2003, S. 4 f. sowie Pomberger und Pree 2004, S. 11 f.

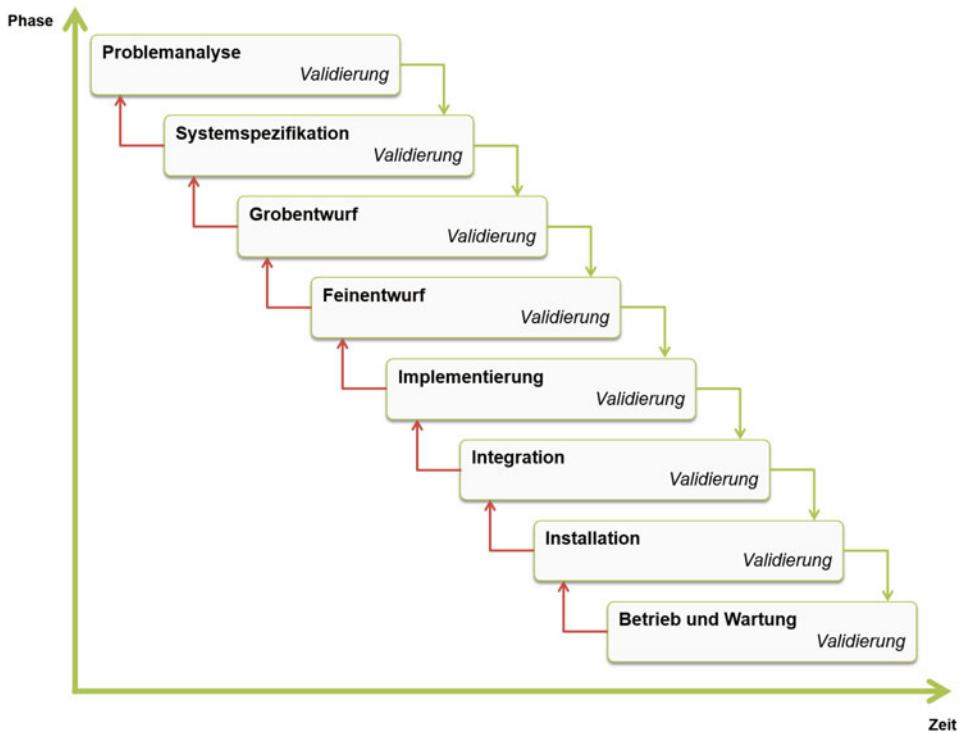
<sup>15</sup> Vgl. Aichele 2006, S. 47 f.

<sup>16</sup> Vgl. Biskup und Fischer 2003, S. 6.

<sup>17</sup> Vgl. Boehm 1988, S. 61 ff.

<sup>18</sup> Vgl. Aichele 2006, S. 47.

<sup>19</sup> Vgl. Horn et al. 2003, S. 314.



**Abb. 5.4** Wasserfallmodell zur Softwareentwicklung. (Eigene Erstellung, in Anlehnung an Aichele 2006, S. 48)

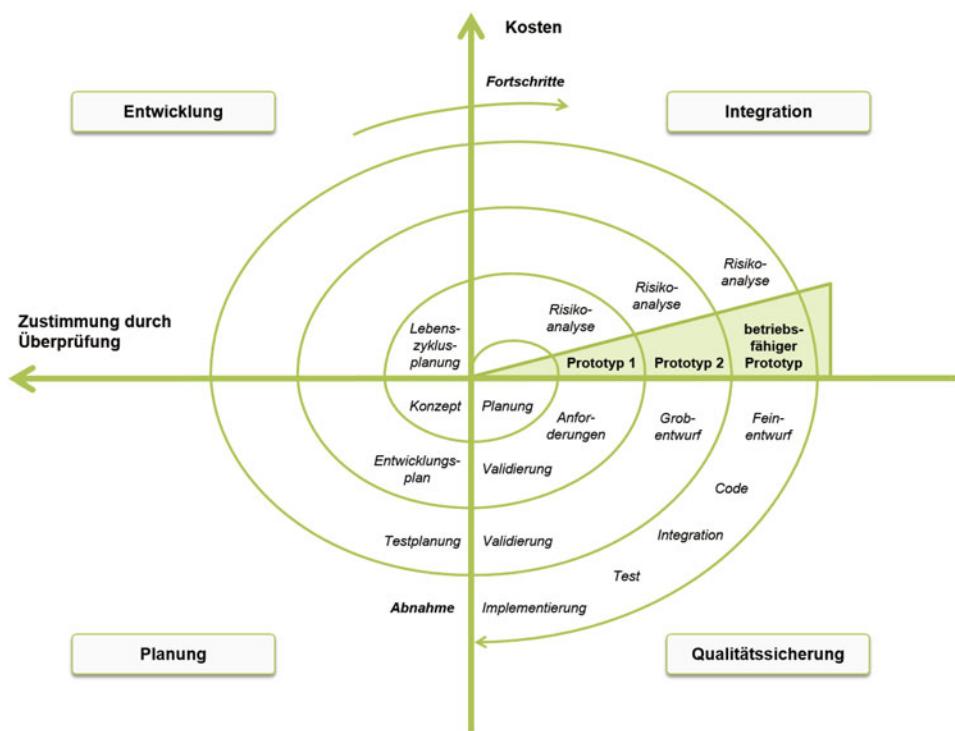
und beseitigen können. Eine weitere Aufgabe der Validierung besteht darin, den nächsten Zyklus zu planen und notwendige Ressourcen festzulegen.<sup>20</sup> Der Aufbau des Spiralmodells wird in Abb. 5.5 dargestellt.

Eine Neuerung zum Wasserfallmodell besteht in der Aufnahme der Prototypenentwicklung sowie der Risikoanalyse. Dadurch besteht für Entwickler die Möglichkeit, den Endanwendern des Softwareproduktes bereits in einem frühen Projektstadium erste lauffähige Ergebnisse zu präsentieren.<sup>21</sup> Neben der daraus resultierenden Evaluierung von Alternativen können weiterhin potenzielle Risiken identifiziert und reduziert werden.<sup>22</sup>

<sup>20</sup> Vgl. Aichele 2006, S. 47.

<sup>21</sup> Vgl. Schmidt 1999, S. 61 f.

<sup>22</sup> Vgl. Aichele 2006, S. 48.



**Abb. 5.5** Spiralmodell zur Softwareentwicklung. (Eigene Erstellung, in Anlehnung an Aichele 2006, S. 48)

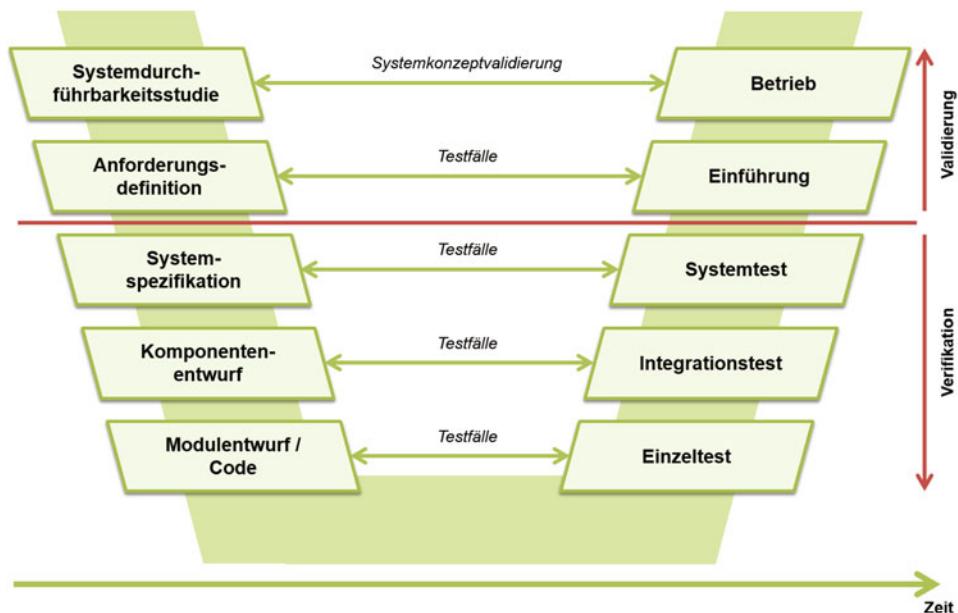
### 5.2.3 Moderne Vorgehensmodelle

Der Übergang von klassischen zu modernen Vorgehensmodellen wird durch eine höhere Flexibilität gegenüber auftretenden Änderungen, die bessere Transparenz der Projekte und Ergebnisse sowie die verstärkte Einbindung des Endnutzers in den gesamten Projektverlauf geprägt. Als eines der ersten modernen Vorgehensmodelle ist in diesem Zusammenhang das V-Modell zu nennen, welches ursprünglich für das Deutsche Bundesministerium für Verteidigung (BMVg) und das Bundesamt für Wehrtechnik und Beschaffung (BWB) erstellt wurde.<sup>23</sup> Neben dem Einsatz in Bundeseinrichtungen konnte das V-Modell, durch die organisationsneutrale Konzeption des Modells, weiterhin in Unternehmen verschiedener Branchen, z. B. Banken oder Versicherungen, zur Entwicklung von Informations- oder Softwaresystemen integriert werden.<sup>24</sup>

Geprägt wird das V-Modell durch eine V-ähnliche Architektur, welche auf die Begriffe Validierung und Verifikation zurückzuführen ist. In diesem Zusammenhang bedeutet

<sup>23</sup> Vgl. Aichele 2006, S. 49.

<sup>24</sup> Vgl. Versteegen 2001, S. 1.



**Abb. 5.6** V-Modell zur Softwareentwicklung. (Eigene Erstellung, in Anlehnung an Krcmar 2010, S. 150)

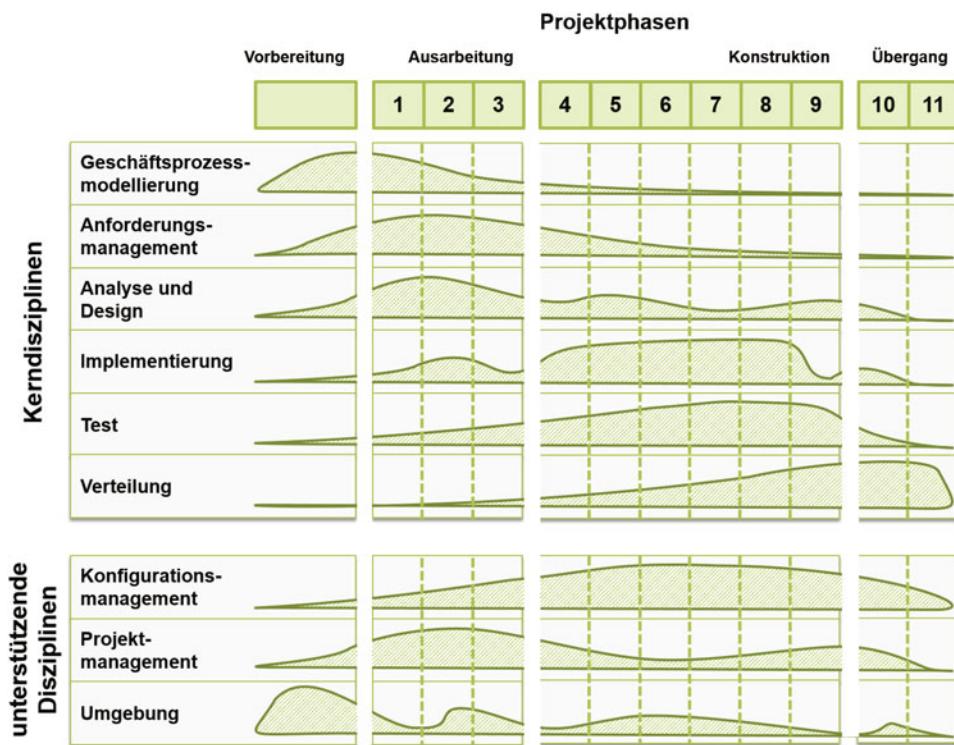
Validierung die Sicherstellung der Angemessenheit des Systems an die Problemstellung. Im Zuge der Verifikation wird die Übereinstimmung zwischen einem Softwareprodukt und seiner Spezifikation überprüft.<sup>25</sup> Für die Durchführung dieser Aufgaben sieht das V-Modell den Einsatz von Testfällen vor, welche einerseits zur Überprüfung der Korrektheit der Software und andererseits zur Bestimmung des Produktwertes eingesetzt werden. In Absprache mit den Endnutzern werden Daten und Informationen aus dem zukünftigen Einsatzgebiet der Software bereitgestellt und zu realistischen Anwendungsszenarien zusammengeführt.<sup>26</sup> Der Softwareentwicklungsprozess wird innerhalb des V-Modells als eine Abfolge von Aktivitäten beschrieben, welche mit einem definierten Endergebnis abgeschlossen werden. Diese Aktivitäten beziehen sich nicht ausschließlich auf die Softwareentwicklung, sondern auch Projektmanagement, Konfigurationsverwaltung und Qualitätssicherung werden betrachtet und jeweils in die Ebenen Vorgehensweise, Methode und Werkzeuganforderung untergliedert.<sup>27</sup> Abbildung 5.6 zeigt die Architektur des V-Modells in Anlehnung an Krcmar.

Ein weiteres modernes Vorgehensmodell ist das Rational-Unified-Process-Modell (RUP). Das Modell basiert auf dem Wasserfall- und Spiralmodell und vereinigt somit

<sup>25</sup> Vgl. Krcmar 2010, S. 150.

<sup>26</sup> Vgl. Balzert 2009, S. 445.

<sup>27</sup> Vgl. Krcmar 2010, S. 150 f.



**Abb. 5.7** RUP-Modell zur Softwareentwicklung. (Eigene Erstellung, in Anlehnung an Kruchten 2004, S. 22)

sequentielle als auch evolutionäre Eigenschaften und Vorgehensweisen. Seit der Einführung im Jahre 1999 durch die Firma IBM entwickelte sich das RUP-Modell zu einem De-facto-Standard, der folgende sechs Best Practices verfolgt:<sup>28</sup>

- Software iterativ entwickeln.
- Anforderungen managen.
- Komponentenbasierte Architekturen verwenden.
- Software visuell modellieren.
- Softwarequalität kontinuierlich prüfen.
- Änderungen kontrolliert in die Software einbringen.

Die dem RUP-Modell zugrunde liegende Architektur soll anhand von Abb. 5.7 verdeutlicht werden. Aus dieser Abbildung zeigt sich, dass im Modell zwischen Projektphasen, auch als dynamische Dimension bezeichnet, und Disziplinen, auch als statische Dimension bezeichnet, unterschieden wird. Auf der vertikalen Achse des RUP-Modells erfolgt die

<sup>28</sup> Vgl. Grechenig et al. 2010, S. 379.

Abbildung der Kerndisziplinen und unterstützenden Disziplinen.<sup>29</sup> Aufbauend auf den Entwicklungsphasen des Spiralmodells (vgl. Abb. 5.5) umfasst das RUP-Modell die Kerndisziplinen Anforderungsmanagement, Analyse und Design, Implementierung und Test, welche durch die Geschäftsprozessmodellierung und Verteilung erweitert werden. Zudem erfolgt weiterhin die Unterscheidung der unterstützenden Disziplinen in Konfigurations- und Projektmanagement sowie in Entwicklungsumgebung, welche sich durch alle Schritte des Softwareentstehungsprozesses hindurchziehen. Der Aufwand jeder einzelnen Disziplin wird durch die Darstellung einer Kurve verdeutlicht. Je höher die Kurve eines einzelnen Prozessschrittes ist, desto höher ist der Aufwand innerhalb einer bestimmten Projektphase<sup>30</sup> (in Abb. 5.7 durch die Zahlen eins bis elf gekennzeichnet).

Die dynamische Dimension, die auf der horizontalen Achse des Modells abgebildet wird, repräsentiert zum einen den zeitlichen Faktor sowie den gesamten Lebenszyklus des jeweiligen Prozessschrittes, welcher weiterhin in die Phasen Vorbereitung, Ausarbeitung, Konstruktion und Übergang untergliedert wird. In der Vorbereitungsphase erfolgen zunächst die Analyse der wesentlichen Geschäftsvorfälle sowie die Beschreibung der Endproduktvision. Weiterhin werden der Umfang des Projektes sowie die damit verbundenen Kosten und Risiken ermittelt. Innerhalb der Ausarbeitsphase erfolgt die Planung der Softwarearchitektur sowie der hierfür notwendigen Ressourcen. Das fertige Softwareprodukt bildet das Endergebnis der Konstruktionsphase. Die Übergangsphase bildet den Abschluss des Projektes, welcher durch die Qualitätsprüfung sowie anknüpfenden Schulungen und letztendlich durch die Übergabe des fertigen Softwareproduktes gekennzeichnet ist.<sup>31</sup>

## 5.2.4 Agile Vorgehensmodelle

Agile Vorgehensmodelle zur Softwareentwicklung entstanden aufgrund der Lean-Management-Bewegung in der japanischen Automobilindustrie. Nach diesen Ansätzen wird versucht, die Bearbeitung eines Projektes durch die systematische Vermeidung von Ressourcenverschwendungen zu verbessern und zu beschleunigen. Tätigkeiten, welche keinen direkten Nutzen für den Kunden bzw. keine Verbesserung des Projektfortschritts bewirken, sollen in diesem Zusammenhang nicht durchgeführt werden. Durch die Vermeidung dieser Arbeitsschritte soll der Kundennutzen fokussiert und der Mehrwert des Produktes gesteigert werden.<sup>32</sup>

Agile Vorgehensmodelle sollen, insbesondere in Domänen mit sich rasch ändernden Anforderungen, die genannten Vorteile aus der japanischen Automobilindustrie auf die Softwareentwicklung übertragen. Die Grundidee der agilen Vorgehensweise besteht in

---

<sup>29</sup> Vgl. Grechenig et al. 2010, S. 380.

<sup>30</sup> Vgl. Krcmar 2010, S. 153.

<sup>31</sup> Vgl. Krcmar 2010, S. 153.

<sup>32</sup> Vgl. Highsmith 2009, S. 33.

der Reduktion der Entwurfsphase auf ein Mindestmaß, der Erzeugung früh ausführbarer Softwareprodukte sowie in der Kundennähe.<sup>33</sup> Diese Idee wurde maßgeblich im Jahr 2001 durch eine Gruppe von Softwareentwicklern geprägt, die einen Erfahrungsaustausch im Zuge von Entwicklungsprojekten vornahmen. Das Ergebnis dieses Erfahrungsaustauschs war ein „agiles Manifest“, das aus verschiedenen Grundsätzen bestand, die durch zwölf weitere Prinzipien konkretisiert wurden. Die Grundsätze dieses Manifests waren im Folgenden:<sup>34</sup>

- Personal mit hoher Qualifizierung sowie die effiziente Zusammenarbeit zwischen den Personen ist eine höhere Bedeutung zuzumessen als Prozessdefinitionen und Werkzeuge.
- Software, die den Anforderungen der Anwender entspricht und eine gute Funktionalität aufweist, ist bedeutender als eine ausführliche Dokumentation.
- Eine verstärkte Zusammenarbeit mit den Endanwendern hat eine größere Bedeutung gegenüber einem ausführlichen Vertragswerk.
- Vorschläge für notwendige Änderungen einbringen zu können weist eine höhere Relevanz auf als das ausschließliche Befolgen eines vordefinierten Plans.

Im Gegenzug zu klassischen Vorgehensmodellen zeichnen sich agile Vorgehensmodelle in der Softwareentwicklung durch relativ kurze Iterationen aus, wobei nach jeder Iteration ein für den Kunden greifbares Resultat erreicht wird, z. B. lauffähige Software. Diese Vorgehensweise wirkt sich insbesondere auf das Anforderungsmanagement aus, da der Umgang mit Anforderungen in solchen einfachen Vorgehensmodellen grundsätzlich auf einer anderen Basis stattfindet als bei komplexeren Vorgehensmodellen. Auf dieser Grundlage können Anforderungsveränderungen bei jeder Iteration neu berücksichtigt werden, sodass keine Grenze für die Aufnahme von Change Requests (CR) existiert. Bei jeder neuen Iteration erfolgt demnach die Entscheidung, welche Anforderungen in der aktuellen Iteration realisiert werden können.<sup>35</sup>

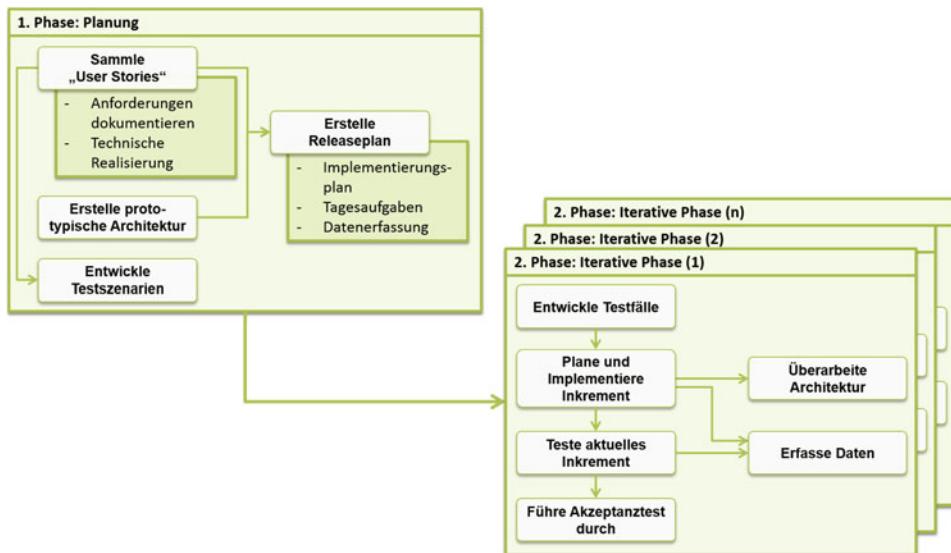
Ein erstes agiles Vorgehensmodell zur Softwareherstellung wurde 1999 durch Kent Beck vorgestellt und als Extreme Programming (XP) bezeichnet. XP sieht eine strikte Arbeitsteilung zwischen den Kunden und Entwicklern vor. Während die Planung und Beschreibung von Vorgängen dem Kunden obliegt, besteht die Aufgabe der Entwicklung in der Umsetzung der Prozesse zu einem Softwareprodukt sowie in der Rückmeldung über Erfolg oder Misserfolg. Zur Vereinfachung der Kommunikation und zur Verbesserung des Wissenstransfers zwischen Kunde und Entwickler werden durch das XP einfache, zusammengehörige Praktiken zur Verfügung gestellt.<sup>36</sup> Abbildung 5.8 zeigt den allgemeinen Prozessverlauf beim Extreme Programming.

<sup>33</sup> Vgl. Hruschka et al. 2009, S. 2.

<sup>34</sup> Vgl. Tremp und Ruggiero 2011, S. 16.

<sup>35</sup> Vgl. Tremp und Ruggiero 2011, S. 16

<sup>36</sup> Vgl. Bunse und Knethen 2008, S. 116.



**Abb. 5.8** Extreme Programming zur Softwareentwicklung. (Eigene Erstellung, in Anlehnung an Bunse und Knethen 2008, S. 115)

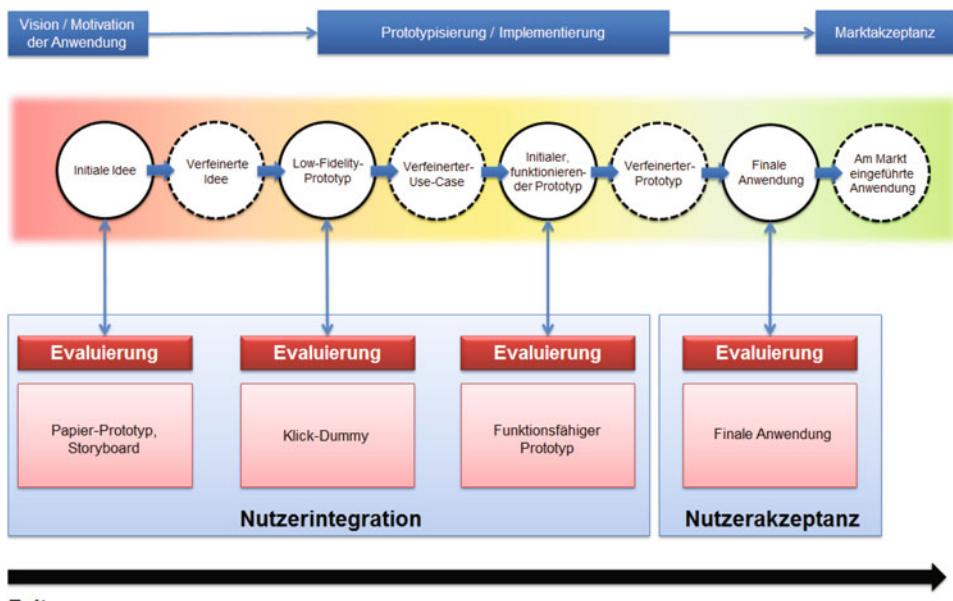
An der Vorgehensweise des XP bestehen zahlreiche Kritikpunkte, die sich auf die fehlende Entwurfsplanung sowie die ständige Beteiligung des Kunden beziehen. Balzert formuliert weiterhin den Einwand, dass der Einsatz des Extreme Programming vor allem in kleineren Projekten negative Auswirkungen auf die Wirtschaftlichkeit des gesamten Projektes hervorbringt.<sup>37</sup> Entgegengesetzt zur bestehenden Kritik an den Methoden des XP ist dennoch, aufgrund des niedrigen Verwaltungsaufwands sowie der expliziten Berücksichtigung technischer und sozialer Aspekte der Entwickler, ein vermehrter Einsatz des Vorgehensmodells zu registrieren.<sup>38</sup>

Bisher wurden Vorgehensmodelle betrachtet, die überwiegend zur Entwicklung von Anwendungen für stationäre Einsatzbereiche konzipiert wurden. Im Gegensatz dazu existieren für die Entwicklung mobiler Applikationen fast keine Vorgehensmodelle. Entwickler stehen somit vor der besonderen Herausforderung, die bereits vorhandenen Vorgehensmodelle auf die mobile Anwendungsentwicklung zu übertragen. Durch die jedoch stark ansteigende Marktentwicklung sowie die immer kürzer ausfallende Entwicklungszeit mobiler Endgeräte ist die Anpassung ausgereifter Vorgehensmodelle nicht immer möglich.<sup>39</sup> Ein Vorgehensmodell, welches Softwarehersteller die Möglichkeit bietet, erfolgreiche mobile Applikationen mittels nutzerorientierten Produktdefinitions- und Entwicklungsprozessen zu entwickeln, ist das Ubiquitous Computing Application

<sup>37</sup> Vgl. Balzert 2009, S. 657.

<sup>38</sup> Vgl. Ruf und Fittkau 2008, S. 46.

<sup>39</sup> Vgl. Bitkom 2012.



**Abb. 5.9** UCAN-Modell zur mobilen Anwendungsentwicklung. (Eigene Erstellung, in Anlehnung an Krcmar 2010, S. 683)

Development and Evaluation Process-Modell (UCAN), welches in Abb. 5.9 vorgestellt wird.

Das UCAN-Modell vereinigt die Ansätze des Wasserfall- und V-Modells und ermöglicht die Integration der Endnutzer in den Entwicklungsprozess. Dies erhöht zum einen den Grad der Nutzerakzeptanz und vermindert gleichzeitig die Fehlerquote innerhalb der Entwicklungsphase. Ein wichtiger Arbeitsschritt dieser Entwicklungsstrategie besteht in der Evaluation der Anwendung zu unterschiedlichen Zeitpunkten während der Implementierungsphase. Die Evaluierung wird in Zusammenarbeit mit den späteren Endnutzern nach jeder Design- und Implementierungsphase durchgeführt. Dadurch wird letztlich eine Anwendung geschaffen, die exakt auf die Bedürfnisse des Nutzers zugeschnitten ist. Der Prozessablauf wird nachfolgend kurz aufgelistet:<sup>40</sup>

- In einem ersten Schritt werden zuerst initiale Ideen evaluiert.
- Danach erfolgt in einem zweiten Schritt, ausgehend von einer verfeinerten Idee, die Evaluierung der ersten Prototypen ohne Funktionalitäten.
- Der erste funktionsfähige Prototyp wird im nachfolgenden Schritt, ausgehend von einer verfeinerten Anforderung, evaluiert.
- Im vierten und letzten Schritt erfolgt die Evaluierung des finalen Endproduktes.

<sup>40</sup> Vgl. Krcmar 2010, S. 683.

Die Wahl und Anwendung eines Vorgehensmodells ist entscheidend für den Erfolg des gesamten Entwicklungsvorhabens. Aus diesem Grund besteht zu Beginn des Entwicklungsprojektes die Notwendigkeit, ein geeignetes Vorgehensmodell zu identifizieren. Hierfür müssen für die Umsetzung des Entwicklungsprojektes infrage kommende Vorgehensmodelle gegenübergestellt und bewertet werden. Innerhalb des nachfolgenden Kapitels werden die in den vorangegangenen Abschnitten vorgestellten klassischen, modernen und agilen Vorgehensmodelle nochmals betrachtet und anhand unterschiedlicher Kriterien voneinander abgegrenzt und bewertet.

### 5.2.5 Bewertungsverfahren zur optimalen Modellauswahl

Die folgenden Ausführungen befassen sich mit der praktischen Auswahl eines Vorgehensmodells zur Anwendungsentwicklung. Hierfür wurden zuvor bereits klassische, moderne und agile Modelle gegeneinander abgegrenzt sowie deren grundlegende Eigenschaften und Ausprägungen erläutert. Das nachfolgende Vorgehen zur Auswahl eines geeigneten Modells orientiert sich an einzelnen Modellkriterien. Entscheidende Faktoren bilden hierbei die jeweiligen Ziele, Konzepte sowie Projektzyklen der bisher betrachteten Vorgehensmodelle. Zur Analyse sowie qualitativen Bewertung der jeweiligen Modelleigenschaften und -bedingungen erfolgte die Aufstellung eines Kriterienkatalogs. Anhand der im Katalog enthaltenen Basis- und Modellkriterien wurde die Beurteilung der Modelle unter Verwendung einzelner Bewertungsmatrizen vorgenommen. Die betrachteten Modellkriterien wurden jeweils durch eine Punktzahl bewertet. Die Vergabe von drei Punkten bedeutet, dass ein jeweiliges Kriterium gut ausgeprägt oder vorhanden ist. Die Angabe von zwei Punkten bedeutet, dass ein Kriterium mittelmäßig, bei einem Punkt nur mäßig ausgeprägt ist. Die Vergabe von keinem Punkt bedeutet, dass ein Kriterium nicht ausgeprägt oder nicht vorhanden ist.

Ein wichtiges Merkmal jedes konkreten Vorgehensmodells ist die Ausrichtung an eine gegebene Problemsituation. Erstes Kriterium bei der Wahl eines Vorgehensmodells ist somit die Anwendbarkeit an die gegebene Aufgabenstellung. Weiterhin müssen Vorgehensmodelle eine gewisse Flexibilität aufweisen, um außerplanmäßige Änderungen bei der Softwareentwicklung berücksichtigen zu können. Werden bereits zu Beginn der Softwareentwicklung wiederkehrende Aufgabenpakete registriert, bietet es sich an, ein Vorgehensmodell anzuwenden, welches die iterative Abwicklung gleichartiger Aktivitäten ermöglicht. Ein Vorgehensmodell muss weiterhin leicht verständlich und einfach strukturiert sein.<sup>41</sup>

Die betrachteten Vorgehensmodelle lassen sich hinsichtlich ihrer Grundprinzipien charakterisieren (vgl. Abb. 5.10).

Unabhängig von den jeweiligen Grundprinzipien und der vorliegenden Problemstellung müssen Vorgehensmodelle die einzelnen Phasen des Softwareentwicklungsprozesses

---

<sup>41</sup> Vgl. Bunse und Knethen 2008, S. 101.

Grundprinzipien	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Anwendbarkeit, Praxisnähe	3	2	2	3	3	3
Flexibilität des Modells	1	2	1	3	2	2
Iterationen vorgesehen	0	3	3	3	3	0
Verständlichkeit, Einfachheit	3	2	3	2	3	3
<b>Summe</b>	<b>7</b>	<b>9</b>	<b>9</b>	<b>11</b>	<b>11</b>	<b>8</b>

**Abb. 5.10** Bewertung der Vorgehensmodelle hinsichtlich ihrer Grundprinzipien

Phasenunterstützung	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Anforderungsermittlung	3	3	3	3	3	3
Analyse	3	3	3	3	3	3
Entwurf	3	3	3	3	3	3
Implementierung	3	3	3	3	3	3
Test	3	3	3	3	3	3
Qualitätssicherung	3	0	0	0	0	3
<b>Summe</b>	<b>18</b>	<b>15</b>	<b>15</b>	<b>15</b>	<b>15</b>	<b>18</b>

**Abb. 5.11** Bewertung der Vorgehensmodelle hinsichtlich ihrer Phasenunterstützung

unterstützen, d. h. Anforderungsermittlung, Analyse, Entwurf, Implementierung, Test und Qualitätssicherung. Des Weiteren müssen durch die Modelle Aktivitäten zum Projektmanagement sowie zur Dokumentation des Projektverlaufs aufweisen.<sup>42</sup> Die Ausprägungen zur Phasenunterstützung der genannten Vorgehensmodelle werden in Abb. 5.11 aufgezeigt.

Neben der Unterstützung der Phasen des Softwareentwicklungsprozesses erhoffen sich Softwarehersteller durch den Einsatz von Vorgehensmodellen, nützliche Werkzeuge bereitgestellt zu bekommen. Hierbei ist der Einsatz verschiedener Werkzeuge denkbar, bspw. für die Modellierung von Anwendungsfällen, die Erstellung lauffähiger Prototypen oder zur Evaluierung des Projekterfolges.<sup>43</sup> Abbildung 5.12 stellt die Möglichkeiten zur Werkzeugunterstützung durch die genannten Vorgehensmodelle dar.

Ein weiterer wichtiger Punkt bei der Auswahl von Vorgehensmodellen ist deren Größe bzw. Komplexität. Die Komplexität eines Vorgehensmodells kann sich hierbei durch verschiedene Aspekte ausdrücken, bspw. die Anzahl der durchzuführenden Phasen, den Anteil an benötigten Mitarbeitern zur Projektrealisierung sowie den Umfang des Softwareprojektes. Insbesondere in Bezug auf die Projektgröße und die verfügbare Anzahl an Mitarbeitern bildet die Komplexität ein entscheidendes Auswahlkriterium. Oftmals

<sup>42</sup> Vgl. Bunse und Knethen 2008, S. 101.

<sup>43</sup> Vgl. Bunse und Knethen 2008, S. 121.

Werkzeuge	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Erstellung von Modellen	0	0	0	3	0	0
Entwicklung von Prototypen	0	3	0	0	3	3
Evaluierung und Validierung	3	3	3	3	0	3
<b>Summe</b>	<b>3</b>	<b>6</b>	<b>3</b>	<b>6</b>	<b>3</b>	<b>6</b>

**Abb. 5.12** Bewertung der Vorgehensmodelle hinsichtlich ihrer Werkzeugunterstützung

Komplexität	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Kompakt, für kleinere Projekte geeignet	3	1	2	1	3	3
Mittel	2	2	2	1	2	3
Umfangreich, für große Projekte geeignet	1	3	3	3	1	2
<b>Summe</b>	<b>6</b>	<b>6</b>	<b>7</b>	<b>5</b>	<b>6</b>	<b>8</b>

**Abb. 5.13** Bewertung der Vorgehensmodelle hinsichtlich ihrer Komplexität

können kleinere Projektteams die hohen Anforderungen komplexer Vorgehensmodelle nicht meistern, was letztendlich zum Scheitern des gesamten Softwareprojektes führen kann.<sup>44</sup> In Bezug auf die Anwendbarkeit auf Softwareprojekte ergibt sich für die genannten Vorgehensmodelle folgendes Bild (vgl. Abb. 5.13).

Obwohl Softwareentwickler gegenwärtig auf eine Vielzahl an unterschiedlichen Vorgehensmodellen zur Softwareentwicklung zurückgreifen können, zeigte die Analyse, dass bisher noch kein einheitliches und standardisiertes Vorgehensmodell für Softwareprojekte vorhanden ist.<sup>45</sup> Ziel der Analyse bestand somit weniger in der Identifikation eines allgemeingültigen und universalen Vorgehensmodells, sondern vielmehr in der Untersuchung und Auswahl eines etablierten klassischen, modernen oder agilen Modells zur bestmöglichen Unterstützung bei der Entwicklung einer mobilen Applikation zur Baufinanzierung.

Das RUP-Modell sowie das Extreme Programming stellten bestmögliche Alternativen bzgl. der Betrachtung der Grundprinzipien dar. Beide Modelle weisen eine hohe Praxisnähe sowie Flexibilität auf. Die bestmögliche Phasenunterstützung bieten das Wasserfallmodell sowie das UCAN-Modell. Im Gegensatz zu den anderen Vorgehensmodellen betrachten beide Modelle Aktivitäten und Aufgaben der Qualitätssicherung. Des Weiteren bieten das Spiral-, RUP- und UCAN-Modell eine optimale Unterstützung des

<sup>44</sup> Vgl. Bunse und Knethen 2008, S. 119.

<sup>45</sup> Vgl. Bunse und Knethen 2008, S. 1 f.

Entwicklungsprozesses durch nützliche Werkzeuge zur Modellierung, zum Prototyping sowie zur Evaluierung. Die Analyse in Bezug auf die Komplexität und Anwendbarkeit auf unterschiedlich ausgeprägte Softwareprojekte von Vorgehensmodellen ergab weiterhin, dass insbesondere das UCAN-Modell die beste Alternative für kleine, mittlere und große Softwareprojekte darstellt.

Die gerade aufgezeigten Bewertungsverfahren können für aufkommende Entwicklungsprojekte adaptiert werden. An dieser Stelle soll jedoch nochmals festgehalten werden, dass die aufgestellten Bewertungen Unterschiede und Gemeinsamkeiten der betrachteten Vorgehensmodelle aufzeigen und verdeutlichen sollten. Da die Bewertungen nicht als allgemeingültig angesehen werden können, ist eine direkte Umsetzung und Anwendung auf jedes vorliegende Softwareprojekt nicht immer möglich. In der Praxis sollte die Wahl eines geeigneten Vorgehensmodells von dem Projektkontext, dem Kunden, den zur Verfügung stehenden Ressourcen sowie von der Zielstellung abhängig gemacht werden.

### 5.2.6 Life-Cycle-Management mobiler Applikationen

Mit dem Begriff Software-Life-Cycle wird die Zeitspanne beschrieben, in der ein Softwaredreprodukt geplant, entwickelt und bis zum Ende seiner Nutzung eingesetzt wird. Ebenso wird hiermit die Vorstellung einer Strukturierung des Zeitraums und der damit zusammenhängenden Aktivitäten und Ergebnisse in Phasen verbunden. In diesem Zuge erfolgen zudem die Festlegung der Reihenfolge und die Beziehungen zwischen diesen Phasen.<sup>46</sup>

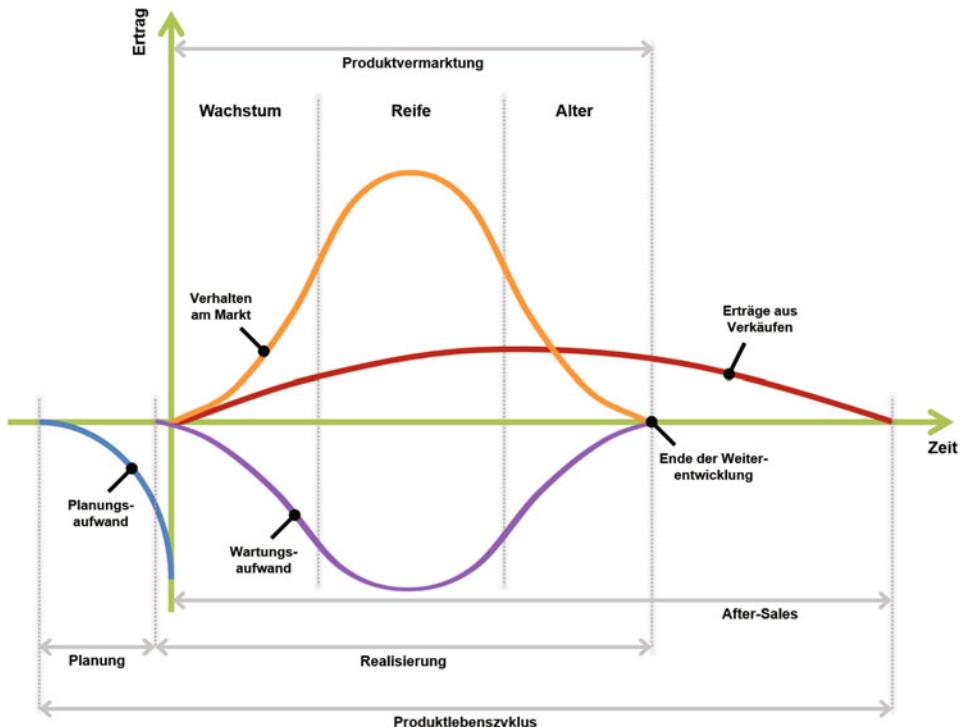
In der Literatur finden sich zahlreiche klassische Produktlebenszyklusmodelle zur Softwareentwicklung, die hauptsächlich die Phasen Einführung, Wachstum, Reife, Sättigung und Niedergang umfassen. Traditionelle Lebenszykluskonzepte stehen jedoch oftmals in der Kritik, da sie die effektive Verweildauer eines Produktes sowie vor- und nachgelagerte Phasen unberücksichtigt lassen.<sup>47</sup> Moderne Lebenszyklusmodelle mobiler Applikationen bestehen bisher nur aus technischer Sicht. In diesem Zusammenhang erfolgt die Betrachtung unterschiedlicher Systemzustände bei der Nutzung mobiler Applikationen. Wird eine Applikation durch den Nutzer gestartet erfolgt der Übergang in den Zustand „Loaded“. Die Nutzung der Applikation wird durch den Zustand „Active“ dargestellt. Der Übergang in den Zustand „Paused“ bezeichnet entweder das Pausieren der Applikationen oder den Wechsel zwischen zwei Applikationen. Eine Applikation ist nach diesem Lebenszyklus genau dann beendet, wenn der Zustand „Destroyed“ erreicht wurde.<sup>48</sup>

Aus betriebswirtschaftlicher Sichtweise wurde bisher kein allgemeingültiger Produktlebenszyklus für mobile Applikationen aufgestellt. Daher wird im Folgenden die Aufstellung eines idealtypischen Lebenszykluskonzeptes für mobile Applikationen vorgenommen. Als Grundlage hierfür dient zum einen der Ansatz von Adizes, der hauptsächlich

<sup>46</sup> Vgl. Pomberger und Pree 2004, S. 11.

<sup>47</sup> Vgl. Blinn et al. 2008, S. 132.

<sup>48</sup> Vgl. Kuassi und Bischel 2012, S. 133.



**Abb. 5.14** Idealtypischer Produktlebenszyklus einer mobilen Anwendung. (Eigene Erstellung, in Anlehnung an Adizes 1988, S. 84 und Blinn et al. 2010, S. 716)

Wachstumsmuster von Organisationen beschreibt<sup>49</sup>, und zum anderen der erweiterte Produktlebenszyklus der hybriden Wertschöpfung nach Blinn<sup>50</sup>. Zur Berücksichtigung der bereits genannten Kritik an bestehenden traditionellen Produktlebenszyklusmodellen wurden diese Ansätze um vor- und nachgelagerte Phasen innerhalb der mobilen Anwendungsentwicklung erweitert. Abbildung 5.14 zeigt den Vorschlag eines idealtypischen Lebenszyklus für mobile Anwendungen.

Der Planungsaufwand stellt den Ertragsverlauf der Planungsphase dar. Hierunter fallen alle Aufwendungen für die Produktentwicklung, die bis zur Realisierung der Applikation anfallen. Zu den Aktivitäten innerhalb der Planung zählen die Ideenfindung, Konzeption, der Entwurf sowie die Implementierung. Die Planungsaktivitäten sind abgeschlossen, sobald die mobile Applikation in die Realisierungsphase übergeht. Die Vermarktung der mobilen Anwendung wird durch die Kurve „Verhalten am Markt“ dargestellt. Zu Beginn der Realisierungsphase befindet sich die Applikation in der Wachstumsphase und muss sich erst am Markt etablieren. Innerhalb der Reifephase wirft die Applikation die größ-

<sup>49</sup> Vgl. Adizes 1988, S. 84.

<sup>50</sup> Vgl. Blinn et al. 2010, S. 716.

ten Erträge ab, bis sie schließlich gegen Ende der Alterungsphase entweder durch einen Relaunch neu auf den Markt gebracht wird oder durch die Entwickler nicht weiter betrachtet wird. Parallel zur Produktvermarktung verläuft der Wartungsaufwand, der seinen Ursprung zeitlich vor dem Erscheinen der Applikation im Markt hat. Grund hierfür sind mögliche Wartungsarbeiten, die bereits gegen Ende der Planungsphase entstehen können. Innerhalb der Produktvermarktung steigt der Wartungsaufwand mit dem Verhalten am Markt an und schmälert somit die erzielten Erträge. Steigende Kundenansprüche, technologische Neuerungen oder zwingende Softwareupdates beschreiben nur einige Gründe für den Verlauf des Wartungsaufwandes. Abschließend wird innerhalb des Produktlebenszyklus die After-Sales-Phase betrachtet. Diese beginnt mit der Fertigstellung und Verbreitung der Applikation. Erträge aus den Verkäufen können auch noch nach dem Entwicklungsende erzielt werden.<sup>51</sup> Der Verlauf jeder Kurve innerhalb des vorgeschlagenen Produktlebenszyklus ist stark von der Applikationsausprägung sowie von der Qualität der Applikation abhängig. Bei der Entwicklung mobiler Applikationen müssen somit innerhalb der Planungsphase vorhandene Applikationstypen auf den Anwendungsbereich analysiert und ausgewählt werden.

---

### 5.3 Markt- und Problemanalyse

Die Markt- und Problemanalyse umfasst die marktgerichteten Grundüberlegungen zur mobilen Anwendungsentwicklung, bspw. welche Zielgruppe am Markt angesprochen und welche Ziele durch die Entwicklung erreicht werden sollen. In dieser Phase kann im Wesentlichen zwischen den Spezifika von individuellen Kundenprojekten und von Marketplace-Projekten unterschieden werden. Während bei der Ausgangssituation eines Kundenprojektes im B2B-Bereich eher eine kundenseitige Anforderungsentwicklung und Problemanalyse von mobilen Anwendungen für die Planungsphase von hoher Bedeutung ist, muss bei Marketplace-Projekten im B2C-Bereich tendenziell dem Ideen- und Kreativitätsmanagement eine hohe Bedeutung zugemessen werden. Nachfolgende Abbildung verdeutlicht wesentliche Aspekte, die im Zuge der Marktphase zur Anwendungsentwicklung zu unterscheiden sind (Abb. 5.15).

In der Phase der Markt- und Problemanalyse ist bei der Entwicklung von B2B-Projekten die Erhebung von Anforderungen und die Durchführung einer strukturierten Projektform zu empfehlen. Die auf den B2C-Markt gerichteten Entwicklungen können aufgrund des indirekten Kundenbezugs eine stärkere Fokussierung auf das eigene Ideenmanagement und die Anforderungserhebung erfordern, die vor der eigentlichen Entwicklung und Bereitstellung im Online-Store erforderlich ist.

---

<sup>51</sup> Vgl. Blinn et al. 2008, S. 716 f.



**Abb. 5.15** Elemente in der Marktphase

### 5.3.1 Markt- und kundenbezogene Entwicklung mobiler Anwendungen

Die mobile Anwendungsentwicklung unterscheidet in der Konzeptionsphase insbesondere zwischen Kundenprojekten und Eigenentwicklungen für verschiedene Online-Stores. Kundenauftragsbezogene Entwicklungen sind dadurch gekennzeichnet, dass der Kunde in den gesamten Entwicklungsprozess des Software Engineering integriert wird. Die Unterteilung des Prozesses in einzelne Projektphasen zu definierten Meilensteinen kann als Vorgehensweise in Kundenprojekten empfohlen werden. Die Anforderungserhebung und Konzeptionierung weisen in Kundenauftragsprojekten eine besondere Bedeutung auf, da die aufgenommenen und umgesetzten Anforderungen ebenfalls als Kriterien für die abschließende Projektabschluss genutzt werden können.

Für die Umsetzung von Kundenauftragsprojekten existiert im Software Engineering eine Vielzahl von definierten Aktivitäten, die insbesondere die Phasen der Problemanalyse, Systemspezifikation und Planung unterstützen.<sup>52</sup> Die Nutzung von Methoden und Vorgehensmodellen des Software Engineering für die mobile Anwendungsentwicklung

<sup>52</sup> Vgl. Pomberger und Pree 2004, S. 12.



**Abb. 5.16** Kunden- und marketplacebezogene Entwicklungsprojekte

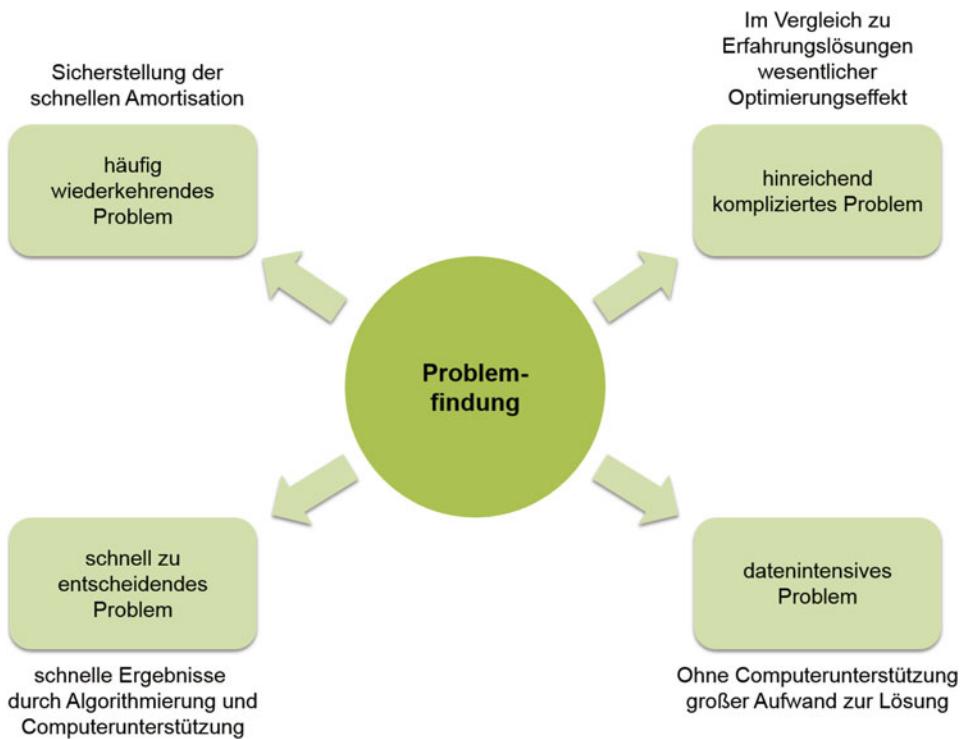
ist je nach Anwendungsfall zu entscheiden. In der Planungs- und Konzeptionsphase sehen die Modelle des Software Engineering die Erstellung von Lasten- und Pflichtenheften vor, um die Kundenanforderungen sowie die ersten Systemspezifikationen und die für die Umsetzung relevante Projektplanung weiter zu detaillieren (Abb. 5.16).

Während bei der kundenauftragsbezogenen Anwendungsentwicklung die gemeinsame Erarbeitung der Problemanalyse und Anforderungen im Vordergrund steht, ist in Bezug auf eine Veröffentlichung der Anwendung über einen Marketplace die Durchführung einer Marktanalyse zu empfehlen. Aufgrund der Fülle von mobilen Anwendungen sollten neben der Beobachtung des Marktes Konkurrenzanalysen durchgeführt werden. Während bei Kundenprojekten ein definiertes Projektziel und eine Abnahme erfolgt, gilt es bei Marketplace-Projekten eine stärkere Bekanntmachung der eigens entwickelten Applikationen zu erreichen, wie z. B. durch Werbeanzeigen, kostenlose Probeversionen und Anzeigen und Berichte in Foren und Zeitschriften (vgl. Kap. 2).

Neben der Marktbeobachtung sind damit auch eine umfassende Validierung der eigenen Produktidee und eine Konkurrenzanalyse verbunden. Insbesondere bei der Vielzahl mobiler Anwendungen, die über verschiedene Marketplaces angeboten werden, ist es wichtig, besondere Funktionalitäten zu konzipieren und umzusetzen, um eine Unterscheidung vom bisherigen Angebot zu erreichen. Die anschließende Veröffentlichung und Bekanntmachung sind ebenso von hoher Bedeutung, um die Publizität der entwickelten Anwendung zu gewährleisten.

### 5.3.2 Problemanalyse und Zielbindung

Das Ziel der Problemanalyse und Zielbildung besteht darin, den Einsatzbereich, für den eine mobile Anwendung angestrebt wird, festzustellen und zu dokumentieren. In diesem Zusammenhang müssen alle Aktivitäten und Tätigkeiten in Verbindung mit der



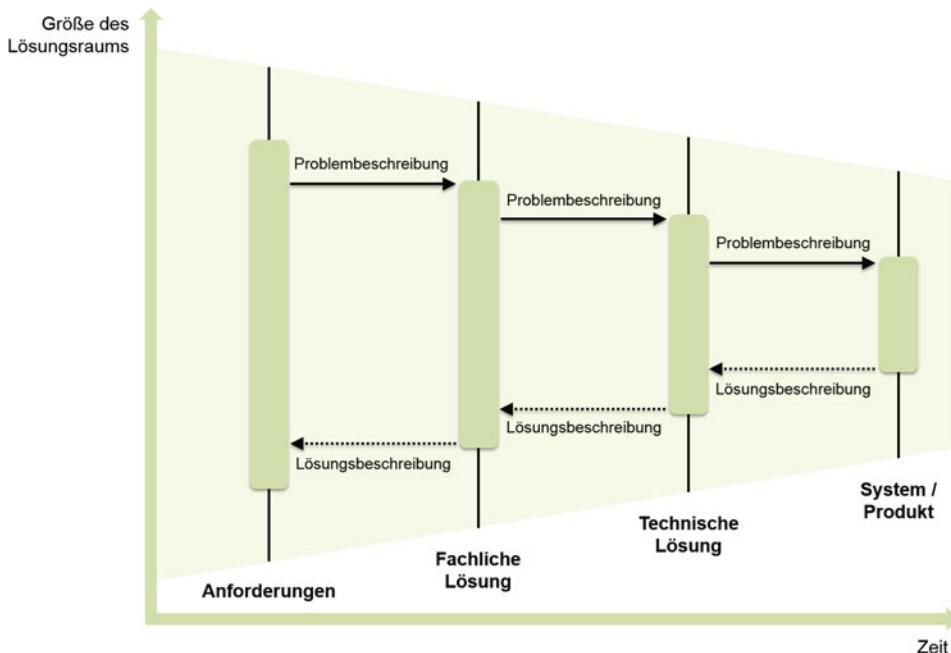
**Abb. 5.17** Merkmale zur Problemfindung. (Eigene Erstellung, in Anlehnung an Lassmann 2006, S. 419)

Entwicklung der Applikation sowie deren Wechselwirkungen als auch für die Realisierung des Projektes notwendige technische, personelle, finanzielle und zeitliche Ressourcen berücksichtigt werden. Die Erhebung des Istzustandes bzw. der Problemstellung sowie die Abgrenzung des Problembereichs stellen wichtige Tätigkeiten im Rahmen der Problemanalyse dar.<sup>53</sup>

Wie bereits zu Beginn des Kapitels erwähnt, können mobile Anwendungen entweder im Auftrag oder für bestimmte Marketplaces entwickelt werden. Je nach Ausrichtung des Entwicklungsprojektes liegen unterschiedliche Herausforderungen und Probleme vor. Im Zuge der Problemfindung bzw. des Problembewusstseins haben sich die in Abb. 5.17 dargestellten Merkmale als zweckmäßig erwiesen, um bedeutungsvolle Problemstellungen auszuwählen. Anzumerken ist, dass nicht immer alle vier Merkmale gleichzeitig vorhanden sein müssen, um das vorliegende Problem genau zu definieren. In Hinblick auf erste Aufwand-Nutzen-Überlegungen geben die Merkmale jedoch wichtige Anhaltspunkte.<sup>54</sup>

<sup>53</sup> Vgl. Pomberger und Pree 2004, S. 12.

<sup>54</sup> Vgl. Lassmann 2006, S. 419.



**Abb. 5.18** Problem- vs. Lösungsbeschreibung im Entwicklungsprozess. (Eigene Erstellung, in Anlehnung an Balzert 2009, S. 437)

Neben der Erfassung des Problembewusstseins soll durch die Durchführung einer Problemanalyse eine vollständige Erfassung des Problems mit allen zugehörigen Rahmenbedingungen erzielt und die Durchführbarkeit des Entwicklungsprojektes untersucht werden. Obwohl die Problemanalyse der Anforderungsdefinition, dem Systementwurf und der Systemimplementierung vorgelagert ist, werden innerhalb der Analyse Ist- und Sollkonzepte einander gegenübergestellt, eine generelle Lösbarkeit der Problemsituation ermittelt und vorläufige Projektpläne aufgestellt. Daher ergeben sich aus der Problemanalyse erste Anforderungen an die Softwarelösung, die im Laufe des Entwicklungsprozesses, aufgrund der zunehmenden Verringerung des Lösungsraums, weiter modifiziert, spezifiziert oder sogar verworfen werden müssen (vgl. Abb. 5.18).<sup>55</sup>

Die Einhaltung konkreter Ziele ist weiterhin eine fundamentale Voraussetzung für die Entwicklung erfolgreicher Softwareprojekte. Ziele müssen insbesondere unter der Berücksichtigung von Kosten, Qualität und Terminen definiert und ausreichend charakterisiert werden. Die Zielbestimmung unterscheidet hierbei globale und projektspezifische Ziele. Letztere lassen sich anhand der Phasen des jeweils eingesetzten Vorgehensmodells zur Softwareentwicklung ableiten. Projektbezogene Ziele beziehen sich aus dem Herleiten inhaltlicher Vorgaben aus der Aufgabenstellung und definieren somit das zu entwickeln-

<sup>55</sup> Vgl. Pohl 2008, S. 20 f.

de Softwareprodukt.<sup>56</sup> Für die Auswahl von Zielen sollten folgende drei Komponenten berücksichtigt werden:<sup>57</sup>

- Die Wichtigkeit eines Ziels gibt an, wie bedeutsam eine Person oder eine Gruppe die Erreichung dieses Ziels erachtet.
- Die Dringlichkeit eines Ziels gibt an, zu welchem Zeitpunkt ein Ziel erreicht werden muss oder kann bzw. bis wann notwendige Aktionen zur Erreichung des Ziels durchgeführt werden müssen.
- Die Erfolgswahrscheinlichkeit eines Ziels gibt an, mit welcher Sicherheit die Erreichung des Ziels gelingen wird.

Durch die genannten Komponenten können Ziele besser bewertet und dadurch Prioritäten für die Auswahl eines zu verfolgenden Ziels festgelegt werden. Für die endgültige Entscheidung für die Auswahl eines Ziels sind somit alle drei Komponenten gleich bedeutsam. Beispiele für Ziele, die durch den Vertrieb einer mobilen Applikation verfolgt werden sollen, können wie folgt lauten:

---

#### Beispiel

- Durch den Vertrieb soll innerhalb von einem Jahr eine Erweiterung des Produktportfolios sowie ein verbesserter Kundenservice ermöglicht werden.
- Durch den Vertrieb wird eine Verringerung der Kosten in den Bereichen Vertrieb und Logistik von zehn Prozent angestrebt.

### 5.3.3 Werkzeuge zur Unterstützung der Markt- und Problemanalyse

Im Rahmen der mobilen Anwendungsentwicklung für den B2C-Markt empfiehlt sich für die Erhebung unklarer Anforderungen oder bei Projekten mit hoher Systemkomplexität der Einsatz von Kreativitätstechniken. Insbesondere bei Applikationen mit hohem Innovationsgrad, die sich kaum oder überhaupt nicht von bestehenden mobilen Anwendungen unterscheiden lassen, können durch verschiedene Methoden und Techniken die Kreativität bei der Softwareentwicklung gesteigert und damit neuartige Anforderungen oder innovative Lösungen hervorgebracht werden.<sup>58</sup>

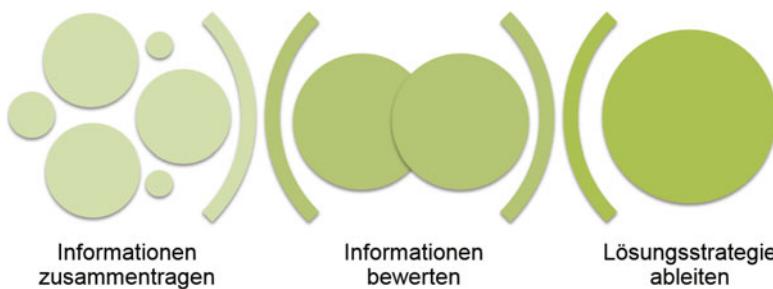
Durch die Anwendung kreativitätsfördernder Techniken sollen, aufbauend auf der zuvor aufgestellten Problemdefinition, Lösungsstrategien gefunden und weiterhin eine optimale Systemstruktur für die Umsetzung des Entwicklungsvorhabens festgelegt wer-

---

<sup>56</sup> Vgl. Gernert 2003, S. 57.

<sup>57</sup> Vgl. Meck 2009, S. 13 ff.

<sup>58</sup> Vgl. Tremp und Ruggiero 2011, S. 63.



**Abb. 5.19** Prozess von der auslösenden Idee bis zur finalen Lösungsstrategie

den.<sup>59</sup> Patzak gibt für die Entwicklung solcher Lösungsstrategien folgende Grundregeln vor:<sup>60</sup>

1. Es soll nicht von vorhandenen Lösungen ausgegangen werden sondern von Idealvorstellungen. Aus der Idealvorstellung soll durch Detaillierung das realisierbare System abgeleitet werden.
2. Es sind alternative Lösungen zu entwickeln und die beste ist auszuwählen. Für die Optimalität einer einzelnen Lösung gibt es keinen absoluten Maßstab. Aus Kombinationen von Lösungen können neue, bessere Lösungsalternativen entwickelt werden.
3. Die Bewertung und die Kritik der Lösungen sollen so spät wie möglich durchgeführt werden.

Diese eher allgemeingültig aufgestellten Regeln geben dennoch wesentliche Empfehlungen beim Einsatz von Kreativitätstechniken für die Generierung sowie bei der Entwicklung innovativer mobiler Anwendungen.

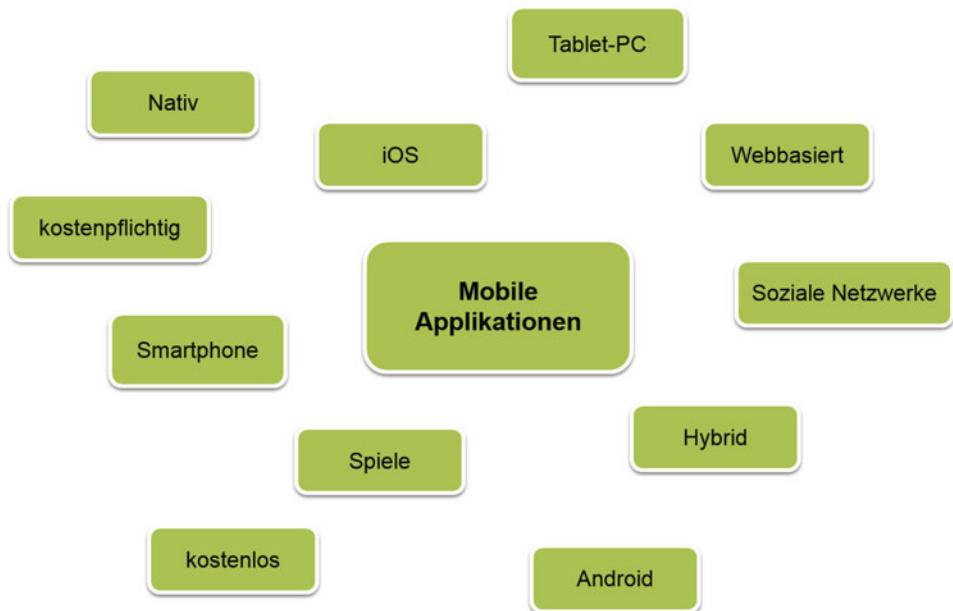
Der Prozess von der auslösenden Idee bis zur finalen Lösungsstrategie unterteilt sich in die Phasen Informationen zusammentragen, Informationen bewerten und Lösungsstrategie ableiten (vgl. Abb. 5.19). Jede Phase enthält unterschiedliche Techniken für das Zusammentragen, die Bewertung und Dokumentation von Informationen, die während der Ideengenerierung durch die Anwendung kreativitätsfördernder Techniken erhoben werden. Nachfolgend werden ausgewählte Techniken vorgestellt.

### 5.3.3.1 Techniken des Zusammentragens von Informationen

Das Sammeln und Zusammentragen von Informationen ist eine umfassende Tätigkeit, die einerseits eine fast unbegrenzte Fülle an Erhebungs- und Recherchemethoden aufweist, andererseits dadurch die wissenschaftliche Auseinandersetzung mit diesem Prozess sowie einen praxisorientierten Zugang zur Tätigkeit des „Sammeln von Informationen“

<sup>59</sup> Vgl. Vogel-Heuser 2003, S. 47.

<sup>60</sup> Patzak 1982, S. 185.



**Abb. 5.20** Brainstormingbeispiel zum Thema „Mobile Applikationen“

erschwert.<sup>61</sup> Auf der anderen Seite haben sich unterschiedliche Prozeduren aus der Vielzahl an Möglichkeiten etabliert, die im Laufe der Zeit als besonders effizient eingestuft wurden und den Suchprozess von Informationen erleichterten.

Eine besonders häufig verwendete und weithin bekannte Methode, um Informationen zu sammeln, ist das Brainstorming. Die Idee hinter dem Brainstorming ist es, in einer freien, offenen und kritikfreien Form Informationen zu einem bestimmten Thema zusammenzutragen. Das Brainstorming wird üblicherweise in einer Gruppe durchgeführt, sodass nicht wie bei einer Einzelperson das Denken in eine einzige Richtung verläuft, sondern durch die einzelnen Gruppenmitglieder immer neue Anstöße und Richtungen erhält.<sup>62</sup>

Für die Durchführung eines Brainstormings gibt es keine allgemein festgelegten Regeln. Vielmehr wird ein freies und ungehemmtes Aussprechen von Gedanken und Ideen angestrebt, die andere Teilnehmer inspirieren können. Die Vorschläge werden gesammelt und für alle Gruppenmitglieder visuell an Pinnwänden oder Flipcharts dargestellt. Insbesondere bei größeren Gruppen empfiehlt sich der Einsatz eines erfahrenen Moderators, der das Gespräch steuert, die erhobenen Informationen festhält und offene Fragen und weitere Denkrichtungen anmoderiert und weiterverfolgt.<sup>63</sup> Abbildung 5.20 zeigt ein Brainstormingbeispiel zum Thema „Mobile Applikationen“.

<sup>61</sup> Vgl. Meck 2009, S. 24.

<sup>62</sup> Vgl. Meck 2009, S. 30.

<sup>63</sup> Vgl. Vogel-Heuser 2003, S. 49.

Ein weiteres Vorgehen, Informationen zu sammeln, stellt die Methode der Multiperspektive dar. Hierbei bedient man sich der Fähigkeit, sich in andere Personen oder Situationen hineinzuversetzen und das vorliegende Problem somit aus einer anderen Perspektive zu betrachten. Bevor die Methode der Multiperspektive angewendet werden soll, ist es ratsam, zunächst selbst genügend Informationen aus dem eigenen Blickwinkel zu sammeln. Ist man an einem Punkt angekommen an dem das Gefühl entsteht, dass die erhobenen Informationen nicht ausreichen, wählt man in Gedanken eine andere Sicht auf das vorliegende Problem aus und versucht sich vorzustellen, wie diese Person den Sachverhalt sehen würde. Die Verlagerung des Blickwinkels bedeutet jedoch nicht, dass die vorliegende Problemsituation dadurch schneller oder effizienter gelöst wird. Die Methode der Multiperspektive zielt vielmehr darauf ab, der eigenen Informationssuche neue Anstöße zu geben.<sup>64</sup>

Eine der Methode der Multiperspektive ähnelnde Kreativitätstechnik stellt das 6-Hut-Denken dar. Bei dieser Methode geht man davon aus, dass ein Problem aus sechs unterschiedlichen Sichten betrachtet und gelöst werden kann. Zur Durchführung dieser Technik werden sogenannte Denkhüte verwendet, wobei jeder Hut symbolisch für eine andere Denkrichtung steht:<sup>65</sup>

- Der weiße Hut beschäftigt sich neutral und objektiv mit konkreten Fakten.
- Der rote Hut befasst sich mit Emotionen und drückt Gefühle aus.
- Der schwarze Hut ist negativ. Hat man ihn auf, sieht man nur die Schattenseiten jeder Idee.
- Der gelbe Hut gilt als sonnig und positiv; hier stehen Optimismus, Hoffnung und freudige Erwartung im Mittelpunkt.
- Der grüne Hut drückt Natur und Fruchtbarkeit aus, letztlich also auch Kreativität und Innovation.
- Der blaue Hut repräsentiert den blauen Himmel, der über allem steht und die Gesamtkontrolle realisiert.

Durch das gedankliche Aufsetzen eines Hutes werden die zuvor aufgezählten Eigenschaften der jeweiligen Person übertragen. Diese nimmt nun verschiedene Standpunkte zum vorliegenden Problem ein und versucht dadurch, eine neue Sichtweise auf die Situation zu erhalten. Die hierbei resultierenden Aussagen aller Teilnehmer werden gesammelt und als Ganzes für die Problemlösung zusammengeführt.<sup>66</sup> Für die mobile Anwendungsentwicklung ergeben sich in Bezug auf die Ideengenerierung daraus neue Möglichkeiten, innovative Anwendungen zu erheben und eine Erweiterung des Horizonts und der eigenen Denkweise zu ermöglichen.

---

<sup>64</sup> Vgl. Meck 2009, S. 28 f.

<sup>65</sup> Scholz 2007, S. 41.

<sup>66</sup> Vgl. Scholz 2007, S. 41.

Nachfolgend wird eine Auswahl an Techniken für die Bewertung und Dokumentation von Informationen gegeben.

### 5.3.3.2 Techniken der Bewertung und Dokumentation von Informationen

Die zuvor beschriebenen Werkzeuge dienten lediglich der Beschaffung und Gewinnung sowie dem Zusammentragen von Informationen. Damit eine finale Lösungsstrategie aus diesen Informationen abgeleitet werden kann, empfiehlt es sich, diese zunächst zu dokumentieren und in einem abschließenden Schritt zu bewerten. Damit wird gewährleistet, dass die zuvor erhobenen Informationen nicht nur aneinandergereiht, sondern für das weitere Problemlösen nutzbar gemacht werden.

Das Protokoll stellt eine der wesentlichen Techniken zum Festhalten von Informationen dar. Ein Protokoll kann bereits während des Zusammentragens von Informationen geführt werden und bietet dadurch den Vorteil, dass wichtige Informationen nicht vergessen oder Aussagen anders als besprochen formuliert werden. Damit das Protokoll möglichst sachlich und objektiv erstellt wird, bietet es sich an, eine neutrale Person als Protokollantin auszuwählen. Bei größeren Gruppengesprächen besteht bei der Führung eines Protokolls jedoch die Gefahr, dass aufgrund der Fülle an Informationen nicht alle besprochenen Aspekte im Protokoll aufgenommen werden.<sup>67</sup>

Eine weitere Möglichkeit der Ergebnisdokumentation bildet die Erstellung einer Mindmap. Bei einer Mindmap handelt es sich um eine Landkarte der eigenen Gedanken zu einem Thema, die im Gegensatz zum Brainstorming Informationen abstrahiert und miteinander verknüpft. Diese Verknüpfung erfolgt in Form von Ästen und Zweigen:<sup>68</sup>

- Äste bilden Informationen oder Sachverhalte und werden als Oberbegriffe in der Mindmap aufgeführt,
- Zweige werden von Ästen abgeleitet und repräsentieren Informationen, die als Unterbegriffe von den Oberbegriffen abgeleitet werden.

Für das Brainstorming aus Abb. 5.20 ergibt sich folgende Mindmap (siehe Abb. 5.21).

Durch die Erstellung einer Mindmap lassen sich alle relevanten Aspekte zu einem bestimmten Themengebiet bzw. zu einer vorliegenden Problemstellung übersichtlich darstellen. Darüber hinaus besitzt eine Mindmap einen adaptiven Charakter, d. h. sie kann erweitert, angepasst und verändert werden. Grundsätzlich unterstützt eine Mindmap bei der Strukturierung eigener Gedanken und gibt Aufschluss über Zusammenhänge und Überschneidungen der erhobenen Informationen.<sup>69</sup>

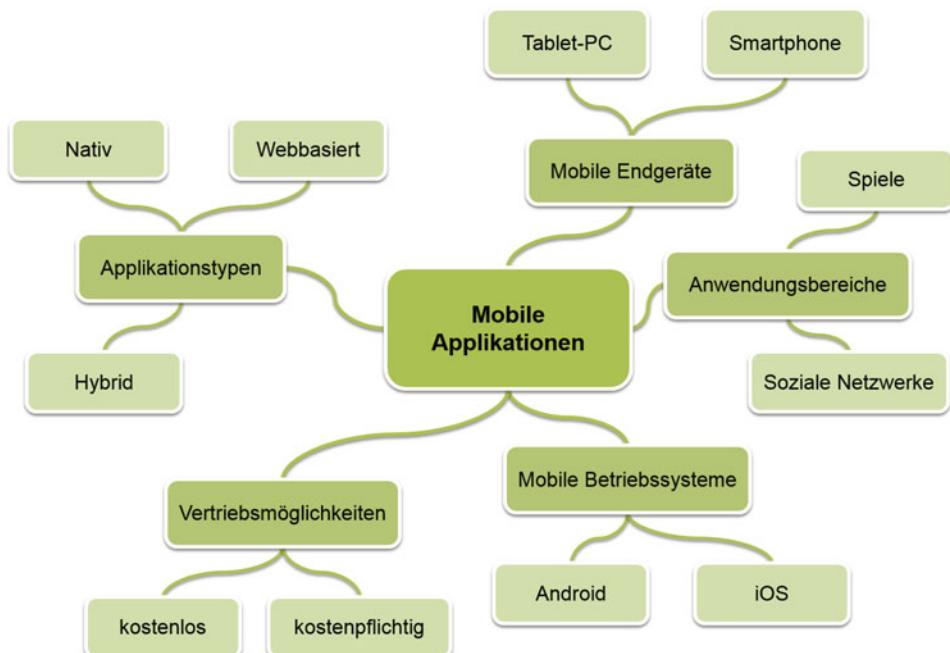
Die Bewertung von Informationen kann qualitativ als auch quantitativ erfolgen. Als Bewertungsraster für vorliegende Informationen bzw. als Anforderungsprofil für notwendige, noch zu beschaffende Informationen, können folgende qualitative Kriterien der

---

<sup>67</sup> Vgl. Meck 2009, S. 31.

<sup>68</sup> Vgl. Meck 2009, S. 32.

<sup>69</sup> Vgl. Meck 2009, S. 33.



**Abb. 5.21** Mindmapbeispiel zum Thema „Mobile Applikationen“

Informationsbewertung verwendet werden: Nützlichkeit, Vollständigkeit, Aktualität und Wahrheit.<sup>70</sup>

Aus der Kenntnis über Handlungsalternativen, spezielle Ereignisse und monetäre Konsequenzen bei der Erhebung von Informationen können ökonomische Bewertungskriterien ermittelt werden.<sup>71</sup> Ziel der Bewertung der Informationen ist es, die eruierten Ideen bzw. Lösungsvorschläge dahin gehend zu analysieren, ob diese weiter verfolgt oder verworfen werden sollen. Für die Bewertung von Informationen haben sich in der Praxis die Grobselektion und die Nutzenanalyse etabliert.

Bei der Grobselektion sollen diejenigen Informationen herausgefiltert werden, die einer genaueren Betrachtung unterzogen werden sollen. Der Fokus liegt hierbei auf besonders guten Informationen oder Ideen für die Umsetzung einer Lösungsstrategie. Schlechte Informationen oder Ideen lassen sich anhand einer der folgenden Aussagen bestimmen:<sup>72</sup>

- Muss-Anforderungen an die mobile Anwendung können nicht erfüllt werden.
- Die Umsetzung des Lösungsvorschlags ist zu teuer gegenüber Kostenvorgaben.
- Die Umsetzung des Lösungsvorschlags dauert zu lange gegenüber Terminvorgaben.

<sup>70</sup> Vgl. Berekoven et al. 2009, S. 24 f.

<sup>71</sup> Vgl. Berekoven et al. 2009, S. 27 f.

<sup>72</sup> Vgl. Dornberger 2006, S. 31.

Zielkriterien	Gewichtung	Mobile Anwendung A		Mobile Anwendung B		Mobile Anwendung C	
		Bewertung	Nutzwert	Bewertung	Nutzwert	Bewertung	Nutzwert
Unterstützte mobile Betriebssysteme	5	4	20	3	15	4	20
Administrationsmöglichkeiten	15	3	45	4	60	2	30
Konfigurationsmöglichkeiten	15	2	30	2	30	3	45
Unternehmensanbindung	20	2	40	4	80	4	80
Usability	10	4	40	2	20	4	40
Datenschutz- und Datensicherheit	20	3	60	4	80	2	40
Herstellersupport	5	1	5	3	15	2	10
Lizenzmodelle	10	2	20	3	30	3	30
Gesamt	100	260		330		295	

**Abb. 5.22** Beispiel für eine Nutzwertanalyse

- Die Umsetzung des Lösungsvorschlags ist aus psychologischen Gründen nicht tragbar.
- Die Umsetzung des Lösungsvorschlags ist aus politischen Gründen nicht durchsetzbar.

Im Allgemeinen liegt im Anschluss an die Grobselektion immer noch eine hohe Variantenvielfalt an guten Ideen zur Lösungsumsetzung vor. Daher bietet sich die Durchführung einer Nutzwertanalyse an die es erlaubt, mehrere Handlungsalternativen gegenüberzustellen und die Alternative mit dem größten Nutzenwert zu bestimmen. Nachfolgend wird der Ablauf einer Nutzwertanalyse beschrieben:<sup>73</sup>

1. Bestimmung der Zielkriterien, die für die Bewertung und Auswahl der Alternativen zugrunde gelegt werden sollen.
2. Gewichtung der einzelnen Zielkriterien, d.h. welches Gewicht soll den zuvor aufgestellten Kriterien beigemessen werden.
3. Bewertung der vorhandenen Handlungsalternativen in Hinblick auf eine mögliche Zielerreichung.
4. Ermittlung der Telnutzenwerte durch die Multiplikation der Zielgewichtungen mit den Zielerreichungsgraden.
5. Ermittlung des Gesamtnutzwertes durch die Addition der einzelnen Telnutzwerte.
6. Vergleich der Gesamtnutzwerte und Auswahl der Handlungsalternative mit dem größten Gesamtnutzwert.

Abbildung 5.22 gibt ein Beispiel für die Durchführung einer Nutzwertanalyse. Hierbei werden verschiedene mobile Anwendungen, die für den Einsatz in einem Unternehmen infrage kommen, einander gegenübergestellt und anhand der Nutzwerte miteinander verglichen. Die Bewertung der vorhandenen Alternativen hinsichtlich der Zielerreichung kann von null (schlecht) bis fünf (sehr gut) erfolgen.

Die Ermittlung der Zielkriterien, die Bewertung der Zielerreichungsgrade als auch die Festlegung der Zielgewichtungen stellen Herausforderungen bei der Durchführung einer Nutzwertanalyse dar. Um Fehler bei der Berechnung der Nutzenwerte zu vermeiden, hat das gesamte Projektteam die Aufgabe die Nutzwertanalyse durchzuführen.<sup>74</sup>

<sup>73</sup> Vgl. Feyhl 2004, S. 95.

<sup>74</sup> Vgl. Aichele 2006, S. 67.

## 5.4 Planungs- und Konzeptionsphase

Im Zuge der Analysephase und Grobplanung besteht das Ziel in der Festlegung des Einsatzbereichs der Softwarelösung und der Definition hierfür notwendiger Aktivitäten und Tätigkeiten. Hierbei spielen die Wechselwirkungen zwischen den Aktivitäten sowie die Entscheidung zur bzw. der Grad der Automatisierung dieser Aktivitäten eine große Rolle. Im Fokus dieser Phase liegt die Realisierung der technischen, personellen, finanziellen und zeitlichen Ressourcen.<sup>75</sup> Zu den wichtigsten Tätigkeiten zählen die Erhebung des Istzustands bzw. der Problemstellung, die Abgrenzung des Problembereichs, die grobe Erfassung und Darstellung der geplanten Systembestandteile, die Betrachtung wirtschaftlicher Projektaspekte sowie die Durchführung einer ersten Aufwandsschätzung. Als Ergebnis der Analyse- und Grobplanungsphase erfolgen die grobe Beschreibung des Istzustands und der Problemstellung, der Projektauftrag, das Lastenheft und ein grober Projektplan.<sup>76</sup>

Je nach Ausgangssituation sind die Gestaltungsmöglichkeiten im weiteren Prozess zu beachten. Während bei Kundenauftragsprojekten die stetige Abstimmung zur Erhebung der Anforderungen im Vordergrund steht, sind diese bei Marketplace-Projekten weitgehend selbst zu definieren und zu erheben, sodass bei einer Eigenentwicklung eigene Prämissen und Ideen im Vordergrund stehen. Die Anwendung der einzelnen Phasen sowie deren Dokumentation sind im Zuge eines strukturierten Vorgehens bei Kundenauftragsprojekten und Marketplace-Projekten dennoch von hoher Bedeutung, um die Strukturierung und Nachvollziehbarkeit solcher Projekte gleichermaßen zu gewährleisten.

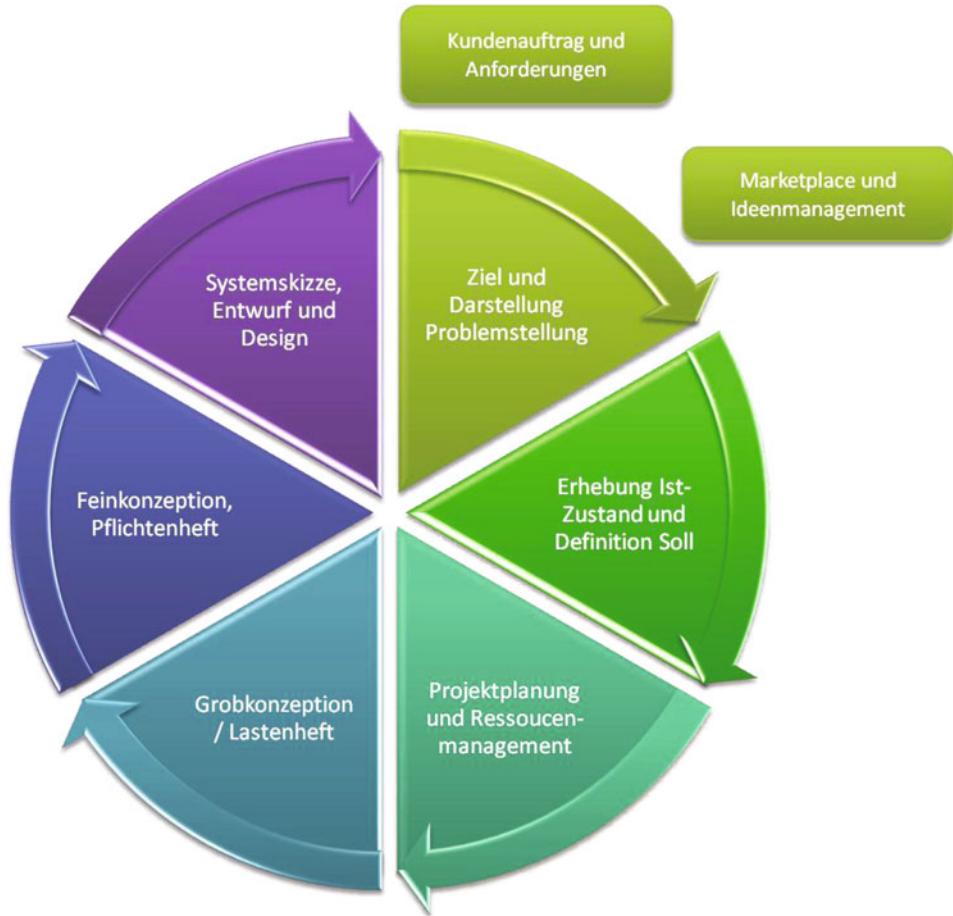
Während das Anforderungsmanagement zur Erhebung und Erarbeitung von funktionalen und nichtfunktionalen Anforderungen betrieben wird, dient die Systemspezifikation zur Erstellung eines Kontraktes zwischen dem Auftraggeber und dem Softwarehersteller. In diesem wird festgelegt, was das neue Softwaresystem leisten soll und welche Bedingungen für die Realisierung gelten. In diesem Zusammenhang ist ebenfalls die Erarbeitung und Festlegung eines Projektplans, die Validierung der Systemspezifikation in Form der Prüfung der Vollständigkeit und Konsistenz der Anforderungen sowie die ökonomische Begründung des Projektes von wesentlicher Bedeutung. Das Ergebnis der Spezifikation in der Planungs- und Konzeptionsphase stellen die Systemspezifikation als Pflichtenheft sowie ein genauer Projektplan dar.<sup>77</sup>

Die Planungs- und Konzeptionsphase stellt den Beginn der Planung und Konzeption der mobilen Anwendungsentwicklung dar. Die Gestaltung dieser Phase hängt im Wesentlichen von dem Ansatzpunkt ab, ob es sich um ein Kundenauftragsprojekt oder ein Marketplace-Projekt handelt. Je nachdem, ob der Entwicklungsansatz die auftragsbezogene Umsetzung für einen bestimmten Kunden oder eine Eigenentwicklung vorsieht, die über den Marketplace vertrieben wird, stehen in dieser Phase unterschiedliche Vorgehens-

<sup>75</sup> Vgl. Pomberger und Pree 2004, S. 12.

<sup>76</sup> Vgl. Pomberger und Pree 2004, S. 12.

<sup>77</sup> Vgl. Rinza 1998, S. 13.



**Abb. 5.23** Vorgehensweise in der Planungs- und Konzeptionsphase

weisen zur Verfügung. Die Vorgehensweise in der Planungs- und Konzeptionsphase wird in Abb. 5.23 verdeutlicht.

#### 5.4.1 Isterhebung und Solldefinition

Technische und fachliche Anforderungen stellen die Grundlage für nahezu alle weiteren Projektaufgaben und Entwicklungsschritte dar und sollten bereits zu Projektbeginn fest definiert werden. Das Erheben und Verwalten von Anforderungen an eine Software bilden den zentralen Schwerpunkt des Anforderungsmanagements. Als Haupttätigkeiten können in diesem Zusammenhang die Ermittlung, Dokumentation, Abstimmung und Verwaltung von Anforderungen genannt werden. Ziel ist es, alle notwendigen Anforderungen für die

Entwicklung eines Produktes festzustellen, um ein Produkt oder eine Software effizient und möglichst fehlerfrei entwickeln zu können.<sup>78</sup>

Anforderungen beschreiben Funktionalitäten und Eigenschaften einer Software, die entweder aus Kundenwünschen oder gegebenen Aufgabenstellungen abgeleitet werden. Aufgabe der Software- bzw. Systementwickler ist, die festgelegten Anforderungen zu verstehen und entsprechend umzusetzen, damit die Kunden des Produktes zufriedengestellt oder Problemlösungen umgesetzt werden können.<sup>79</sup> Pohl definiert den Begriff Anforderung wie folgt:<sup>80</sup>

► **Anforderung:**

1. Eine Bedingung oder Eigenschaft, die ein System oder eine Person benötigt um ein Problem zu lösen oder ein Ziel zu erreichen.
2. Eine Bedingung oder Eigenschaft, die ein System oder eine Systemkomponente aufweisen muss, um einen Vertrag zu erfüllen oder einem Standard, einer Spezifikation oder einem anderen formell auferlegten Dokument zu genügen.
3. Eine dokumentierte Repräsentation einer Bedingung oder Eigenschaft, wie in (1) oder (2) definiert.

In der Softwareentwicklung werden unterschiedliche Typen von Anforderungen differenziert. Anforderungen an ein Softwaresystem werden in funktionale und nichtfunktionale Anforderungen unterschieden. Funktionale Anforderungen beschreiben das Verhalten sowie die Funktionen der Software. Ergebnis der Erhebung dieser Anforderungen bildet die Abgrenzung des Leistungsumfangs einer Software. Im Gegensatz hierzu beschreiben nichtfunktionale Anforderungen Möglichkeiten zur Realisierung funktionaler Anforderungen und geben hierfür feste Rahmenbedingungen vor, bspw. Benutzbarkeit, Zuverlässigkeit oder Wartbarkeit. Eine Unterteilung in Entwickler- und Anwendersicht dient der besseren Strukturierung funktionaler und nichtfunktionaler Anforderungen.<sup>81</sup> Abbildung 5.24 stellt die unterschiedlichen Anforderungstypen grafisch dar.

#### **5.4.2 Projektplanung und Ressourcenmanagement**

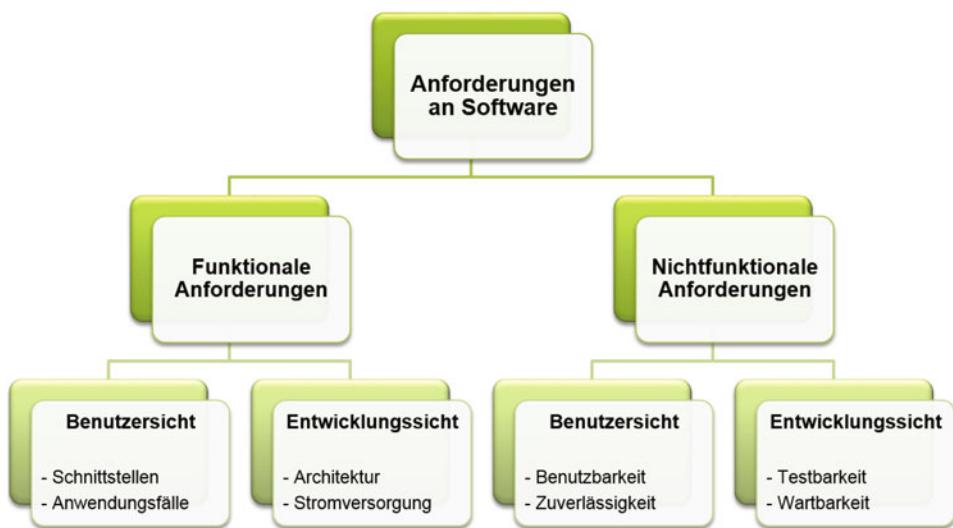
Auf der Basis der vorhergehenden Anforderungsentwicklung ist innerhalb der Planungs- und Konzeptionsphase die Erstellung eines Projektplans in Verbindung mit dem Ressourcenmanagement zur Entwicklung von mobilen Anwendungen vorzunehmen, um den

<sup>78</sup> Vgl. Grande 2011, S. 11.

<sup>79</sup> Vgl. Krcmar 2010, S. 164 f.

<sup>80</sup> Vgl. Pohl 2008, S. 13.

<sup>81</sup> Vgl. Krcmar 2010, S. 164 f.



**Abb. 5.24** Unterschiedliche Anforderungstypen. (Eigene Erstellung, in Anlehnung an Krcmar 2010, S. 165)

Projektablauf in zeitlicher und ressourcenbezogener Hinsicht abzubilden. Von hoher Bedeutung ist hierbei die Bereitstellung und Einplanung von personellen und sachlichen Ressourcen, wie z. B. der technischen Ausstattung.

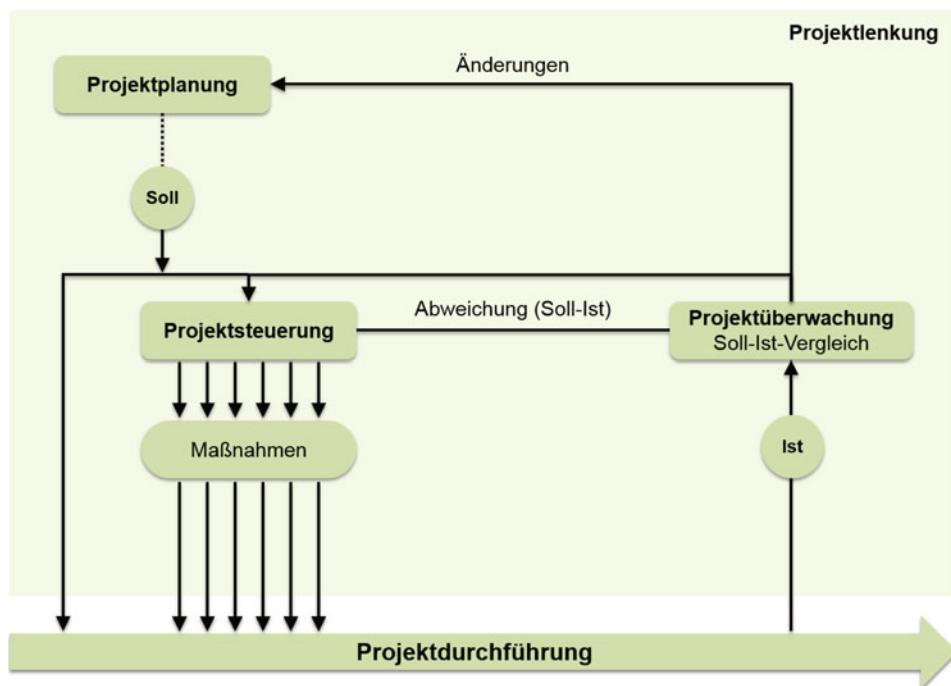
Im Wesentlichen besteht die Projektplanung aus einer statisch strukturellen und einer dynamisch ablauforientierten Planung sowie einer initialisierenden Projektorganisation.<sup>82</sup> Die Projektplanung beginnt mit der Erhebung aller mittelbaren und unmittelbaren Projektaktivitäten, die zur Erreichung des Projektziels erforderlich sind. Platz und Schmelzer definieren den Begriff Projektplan wie folgt:<sup>83</sup>

► **Projektplanung** meint die systematische Informationsgewinnung über den zukünftigen Ablauf eines einmaligen Vorhabens samt gedanklicher Vorwegnahme des für das Erreichen seiner Ziele notwendigen Handelns.

Durch das Aufstellen eines Projektplans werden mehrere Ziele verfolgt: Einerseits sollen realistische Sollvorgaben hinsichtlich der zu erbringenden Arbeitsleistung sowie deren Termine ermittelt und andererseits der Ressourceneinsatz und zulässige Kosten kalkuliert werden. Darüber ermöglicht die Aufstellung eines Projektplans die Untergliederung des Gesamtprojekts in Teilprojekte und Arbeitspakete, sodass einzelne Aktivitäten bes-

<sup>82</sup> Vgl. Aichele 2006, S. 74.

<sup>83</sup> Platz und Schmelzer 1986, S. 131.



**Abb. 5.25** Einbettung der Projektplanung in den Projektablauf. (Eigene Erstellung, in Anlehnung an Litke 2007, S. 84)

ser geplant werden können.<sup>84</sup> Abbildung 5.25 zeigt die Projektplanung als Teil der Projektlenkung, die im Weiteren die Projektüberwachung und -steuerung umfasst.

Der in Abb. 5.25 dargestellte Projektablauf lässt sich wie folgt interpretieren:<sup>85</sup>

- Die Projektplanung erarbeitet Sollvorgaben für die Projektdurchführung.
- Die Projektüberwachung führt daraufhin einen Soll-Ist-Vergleich durch und meldet Abweichungen an die Projektsteuerung.
- In Verbindung mit der Projektüberwachung erarbeitet und leitet die Projektsteuerung Maßnahmen ein, um Abweichungen in der Projektdurchführung zu korrigieren.
- Sollten die der Projektdurchführung zur Verfügung stehenden Regelmechanismen nicht ausreichen, werden von der Projektplanung entsprechende Änderungen vorgeschlagen.

<sup>84</sup> Vgl. Litke 2007, S. 83.

<sup>85</sup> Vgl. Litke 2007, S. 83.

Die Frage nach dem wer bzw. was zur Umsetzung des Entwicklungsprojekts benötigt wird, stellt sich auf Grundlage der Aktivitäten- und Terminplanung. Im Fokus der Ressourcenplanung liegen hierbei folgende Einheiten:<sup>86</sup>

- Personal des eigenen oder eines anderen Unternehmens,
- Einrichtungen, also Standorte und Räumlichkeiten, zur Durchführung der Projektarbeit,
- Sachmittel und sonstige Projektausstattung, wie z. B. Flipcharts, Beamer, PCs,
- Schulungskosten und Reisekosten.

Die Zuweisung der Ressource „Mitarbeiter“ zu einer Aktivität sollte unter Berücksichtigung des Wissensstandes über Inhalte und Umfang der geplanten Aktivitäten der disponierenden Person erfolgen. Besondere Herausforderungen für das Ressourcenmanagement ergeben sich bei der Auswahl geeigneter Ressourcen. Für eine bestmögliche Auswahl sollten daher die drei Faktoren Ressourcen, Zeit und Kosten optimal aufeinander abgestimmt sein. Generell gilt hierbei, dass Ressourcen des eigenen Unternehmens wichtige Ressourcen mit einer geringen zeitlichen Verfügbarkeit sind und Ressourcen externer Unternehmen kostenintensivere Lösungen darstellen.<sup>87</sup>

Die erfolgreiche Durchführung des Entwicklungsvorhabens setzt voraus, dass während der Laufzeit eines Projektes das Projektteam personell nicht verändert wird, da dies zu einem hohen Einarbeitungsaufwand für bestehende und neue Ressourcen führen würde. Des Weiteren sollte die Auswahl der Ressourcen des eigenen Unternehmens mit der jeweiligen Bereichsleitung des Fachbereichs abgestimmt und koordiniert sowie die Konsequenzen der Freistellung der Ressource deutlich gemacht werden:<sup>88</sup>

- Die Ressource steht ganz oder nur teilweise für die Tagesarbeit zur Verfügung.
- Die Ressource arbeitet nicht nur während der Projektsitzungen, sondern auch in der Abteilung an den Aufgaben des Projektes.
- Dauerhafte Doppelbelastungen aus Linien- und Projektarbeit sind auszuschließen.

Die bereits in Abschn. 5.3.3.2 vorgestellte Nutzwertanalyse bietet sich zur Berechnung und Kalkulation der in Verbindung mit der Freistellung von Ressourcen anfallenden Kosten an. Die Nutzenanalyse und die projektbezogene Kapitalwertmethode stellen weitere Methoden zur Ermittlung der Wirtschaftlichkeit von Projekten dar, welche nachfolgend nicht weiter betrachtet werden.<sup>89</sup>

---

<sup>86</sup> Vgl. Aichele 2006, S. 132.

<sup>87</sup> Vgl. Aichele 2006, S. 133.

<sup>88</sup> Vgl. Aichele 2006, S. 133.

<sup>89</sup> Vgl. Aichele 2006, S. 64 ff.

**Abb. 5.26** Ausgewählte Aspekte des Lastenheftes



### 5.4.3 Lasten- und Pflichtenheft

Das Lastenheft stellt die Zusammenstellung aller Anforderungen aus Sicht des Auftraggebers, wie z. B. Kunden und Anwender, an ein Projekt bzw. an ein Zielsystem oder ein zu entwickelndes Produkt dar. Das Lastenheft enthält alle relevanten Randbedingungen und wird in der Regel von dem Auftraggeber formuliert. Das Deutsche Institut für Normung definiert nach der DIN 69905 den Begriff Lastenheft als „Gesamtheit der Forderungen an die Lieferungen und Leistungen eines Auftragnehmers“. Wesentlicher Bestandteil des Lastenhefts ist somit die Beschreibung der Anforderungen des Auftraggebers (vgl. Abb. 5.26).<sup>90</sup>

Das Lastenheft beinhaltet die Ergebnisse der Grobplanung und ist somit eng mit der eigentlichen Grobkonzeption verbunden. Neben der Definition des Projektziels sind die Anforderungsentwicklung und das -management von Projekten von wesentlicher Bedeutung, um die Grobdarstellung der technischen Aspekte im Projekt vorzunehmen und die einzusetzenden Ressourcen und die Finanzkalkulation darzustellen. Des Weiteren kann im Zuge des Projektmanagements, aufgrund der groben Spezifikationen im Lastenheft, eine terminliche Projektplanung erfolgen.

Das Pflichtenheft stellt aus der Sicht des Auftragnehmers die formelle sowie auch detaillierte Antwort auf die Anforderungen des Auftraggebers dar, die zuvor im Lastenheft beschrieben wurden. Die zu erbringenden Ergebnisse des Auftragnehmers werden hierbei in erforderliche Tätigkeiten (Pflichten) umgesetzt. Im Zuge der DIN 69901 wird das Pflichtenheft als ausführliche Beschreibung der Leistungen, wie z. B. technische, wirtschaftliche und organisatorische Leistungen, beschrieben, die zur Erreichung der Projektziele notwendig sind. Gemäß der DIN 69901 sind im Pflichtenheft die vom Auftragnehmer erarbeiteten Realisierungsvorgaben niedergelegt.<sup>91</sup>

<sup>90</sup> Vgl. Stolle und Herrmann 2006, S. 80.

<sup>91</sup> Vgl. Stolle und Herrmann 2006, S. 81.

In Hinblick auf die Entwicklung von kundenindividuellen mobilen Applikationen spielt die Erstellung des Pflichtenhefts eine wesentliche Rolle zur beiderseitigen Festlegung des Realisierungsvorschlags sowie der hierfür notwendigen Leistungen zur Umsetzung der Anforderungen. Im Pflichtenheft erfolgt die verbindliche Aussage und die Realisierbarkeit sowie die bzgl. der Umsetzung benötigten Aufwände zur Erreichung des Projektziels. Hierfür werden die Anforderungen aus dem Lastenheft bzw. der vorangegangenen Anforderungserhebung aufgegriffen, konkretisiert und ggf. angepasst. Diese Detaillierung der Anforderungen erfolgt bereits in Hinblick auf die konkrete Ausgestaltung der Realisierung und wie diese im Konkreten umgesetzt wird.<sup>92</sup> Bei der Entwicklung von Marketplace-Projekten bildet das Ergebnis der Planungs- und Konzeptionsphase die Auffertigung eines Feinkonzeptes, welches die Anforderungen, Ideen und Entwurfsvorschläge strukturiert dokumentiert.

Im Gegenzug zu den beiden durch DIN-Normen eindeutig definierten Begriffen „Lastenheft“ und „Pflichtenheft“ werden die beiden Begriffe „Leistungsbeschreibung“ und „Konzept“ unterschiedlich verwendet und interpretiert. Die Leistungsbeschreibung definiert im Detail, welche Leistungen durch den Auftragnehmer zu erbringen sind, und ist somit ein zentraler Bestandteil eines Angebots oder eines Vertrages. Darüber hinaus existieren bei der Leistungsbeschreibung, anders als bei anderen Angebotsbestandteilen, wie z.B. die AGB, keine unmittelbaren rechtlichen Bestimmungen in Hinblick auf die Form und Gestaltung. Eine vollständige, fehlerfreie und eindeutige Leistungsbeschreibung ermöglicht eine erfolgreiche Projektabwicklung und geht aus einer vorhergehenden Konzeptionsphase hervor.<sup>93</sup> Der Begriff des Konzeptes wird mitunter synonym zum Begriff der Leistungsbeschreibung verwendet, kann jedoch spezielle Besonderheiten in Bezug auf den geplanten Projektlauf beinhalten. Je nach dem Grad der Detaillierung des Konzeptes wird zwischen den beiden Begriffen eines Grob- bzw. eines Feinkonzeptes unterschieden. In Verbindung mit dem Begriff der Leistungsbeschreibung ist ein Grobkonzept als vorläufige Leistungsbeschreibung zu sehen. Die Erstellung eines Grobkonzepts findet dann Anwendung, wenn die Anforderungen des Auftraggebers unklar oder unvollständig sind.<sup>94</sup>

#### **5.4.4 Die Entwurfsphase**

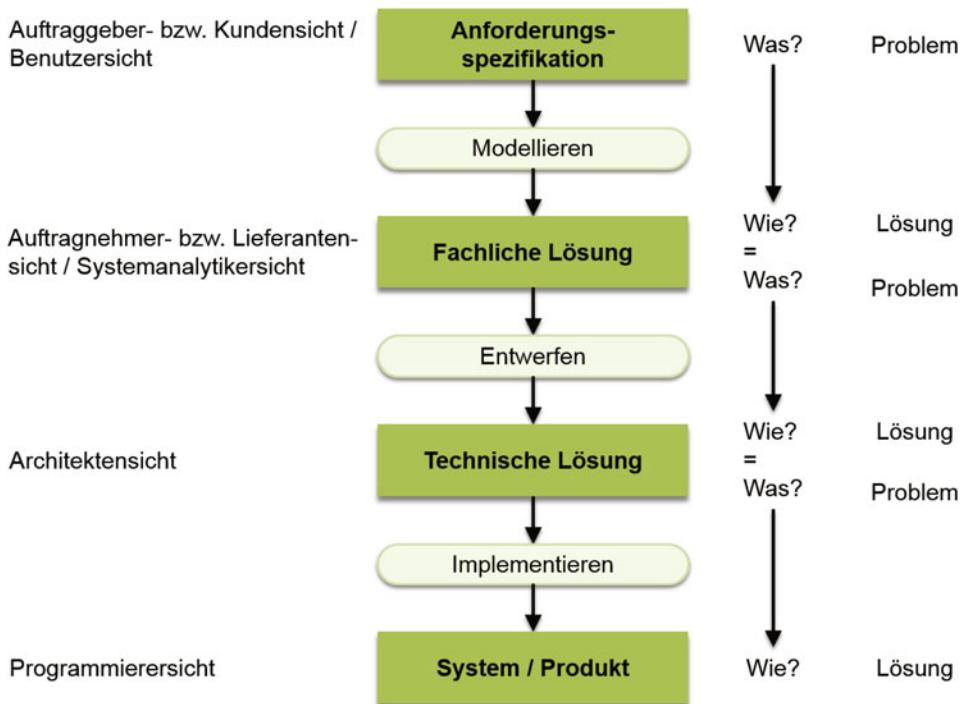
Die Planung- und Konzeptionsphase beinhalten den Entwurf der mobilen Applikationen auf der Grundlage der vorangegangenen Phasen im Vorgehensmodell. Die Ergebnisse aus der Markt- und Problemanalyse sowie die Anforderungen an die zu entwickelnde Applikation finden in diesem Schritt entsprechende Betrachtung. In der Entwurfsphase besteht die primäre Aufgabe darin, eine softwaretechnische Lösung, im Sinne einer Softwarearchitektur, zu entwickeln. Problematisch hierbei ist, dass innerhalb der Phase des

---

<sup>92</sup> Vgl. Stolle und Herrmann 2006, S. 81.

<sup>93</sup> Vgl. Stolle und Herrmann 2006, S. 82.

<sup>94</sup> Vgl. Stolle und Herrmann 2006, S. 82.



**Abb. 5.27** Beispiel für den Prozess eines Softwareentwurfs. (Eigene Erstellung, in Anlehnung an Balzert 2011, S. 481)

Softwareentwurfs eine Vielzahl von Einflussfaktoren berücksichtigt werden muss, die sich gegenseitig bedingen. Der Entwurf erfolgt häufig in zwei Schritten:<sup>95</sup>

- Grobentwurf, der Festlegung der umfassenden Softwarearchitektur, „Entwurf im Großen“
- Feinentwurf, dem Entwerfen der einzelnen Subsysteme und Komponenten im Detail, „Entwurf im Kleinen“

Je nach Art des einzusetzenden Vorgehensmodells (vgl. Abschn. 5.2) kann der Entwurfsprozess abweichen. In nachfolgender Abbildung erfolgt die beispielhafte Darstellung eines Entwurfsprozesses (vgl. Abb. 5.27).

Auf Grundlage der vorangegangenen Anforderungsspezifikation erfolgt im Entwurfsprozess die Erstellung der fachlichen Lösung. Die fachliche Lösung beinhaltet, je nach Vorgehen, das Pflichtenheft, ein (z. B. objektorientiertes) Analysemodell, das Konzept sowie Prototypen oder Abbildungen von bereits erstellten Benutzeroberflächen.<sup>96</sup>

<sup>95</sup> Vgl. Balzert 2011, S. 6.

<sup>96</sup> Vgl. Balzert 2011, S. 481 f.

Auf dieser Ebene erfolgt die Beschreibung der fachlichen Prozessgestaltung und des Anwendungskontextes. Als Werkzeuge zur Erstellung der fachlichen Lösung können Modellierungsmethoden (vgl. Abschn. 5.4.5), Mock-up-Screens und Prototypen genutzt werden.

Das Ergebnis des Entwurfs ist die technische Lösung, welche die vorhergehende fachliche Lösung aufgreift. Für den Entwurf der technischen Lösung kann mitunter der Ansatz des System- und Komponentenentwurfs betrachtet werden. Dieser dient der Festlegung, welche Systemspezifikationen und welche hiermit verbundenen Anforderungen durch welche Systemkomponenten abgedeckt werden und wie diese Komponenten miteinander agieren sollen. Als wichtigste Tätigkeiten zählen hierzu der Entwurf der Systemarchitektur durch die Definition der Systemkomponenten sowie deren Wechselwirkung. Weiterhin sind der Entwurf des logischen Datenmodells sowie der Entwurf der algorithmischen Struktur einzelner Komponenten und deren Verarbeitungsprozeduren von hoher Bedeutung. Die Validierung der Algorithmen und Komponenten dient zur Überprüfung des Entwurfs in Hinblick auf die Anforderungsrealisierung. Als Ergebnis der Entwurfsphase sind die Beschreibung des logischen Datenmodells, die Systemarchitektur, die algorithmische Struktur der Komponenten sowie die Dokumentation der Entwurfsentscheidungen zu nennen.<sup>97</sup>

Der fachliche und technische Entwurf bildet die Ausgangsposition für die nachfolgende Implementierung. Der Entwurfsprozess kann mitunter Auswirkungen auf vorhergehende Phasen erzeugen, wie z. B. auf die Detaillierung und ggf. Anpassung von Anforderungen, sowie auf die Planung der softwaretechnischen und hardwaretechnischen Umsetzung der Lösung.

#### **5.4.5 Werkzeuge zur Planungs- und Konzeptionsphase**

Business Reengineering, Geschäftsprozessoptimierung sowie Lean Management sind Begriffe, die in den letzten Jahren verstärkt im Zusammenhang mit Rationalisierungs- und Modernisierungsprojekten in der Industrie genannt wurden. Ziele dieser Projekte beruhen auf der Optimierung der organisationsinternen Prozesse, Reduktion der Kosten sowie der Stärkung der Marktposition und damit auch der Maximierung des Umsatzes und Gewinns. Zur Erreichung dieser Ziele erfolgt im Zuge der Auswahl und Anwendung geeigneter Optimierungskonzepte der Einsatz unterschiedlicher Modellierungsmethoden zur Abbildung der betriebswirtschaftlichen Zusammenhänge.<sup>98</sup>

Modellierungsmethoden ermöglichen die problembezogene, grafische Darstellung der unternehmerischen Realität in Form von Modellen. Modelle sind zugänglicher, leichter manipulierbar und kostengünstiger als das Original und eignen sich somit besonders für die vereinfachte Abbildung von Unternehmensstrukturen. Im Laufe der Zeit hat sich

---

<sup>97</sup> Vgl. Rinza 1998, S. 13.

<sup>98</sup> Vgl. Aichele 2006, S. 227 f.

aufgrund unterschiedlicher betriebswirtschaftlicher Problemstellungen und Geschäftsprozesse eine Vielzahl an Modellierungsmethoden differenziert. Hierzu zählen Modelle zur Darstellung von Unternehmens-, Daten- oder Prozessstrukturen.<sup>99</sup> Der Entwurf von Daten- und Prozessmodellen spielt insbesondere in den frühen Entwicklungsphasen der Softwareentwicklung eine bedeutende Rolle, bspw. bei der Modellierung von Softwarestrukturen zur Unterstützung der Komplexitätsbeherrschung.<sup>100</sup> Weitere Anwendung findet die Modellierung innerhalb der Beschreibungs- und Entwurfsphase von Softwareprojekten zur Darstellung von Funktionsstrukturen, Programmabläufen oder typischen Anwendungsfällen.

In den nachfolgenden Abschnitten werden etablierte Modellierungsmethoden zur grafischen Darstellung von Daten- und Prozessstrukturen innerhalb der Anwendungsentwicklung vorgestellt sowie die Notation dieser Modellierungsformen erläutert.

#### 5.4.5.1 Methoden zur Programmablaufmodellierung

Der innerhalb der Planungs- und Konzeptphase spezifizierte System- bzw. Softwareentwurf muss für die spätere Umsetzung in einen Programmumentwurf umgesetzt werden. Hierbei werden hauptsächlich die programmspezifischen Aufgaben und Funktionen in Form von Programmablaufplänen oder Struktogrammen modelliert. Diese Modelle können in der nachfolgenden Entwicklungsphase in die ausgewählte Programmiersprache übertragen werden. Zur Darstellung von Programmfunctionen und -abläufen werden nachfolgend die zuvor genannten grafischen Ansätze vorgestellt. In der Praxis haben sich ebenfalls auch textuelle Darstellungsformen etabliert, wie z. B. Pseudocode, die im Folgenden nicht weiter betrachtet werden.

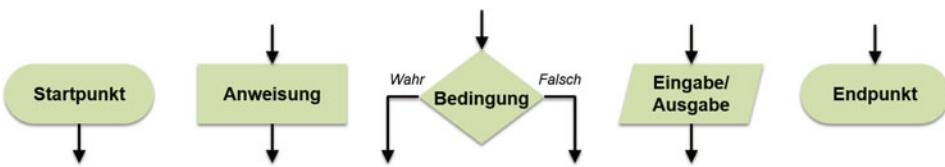
Bereits zu Beginn der 60er-Jahre wurde versucht, die Ablaufstruktur für einen Algorithmus grafisch so darzustellen, dass die Reihenfolge der Ausführung der einzelnen Programmaktionen auf einen Blick erfasst werden kann. Hierfür wurden einfache grafische Symbole wie Linien, Kreise oder Rechtecke verwendet, um die Struktur eines Algorithmus möglichst detailliert darzustellen.<sup>101</sup> Ergebnis bildet ein sogenannter Programmablaufplan (PAP), der in einem weiteren Schritt Befehl für Befehl in ein Programm umgesetzt wird. Der PAP wurde im Laufe der Zeit optimiert und ist mit seinem Symbolvorrat unter der DIN 66001 genormt. Abbildung 5.28 zeigt eine Auswahl an Symbolen zur Erstellung eines PAP nach der DIN 66001.

Abbildung 5.29 zeigt ein Beispiel für einen PAP, welches die Berechnung des arithmetischen Mittels grafisch darstellt. Wie aus dem Beispiel ersichtlich, liegt der Vorteil dieser Darstellungsform darin, Programmfunctionen schnell und einfach abzubilden und dem Leser dadurch einen Überblick über deren Komplexität zu verschaffen. Die Anwendung

<sup>99</sup> Vgl. Aichele 2006, S. 228.

<sup>100</sup> Vgl. Diederichs 2004, S. 114 f.

<sup>101</sup> Vgl. Pomberger und Dobler 2008, S. 69.



**Abb. 5.28** Ausgewählte Symbole für Programmablaufpläne nach DIN 66001. (Eigene Erstellung, in Anlehnung an Pomberger und Dobler 2008, S. 69)

eines PAP ist insbesondere bei kleinen, linearen Programmen sinnvoll. Bei komplexeren Programmabläufen oder -funktionen zeigen sich verschiedene Nachteile:<sup>102</sup>

- Der PAP kann zahlreiche Programmverzweigungen sowie Vorwärts- und Rückwärts-sprünge enthalten.
- Der PAP kann ggf. nicht mehr in einzelne Segmente zerlegt werden, sodass eine zeitgleiche Bearbeitung verschiedener Programmteile nicht möglich ist.
- Der PAP kann aufgrund unübersichtlicher Schleifenbildung schlecht verändert oder erweitert werden.

Struktogramme stellen eine ebenfalls genormte Form der grafischen Darstellungsmethoden von Programmabläufen dar. Sie haben ihren Ursprung in der strukturierten Programmierung, welche durch den holländischen Mathematiker Dijkstra ausgelöst und weiterentwickelt wurde. Im Gegensatz zur linearen Programmierung verfolgt die strukturierte Programmierung einen Top-down-Ansatz, der das gesamte Programm bis auf die Ebene in weitgehend voneinander unabhängige Systembausteine zerlegt. Für die grafische Darstellung dieser Systembausteine bzw. Strukturblöcke haben 1973 Ike Nassi und Ben Shneidermann die bereits erwähnten Struktogramme (auch Nassi-Shneidermann-Diagramme genannt) vorgeschlagen, die seit 1985 durch die DIN 66261 genormt sind.<sup>103</sup>

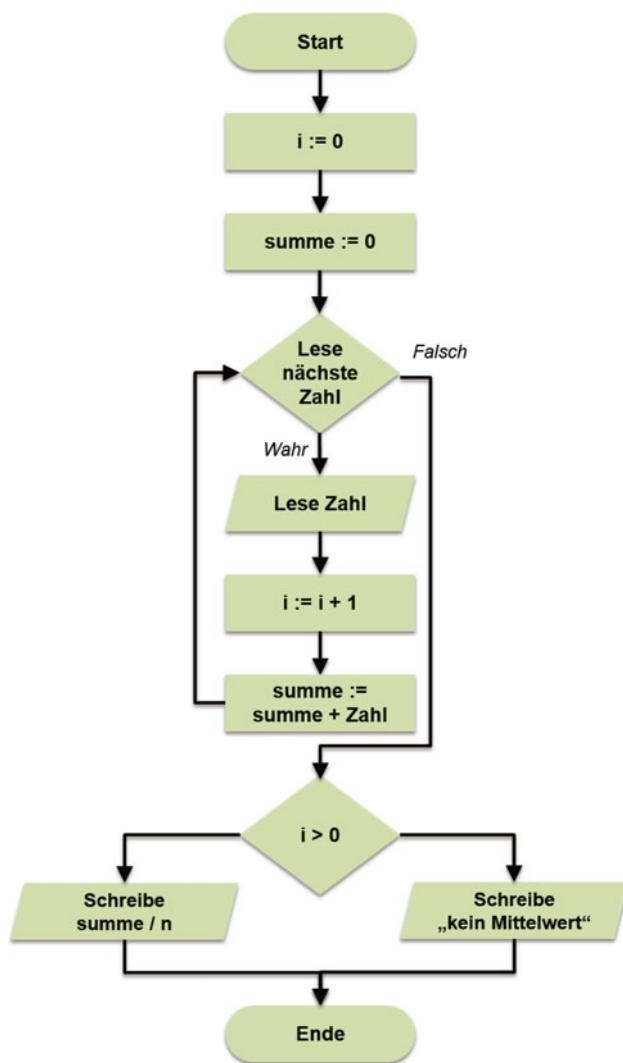
Abbildung 5.30 zeigt eine Auswahl an Symbolen für die Erstellung von Struktogrammen nach der DIN 66261. Bei der Erstellung eines Struktogramms müssen die einzelnen Strukturblöcke so miteinander verknüpft werden, dass die Unterkante des vorhergehenden Strukturblocks mit der Oberkante des nachfolgenden Blocks zusammenfällt. Das hierbei entstehende Konstrukt ergibt eine in sich geschlossene Einheit mit genau einem Eingang bzw. Startpunkt (Oberkante des ersten Strukturblocks) und einem Ausgang bzw. Endpunkt (Unterkante des letzten Strukturblocks). Abbildung 5.31 stellt den Algorithmus zur Ermittlung des Mittelwerts als Struktogramm dar.

Im Gegensatz zu Programmablaufplänen weisen Struktogramme den Nachteil auf, dass beliebige Sprünge zu anderen Strukturblöcken nicht abgebildet werden können. Des Weiteren sind Struktogramme aufwendig zu zeichnen und schwer zu ändern bzw. zu modifizieren.

<sup>102</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 265.

<sup>103</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 266.

**Abb. 5.29** PAP zur Berechnung des arithmetischen Mittels. (Eigene Erstellung, in Anlehnung an Pomberger und Dobler 2008, S. 70)



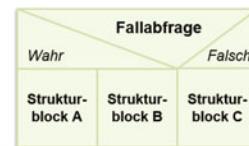
Reihung (Folge)



Verzweigung



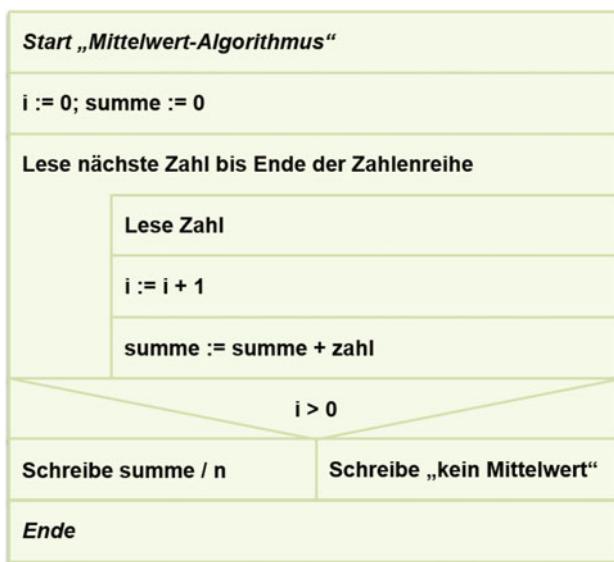
Wiederholung



Fallunterscheidung

**Abb. 5.30** Ausgewählte Symbole für Struktogramme nach DIN 66261. (Eigene Erstellung, in Anlehnung an Stahlknecht und Hasenkamp 2005, S. 268)

**Abb. 5.31** Struktogramm zur Berechnung des arithmetischen Mittels



#### 5.4.5.2 Methoden zur Datenmodellierung

Die Durchführung der Datenmodellierung ist methodisch besonders anspruchsvoll als auch komplex und zielt auf die möglichst exakte Abbildung realer Datenbestände ab. Während bspw. für die Darstellung einer Funktion lediglich die Funktion selbst betrachtet wird, werden für die Strukturierung von Daten vielfältige Begriffe wie Entitytypen, Beziehungstypen und Attribute verwendet. Als am weitesten verbreitete Entwurfsmethode für Datenstrukturen gilt das Entity-Relationship-Modell (ERM) von Chen, das im Folgenden dargestellt und erläutert wird.<sup>104</sup> Das ERM-Modell soll anschließend durch ein Beispiel verdeutlicht werden.

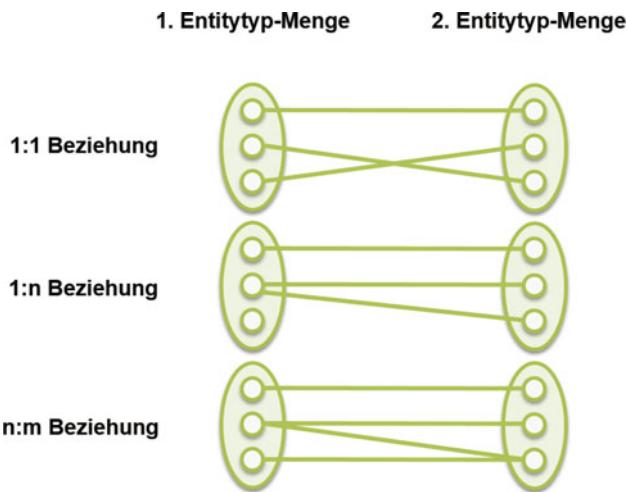
Das ERM gehört zur Gruppe der semantischen Datenmodelle und ist durch eine klare Definition sowie einheitliche grafische Darstellungsobjekte gekennzeichnet. Grundgedanke des ERM ist die Darstellung von Objekten (Entities) sowie deren Beziehungen (Relationships) zueinander.<sup>105</sup> Entities bilden alle realen oder abstrakten Dinge, Objekte und Ereignisse der realen Welt ab, welche sich durch unterschiedliche Eigenschaften eindeutig beschreiben lassen. Typischerweise werden innerhalb eines ERM gleichartige Entities zu einer einzigen Menge zusammengefasst und als Entitytyp bezeichnet. Innerhalb eines Geschäftsprozesses kann ein Entitytyp bspw. einen Kunden-, Lieferanten- oder Artikelstamm darstellen. Entitytypen werden im ERM als Rechteck modelliert.<sup>106</sup> Eine Beziehung ist eine logische Verknüpfung zwischen mindestens zwei Entitäten. Beziehungstypen werden im ERM durch Rauten dargestellt und sind mit den ihnen zugeordneten

<sup>104</sup> Vgl. Scheer 1997, S. 31.

<sup>105</sup> Vgl. Mertens et al. 2005, S. 62.

<sup>106</sup> Vgl. Scheer 1997, S. 31.

**Abb. 5.32** Arten von Beziehungstypen im ERM.  
(Eigene Erstellung, in Anlehnung an Scheer 1997, S. 34)



Entitytypen verbunden. Anhand des Komplexitätsgrades einer Beziehung lassen sich drei unterschiedliche Beziehungstypen unterscheiden: 1:1-, 1:n-, n:m-Beziehungen (vgl. Abb. 5.32). Eine 1:1-Beziehung bringt zum Ausdruck, dass zu jedem Element der ersten Menge maximal ein Element der zweiten Menge zugeordnet ist. Bei einer 1:n-Beziehung kann ein Entity der ersten Menge genau einem oder mehreren Entities aus der zweiten Menge zugeordnet werden. Bei der n:m-Beziehung steht jedes Element der ersten Menge mit einem oder mehreren Elementen der zweiten Menge in Beziehung.<sup>107</sup> Eine mögliche Konditionalität (keine Zuordnung eines Elements) kann durch die Angabe eines c1 (oder c), cn, cm ausgedrückt werden.

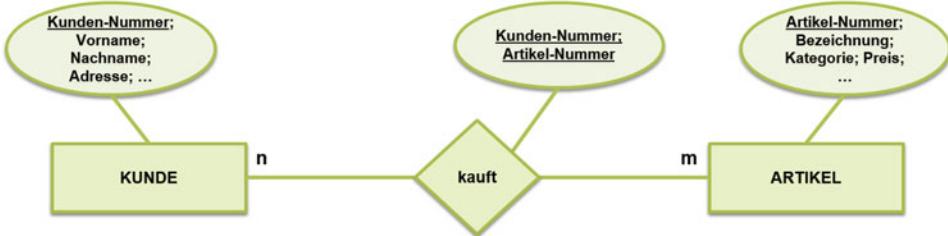
Attribute sind Eigenschaften von Entity- und Beziehungstypen und werden im ERM durch Kreise repräsentiert. Attribute enthalten eine konkrete Ausprägung zur exakten Klassifizierung von Entities und Beziehungen, bspw. den Namen eines Kunden oder die Artikelnummer. Zur Abgrenzung sowie zur Definition einzelner Entitytypen gegenüber weiteren im ERM enthaltenen Entities müssen eindeutige Schlüsselattribute bestimmt werden. Diese werden im ERM durch Unterstreichen der jeweiligen Ausprägung gekennzeichnet.<sup>108</sup> Abbildung 5.33 stellt das Konzept der ERM-Modellierung nochmals grafisch anhand eines ausgewählten Beispiels dar.

Das Beispiel aus Abb. 5.33 stellt die Beziehung zwischen einem Kundenstamm und einem Artikelstamm dar. Die Beziehung „kauft“ signalisiert den Zusammenhang zwischen Kunden und Artikeln. Durch die Angabe der n:m-Beziehung kann das ERM wie folgt interpretiert werden:

- Ein Kunde kauft i. d. R. einen oder mehrere Artikel.
- Ein Artikel wird i. d. R. von einem oder mehreren Kunden gekauft.

<sup>107</sup> Vgl. Scheer 1997, S. 33 f.; Mertens et al. 2005, S. 63.

<sup>108</sup> Vgl. Scheer 1997, S. 33.



**Abb. 5.33** Beispiel eines ER-Modells

Der Beziehungstyp „kauf“ wird weiterhin durch die Schlüsselattribute der Entitytypen „Kunde“ und „Artikel“ genau bestimmt und weist selbst keine weiteren Eigenschaften auf.

Die Modellierung von Datenstrukturen mittels ER-Modellen hat sich insbesondere bei der Konzeption und Erstellung von Datenbanken sowie Datenbanksystemen etabliert. Weiterhin werden ER-Modelle zur Modellierung von Datenstrukturen innerhalb von Softwareanwendungen als auch für den Softwareentwurf eingesetzt. Nachteile der Datenmodellierung bestehen in der fehlenden Darstellung exakter Geschäftsprozess- oder Programmabläufe. Hierfür müssen Prozessmodelle eingesetzt werden, die nachfolgend näher erläutert und beschrieben werden.

#### 5.4.5.3 Methoden zur Prozessmodellierung

Im Gegensatz zu den statischen Datenmodellen beschreiben Prozessmodelle eine dynamische Sicht innerhalb eines Informationsmodells. Ziel der Visualisierung von Prozessen ist es, Prozessabläufe grafisch, übersichtlich und einfach darzustellen.<sup>109</sup> Für die Modellierung von Prozessen können verschiedene Darstellungsformen unterschieden werden, nämlich die Prozessablaufdarstellung und die Swimlanedarstellung. In der Literatur findet sich zu diesen Hauptformen der Prozessmodellierung eine Vielzahl an Darstellungsvarianten, die sich in der Festlegung und Verwendung von Symbolen, Verzweigungen und Schnittstellen voneinander unterscheiden. Innerhalb dieser Arbeit wird zunächst die ereignisgesteuerte Prozesskette (EPK) als Variante der Prozessablaufdarstellung aufgeführt und ausführlicher erläutert. Exemplarisch für die Gruppe der Swimlane-Diagramme erfolgt anschließend die Betrachtung und Beschreibung der Business-Process-Model-and-Notation-Methode (BPMN).

Mit dem Beschreibungsverfahren der EPK erfolgt die Darstellung ablaufbezogener Zusammenhänge von Funktionen und Ereignissen. Ereignisse bilden den Auslösemechanismus von Funktionen und können ebenso Ergebnisse von Funktionen darstellen. Ereignisse, die keine erzeugende Funktion besitzen, werden Startereignisse genannt. Endereignisse eines Prozesses stellen Ereignisse dar, welche keine konsumierende Funktion auslösen. Im Gegensatz zu einer Funktion als zeitverbrauchendem Geschehen sind Ereignisse immer auf einen bestimmten Zeitpunkt bezogen. Ereignisse und Ergebnisse

<sup>109</sup> Vgl. Aichele 2006, S. 232.

werden innerhalb der EPK als Hexagone dargestellt und können durch logische Operatoren miteinander verknüpft werden. Die EPK sieht hierzu die Unterscheidung in die Operatoren „und“, „oder“ sowie „exklusives oder“ vor. Die Angabe von Operatoren bietet weiterhin die Möglichkeit, wiederkehrende, alternative oder parallele Abläufe zu modellieren.<sup>110</sup> Somit eignet sich die EPK neben der Definition und Kontrolle von Workflows ebenfalls zur Konfiguration von Software sowie zur Softwareentwicklung. Anhand des Beispiels „Kundenauflauftrag bestätigen“ werden in Abb. 5.34 die beschriebenen Elemente der EPK zusammenhängend dargestellt.

Eine weitere Methode zur Prozessmodellierung stellt die BPMN-Methode dar, welche seit dem Jahr 2005 durch die Object Management Group (OMG) gepflegt und weiterentwickelt wird. Schwerpunkt der Modellierungsmethode liegt auf der grafischen Darstellung von Geschäftsprozessen sowie betriebswirtschaftlichen Arbeitsabläufen.<sup>111</sup> Die Zielsetzung der BPMN-Methode wird durch die aktuelle Versionsspezifikation wie folgt definiert:<sup>112</sup>

- The primary goal of **BPMN** is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes.

Die BPMN-Modellierungsmethode ist innerhalb der letzten Jahre immer mehr in das Blickfeld von Softwareentwicklern gerückt, da sie zwei wichtige Funktionen bei der Entwicklung von Anwendungssoftware übernehmen kann. Zum einen kann eine Kommunikationsbrücke zwischen der fachlichen und technischen Prozessebene gebildet werden, zum anderen strebt die Methode die direkte Ausführbarkeit derart erstellter Prozesse an. Ein BPMN-Prozess befindet sich prinzipiell innerhalb eines sogenannten Pools, welcher einen Benutzer oder ein System repräsentiert. Ein Pool kann in einzelne Swimlanes unterteilt werden, die bspw. Abteilungen oder Benutzerrollen darstellen. Der grundlegende Aufbau eines BPMN-Prozesses erinnert somit an die Unterteilung eines Schwimmbeckens in einzelne Bahnen, wobei sich jeder Wettkampfteilnehmer nur innerhalb seiner Bahn bewegen kann.<sup>113</sup> Zur Modellierung eines Prozesses mittels BPMN können folgende grafische Elemente verwendet werden:<sup>114</sup>

- Flow Objects stellen Knoten innerhalb der BPMN-Diagramme dar. Sie können entweder als Ereignisse, Aufgaben oder Verzweigungspunkte in einem Prozessmodell auftreten. Aufgaben werden als Rechtecke, Verzweigungen als Rauten und Ereignisse als Kreise dargestellt.

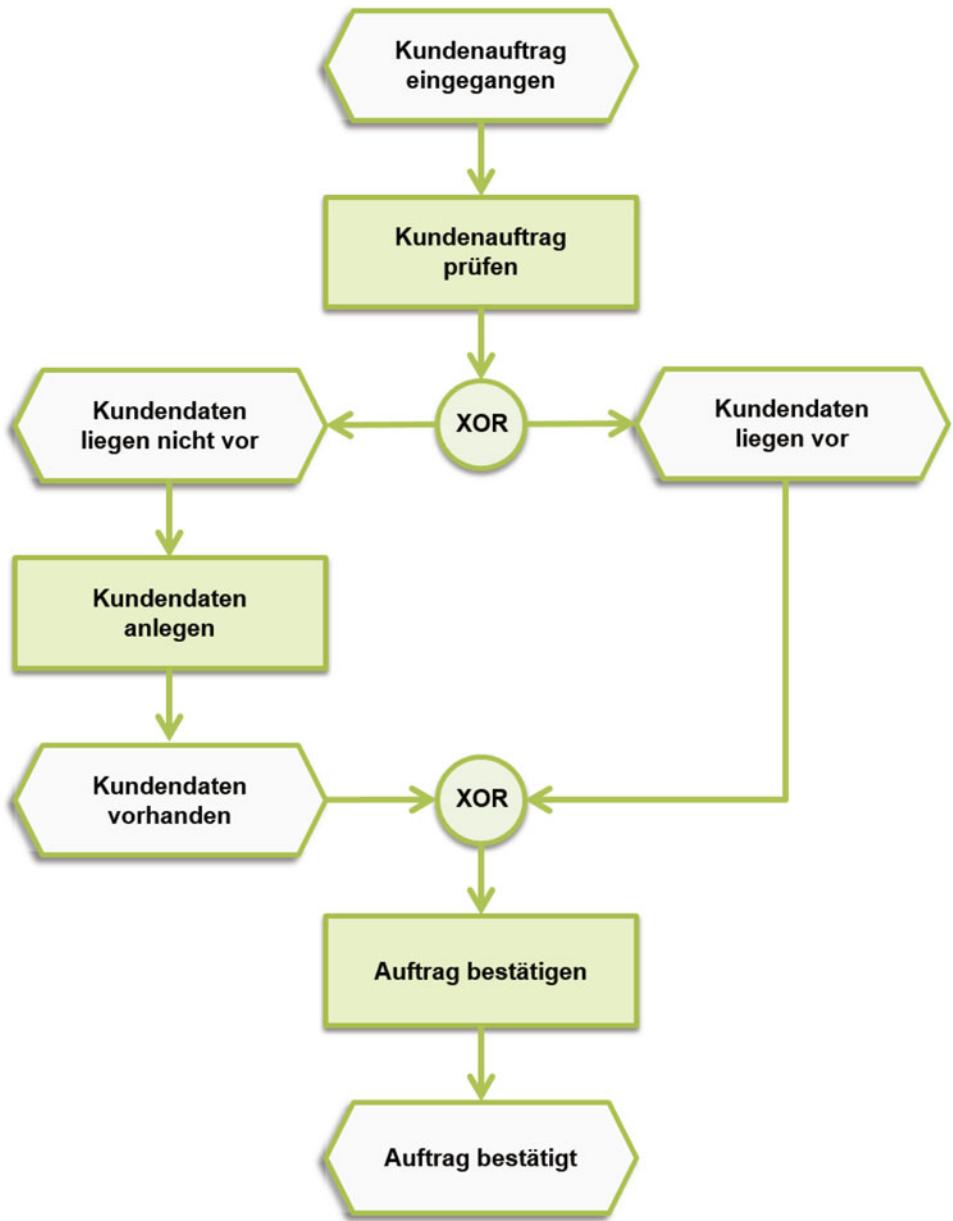
<sup>110</sup> Vgl. Aichele 2006, S. 233.

<sup>111</sup> Vgl. Allweyer 2009, S. 8.

<sup>112</sup> Vgl. Object Management Group 2011, S. 1.

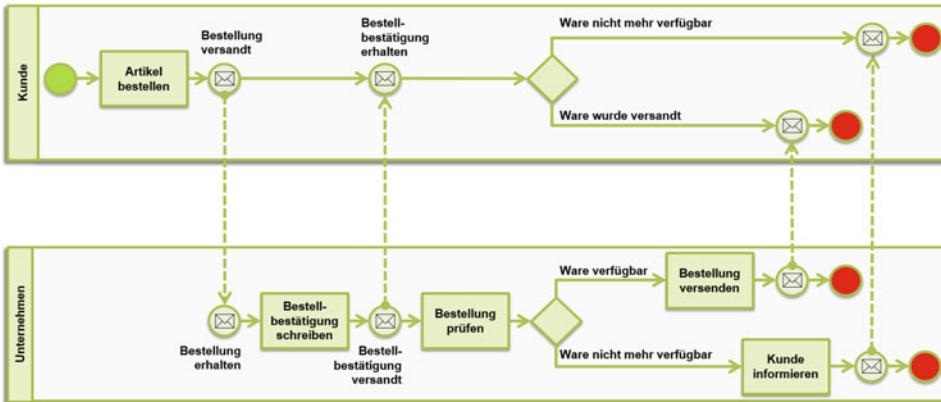
<sup>113</sup> Vgl. Allweyer 2009, S. 13 f.

<sup>114</sup> Vgl. Object Management Group 2011, S. 27 f.



**Abb. 5.34** Beispiel einer EPK zum Prozessablauf „Kundenauftrag bestätigen“

- Connecting Objects stellen die Verbindung zwischen den einzelnen Elementen innerhalb des Prozesses dar. Hierzu werden zwei verschiedene Kantenarten unterschieden.



**Abb. 5.35** Beispiel eines BPMN-Modells

Während Sequence Flows Aufgaben, Ereignisse und Verzweigungen verbinden, dienen Message Flows zur Illustration von Meldungen zwischen verschiedenen Objekten.

- Artefakte stellen Elemente dar, die zur Unterstützung von Aufgaben eingesetzt werden können oder dem besseren Verständnis des Prozessverlaufes dienen. Zu den Artefakten zählen Datenobjekte, Gruppierungsmöglichkeiten sowie Annotationen.

Zum besseren Verständnis wird in Abb. 5.35 ein einfaches BPMN-Modell angegeben. Das Modell setzt sich aus den beiden Pools „Unternehmen“ und „Kunde“ zusammen. Abgebildet wird ein allgemeiner Bestellprozess, welcher durch den Kunden ausgelöst wird.

In der Praxis kann der Ablauf zur Erstellung und Durchführung eines Bestellauftrages wesentlich komplexer und umfangreicher sein. Das in Abb. 5.35 dargestellte Modell stellt eine starke Vereinfachung dar, um ein übersichtliches Beispiel zu erhalten, an dem sich die verschiedenen Elemente der BPMN gut erläutern lassen. Beispielsweise bleibt im oben dargestellten Prozess die Mitteilung über den Versand der Ware unberücksichtigt. Die Modellierung von Softwareprojekten kann weiterhin durch rein technische Modelle, wie bspw. Anwendungsfall- oder Klassendiagramme, ermöglicht werden. Die Betrachtung dieser Modelle wird innerhalb des vorliegenden Kapitels jedoch nicht weiter verfolgt.

## 5.5 Entwicklungs- und Testphase

Die Implementierungs- bzw. Entwicklungsphase dient dem Ziel, die in der Entwurfsphase enthaltenen Ergebnisse in eine Form zu bringen, die auf mobilen Endgeräten ausführbar ist. Hierzu zählen die vollständige Detaillierung der einzelnen Komponenten, die Überführung der Algorithmen in die entsprechende Programmiersprache, die Entscheidung über Kauf bzw. Wiederverwendung bereits genutzter und verfügbarer Komponenten, die Über-

tragung des logischen Datenmodells in ein physisches Datenmodell, die Übersetzung und Prüfung der syntaktischen Korrektheit der Algorithmen, das Testen, d. h. die Prüfung der semantischen Korrektheit der Systemkomponenten, sowie hieraus resultierende syntaktische und semantische Korrekturen der fehlerhaften Systemkomponenten. Das Ergebnis der Implementierungsphase stellen die getesteten Systemkomponenten, die Protokolle der Komponententests und die Realisierung des physischen Datenmodells dar.<sup>115</sup>

Die Entwicklung von mobilen Anwendungen unterscheidet sich in verschiedener Hinsicht von der Entwicklung klassischer PC- oder webbasierter Anwendungen. In Verbindung mit der mobilen Anwendungsentwicklung zeigen sich besondere Merkmale, bspw. die unterschiedliche Anzeigegröße auf Endgeräten, die Geräteperformance sowie spezielle Entwicklungs- und Debuggingmöglichkeiten. Sensoren auf den mobilen Endgeräten ermöglichen in Verbindung mit unterschiedlichen Backendservices eine Vielzahl von Möglichkeiten in Hinblick auf Funktionalität und Usability.<sup>116</sup>

Bei der Entwicklung von Applikationen für mobile Endgeräte wird im Grundlegenden zwischen den beiden verschiedenen Ansätzen unterschieden, ob eine native Anwendung, eine webbasierte Anwendung oder eine Kombination aus beiden Ansätzen (hybride Anwendung) entwickelt werden soll. Die Wahl des Entwicklungsansatzes wirkt sich hierbei auf die konkrete Form der Entwicklung, die Nutzung der Programmiersprachen und Entwicklungsumgebungen aus:<sup>117</sup>

- Native Anwendungen werden für ein spezielles Betriebssystem erstellt und somit für eine konkrete Plattform, wie z. B. Android oder iOS, entwickelt. Die Entwicklung erfolgt mit einem Software Development Kit (SDK). Nativ entwickelte Anwendungen sind nur auf der jeweilig dafür vorgesehenen Zielplattform lauffähig.
- Webbasierte Anwendungen sind plattformunabhängig und gewähren in dieser Hinsicht eine größere Flexibilität. Dies wird durch die Nutzung von HTML5 als plattformunabhängige Basistechnik ermöglicht, sodass die Applikation innerhalb eines Browsers auf allen modernen mobilen Endgeräten lauffähig ist.

Nachfolgend werden Programmiersprachen zur mobilen Anwendungsentwicklung beschrieben sowie näher auf die Nutzung und Darstellung von Entwicklungsumgebungen und auf das Testmanagement eingegangen.

### **5.5.1 Programmiersprachen zur mobilen Anwendungsentwicklung**

Programmiersprachen stellen im Vergleich zur Kommunikation zwischen Menschen künstliche Sprachen dar, die zur Kommunikation zwischen Anwendern und Compu-

---

<sup>115</sup> Vgl. Rinza 1998, S. 13.

<sup>116</sup> Vgl. Strang und Lichtenstern 2012, S. 419.

<sup>117</sup> Vgl. Rühl und Schenkel 2012.

tern sowie zur Verarbeitung von Informationen dienen. Softwareentwickler verwenden Programmiersprachen für die Formulierung einer für den Computer verständlichen Anwendung. In diesem Zusammenhang lässt sich der Begriff Programmierung wie folgt definieren:<sup>118</sup>

► **Programmierung** Der Begriff der Programmierung beschreibt die Umsetzung der funktionalen Beschreibung eines Software-Systems in den Quelltext einer bestimmten Programmiersprache.

Für die erfolgreiche Umsetzung von Programmen müssen Syntax und Semantik der Programmiersprache eingehalten werden. Mithilfe kontextfreier Grammatiken definiert die Syntax alle zulässigen Wörter, die durch eine Programmiersprache formuliert werden können. Durch die Semantik werden den einzelnen Wörtern der Syntaxdefinitionen bestimmte Bedeutungen zugewiesen. Die Beschreibung der Semantik erfolgt üblicherweise auf textueller Grundlage.<sup>119</sup> Je nach Abstraktionsgrad und Struktur können Programmiersprachen in folgende Bereiche eingeteilt werden:<sup>120</sup>

- **Maschinensprachen:** Beschreiben eine Reihenfolge an einzelnen Befehlen, die durch die Hardware eines Prozessors festgelegt ist.
- **Assemblersprachen:** Unterscheiden sich von Maschinensprachen durch die Anwendung von Befehlswörtern mit zugeordneten Parametern.
- **Höhere Programmiersprachen:** Beschreiben algorithmische Verfahren in einer rechnerunabhängigen Form.
- **Anwendungsorientierte Sprachen:** Enthalten Syntax und Semantik für einen eingeschränkten Anwendungsbereich.
- **Dokumentenbeschreibungssprachen:** Beschreiben die logische Struktur von Textdokumenten.

Programmiersprachen werden hauptsächlich im Bereich der Anwendungsentwicklung eingesetzt. Neben der Auswahl der richtigen Programmiersprache sind Erfahrungen sowie das methodische Vorgehen bei der Softwareentwicklung entscheidend für den Projekterfolg. Für die Entwicklung mobiler Anwendungen werden gegenwärtig die Programmiersprachen Java, C#, Objective C oder HTML 5 eingesetzt.

Mit dem Aufkommen des Internets änderte sich die Zielrichtung der Entwicklung auf die Unterstützung von Internetseiten durch grafikfähige und interaktive Programmelemente. In diesem Zusammenhang hat sich die Programmiersprache Java etabliert, die sich seit der Veröffentlichung im Jahr 1996 durch die Firma Sun Microsystems zu einer umfangreichen und leistungsstarken Softwaretechnologie entwickelt hat. Für die

---

<sup>118</sup> Henning et al. 2007, S. 19.

<sup>119</sup> Vgl. Henning et al. 2007, S. 10.

<sup>120</sup> Vgl. Victor 2007, S. 199.

Entwicklung und Anwendung von Java-Programmen werden ein Editor für die Eingabe und Bearbeitung des Quellcodes, ein Compiler für die Übersetzung des Programms sowie ein Java-Laufzeitsystem für die Ausführung des Programms benötigt. Die Entwicklung mobiler Anwendungen für das Android-Betriebssystem setzt weiterhin die Installation zusätzlicher Java-Bibliotheken in die Laufzeitumgebung voraus.<sup>121</sup>

Die Programmiersprache C ist eine von Dennis Richie und Brian Kernighan im Jahr 1978 veröffentlichte Sprache, die zur Bearbeitung nichtnumerischer Probleme für die Maschinenprogrammierung entwickelt wurde. C zählt zu den imperativen Programmiersprachen und besitzt ebenfalls, wie die Sprache Java, eine Funktionsbibliothek, die durch spezielle Bibliotheken beliebig modifiziert werden kann.<sup>122</sup> Im Laufe der Zeit haben sich viele weitere Sprachen aus der ursprünglichen C-Sprache entwickelt, wie bspw. C++ oder C# (C sharp). C# bietet eine Reihe an Eigenschaften, Methoden und Attributen zur Entwicklung komponentenorientierter Programme und stellt somit eine objektorientierte Version der C-Sprache dar.

Objective-C wurde von Brad Cox und Tom Love zu Beginn der 80er-Jahre entwickelt und stellt ebenfalls wie C# eine Erweiterung der Programmiersprache C dar. Objective-C ist eine strikte Obermenge der Programmiersprache C und erlaubt eine nahtlose Vermischung der C-Syntax mit der Objective-C-Syntax. Dieser hybride Ansatz ermöglicht es, Objective-C-Eigenschaften aus den imperativen und objektorientierten Programmierparadigmen zuzuweisen. Objective-C liefert heute die Grundlage für Apples Betriebssysteme Mac OS X und iOS.<sup>123</sup>

Das anfängliche Ziel von HTML bildete der Austausch von Informationen über einen einfachen und strukturierten digitalen Weg. Gegenwärtige Webtechnologien verlangen jedoch mehr als diesen ursprünglichen Ansatz, sodass die HTML-Sprache seit der Veröffentlichung ständig weiterentwickelt und erweitert wurde. Aktuell befindet sich die Programmiersprache in der fünften Version. Webbasierte Applikationen beruhen, aufbauend auf der Internettechnologie und der damit verbundenen Client-Server-Architektur, auf HTML-gestützten Webseiten, die zur Ausführung und Nutzung einen Internetbrowser benötigen. Webbasierte Anwendungen unterscheiden sich im Gegensatz zu herkömmlichen Webseiten darin, dass sie ein ähnliches Aussehen und Verhalten wie bei einer nativen Anwendung erreichen. Gleichzeitig unterliegen Sie jedoch Einschränkungen, wie bspw. dem begrenzten Zugang zu den Hardwareressourcen mobiler Endgeräte sowie der geringen Performance bei multimedialen Anwendungen.<sup>124</sup>

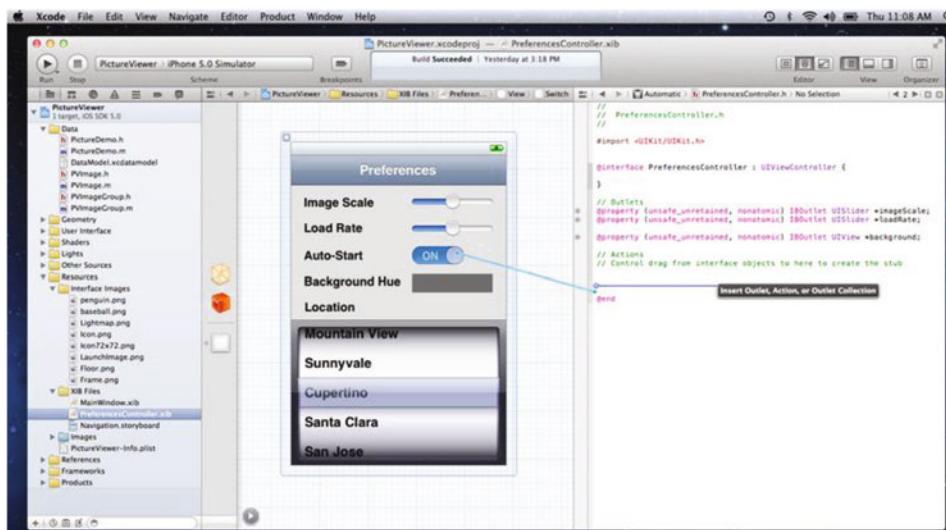
---

<sup>121</sup> Vgl. Felker 2011, S. 47.

<sup>122</sup> Vgl. Erlenkötter 2005, S. 11 f.

<sup>123</sup> Vgl. Gall et al. 1995, S. 32 f.

<sup>124</sup> Vgl. Albert und Stiller 2012, S. 148 ff.



**Abb. 5.36** Die Xcode-Entwicklungsumgebung mit Sicht auf das Projekt und zugehörige Ressourcen.  
(Rühl und Schenkel 2012)

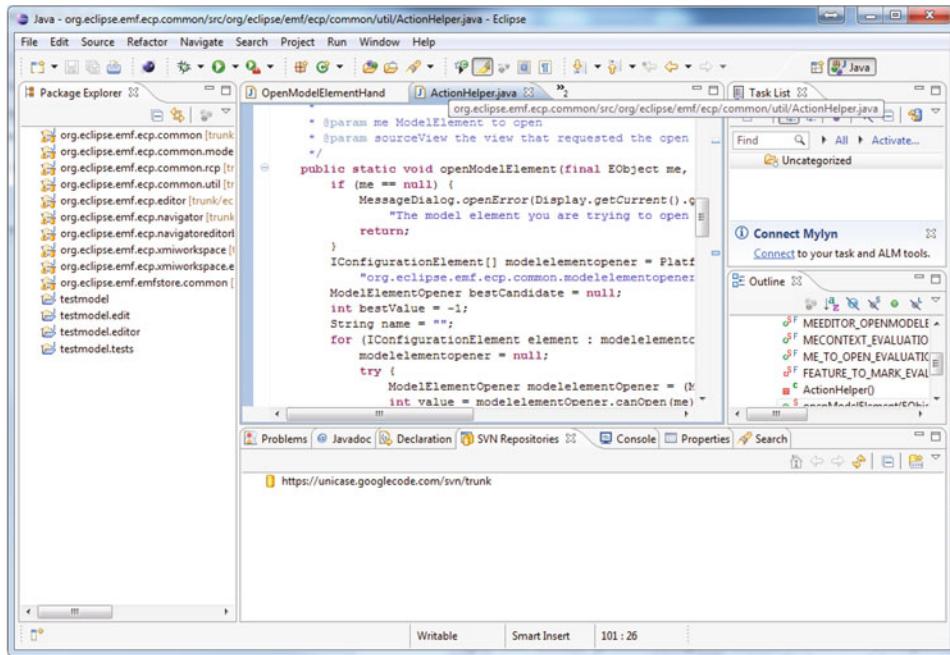
### 5.5.2 Nutzung und Darstellung von Entwicklungsumgebungen für mobile Anwendungen

Bei der Entwicklung von mobilen Anwendungen werden, je nach Zielplattform, Entwicklungsansatz sowie persönlichen Kenntnissen und Fertigkeiten, unterschiedliche Ansätze verfolgt. Die Integration des Android Development Kits in die Entwicklungsumgebung Eclipse verfolgt in diesem Kontext andere Rahmenbedingungen, als bspw. die Entwicklung von iOS-Anwendungen in der Apple Entwicklungsumgebung Xcode (siehe Abb. 5.36). Während die Programmierung von Anwendungen in Eclipse (siehe Abb. 5.37) mittels der Programmiersprache Java erfolgt, werden mobile Applikationen für das iOS-Betriebssystem auf Basis der Programmiersprache Objective-C entwickelt (vgl. Abschn. 5.5.1).<sup>125</sup>

Die Organisation eigener Entwicklungsprojekte erfolgt innerhalb der beiden Entwicklungsumgebungen Xcode und Eclipse durch eine paketorientierte Struktur, in der für die jeweiligen Projekte benötigte Programmelemente, Layouts und Klassen verwaltet werden können. Die Entwicklungsumgebungen erlauben zudem die sukzessive Weiterentwicklung sowie den Import und Export von Entwicklungspaketen.

Das Layout für mobile Anwendungen wird mit dem in Eclipse integrierten Layoutdesigner vorgenommen. Somit ist es möglich, eine grafische Benutzeroberfläche gemäß dem Entwurf einzurichten sowie Funktionalitäten (bspw. Buttons, Textfelder etc.) zur Interaktion mit dem Anwender zu integrieren. Hierbei wird in den Quelltext der Rahmen

<sup>125</sup> Vgl. Rühl und Schenkel 2012.



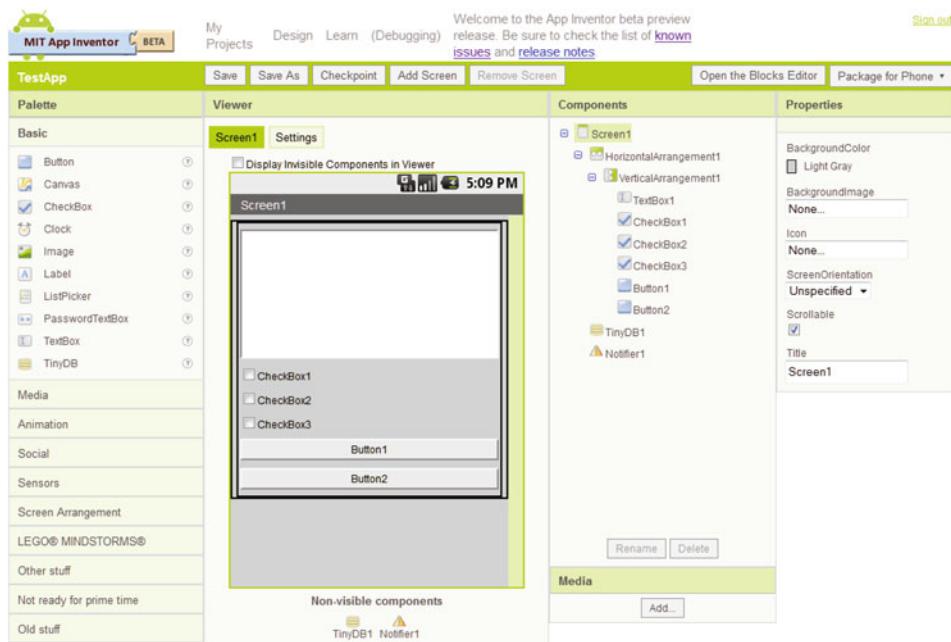
**Abb. 5.37** Die Eclipse-Entwicklungsumgebung mit Sicht auf das Projekt und zugehörige Ressourcen

zur Nutzung und Ausgestaltung der grafischen Elemente für die Programmsteuerung mit integriert und kann je nach Bedarf ausgeprägt werden.

Für die Entwicklung mobiler Anwendungen für Android bietet Google weiterhin eine eigene webbasierte Entwicklungsumgebung namens App Inventor an. Die Entwicklung einer Anwendung mittels App Inventor erfolgt in zwei voneinander getrennten Arbeitsschritten. Im ersten Schritt erfolgt durch den Entwickler die Gestaltung der Applikation. Über die Browseranwendungen wird hierzu der sogenannte Designer gestartet. Zur Gestaltung der Applikation müssen aus einer integrierten Funktionsbibliothek gewünschte Steuerelemente, bspw. Textfelder, Schaltflächen oder Beschriftungen, auf die zukünftige Programmoberfläche der Applikation gezogen, benannt und positioniert werden.<sup>126</sup>

Im zweiten Schritt erfolgt durch die Zuweisung von Methoden und Funktionen die Bearbeitung der Steuerelemente. In einem separaten Java-Editor müssen für die Herstellung lauffähiger Algorithmen funktionale Blöcke miteinander verbunden werden, die ähnlich dem Befehlssatz einer klassischen Programmiersprache, die Syntax der Entwicklungssprache des App Inventors darstellen. Die Entwicklung mobiler Applikationen beruht somit auf dem systematischen Zusammenwirken der Steuerelemente aus dem Designer sowie

<sup>126</sup> Vgl. Kloss 2011, S. 59 f.



**Abb. 5.38** Designansicht des App Inventors

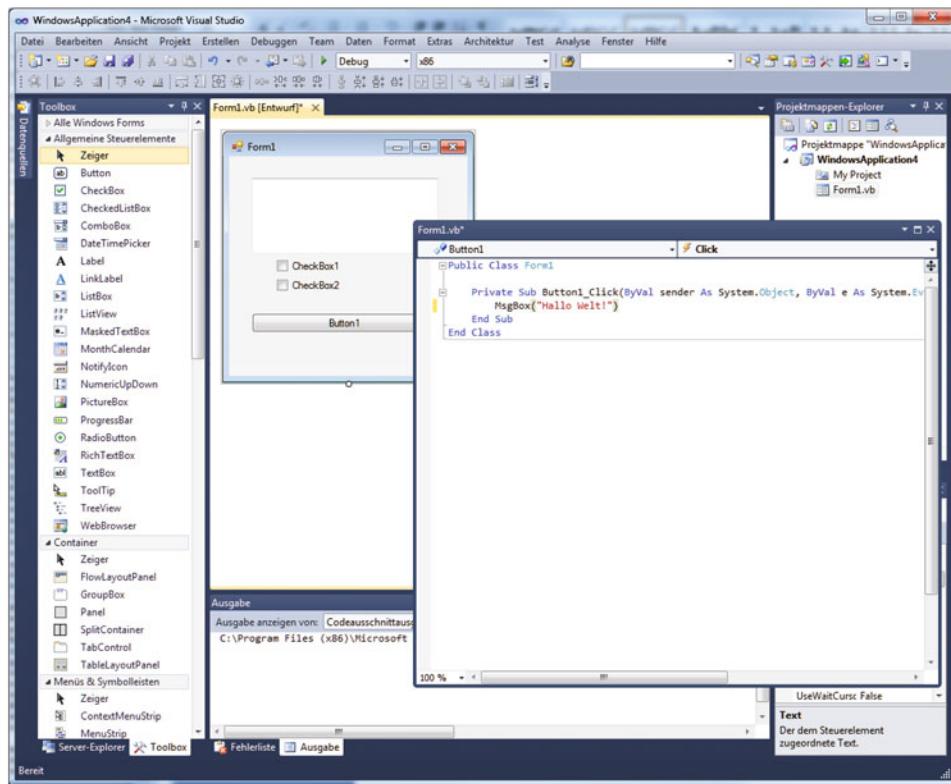
der funktionalen Blöcke aus dem sogenannten Blocks-Editor.<sup>127</sup> Abbildung 5.38 zeigt den Designer des App Inventors.

Die Entwicklung von Windows-Phone-Anwendungen kann über die von Microsoft zur Verfügung gestellte Entwicklungsumgebung Visual Studio ermöglicht werden. Programmierer können mittels Visual Basic klassische Windows-Programme sowie mobile Anwendungen oder dynamische Webseiten unter Verwendung der Programmiersprachen C, C++ oder C# entwickeln. Des Weiteren können Entwickler für die individuelle Anpassung an die Arbeitsbedürfnisse benutzerdefinierte Modifikationen und Ergänzungen in die Entwicklungsumgebung integrieren oder die bereits vorhandenen Funktionen für das Dokumentieren, Analysieren und Testen der Software nutzen.<sup>128</sup>

Gegenwärtig werden vier unterschiedliche Versionen von Visual Studio ausgeliefert: Visual Studio Express, Professional, Premium und Ultimate. Die Express-Versionen der Entwicklungsumgebung sind kostenlos, im Funktionsumfang reduziert und auf nur eine Programmiersprache beschränkt, bspw. Visual C# Express oder Visual C++ Express. Mit Visual Studio Professional können Entwickler im Gegensatz zur Express-Variante mehrere Programmiersprachen verwenden und Anwendungen für mobile Endgeräte entwickeln. Die Premium- und Ultimate-Varianten der Entwicklungsumgebung enthalten

<sup>127</sup> Vgl. Kloss 2011, S. 75 f.

<sup>128</sup> Vgl. Microsoft 2013.



**Abb. 5.39** Designansicht von Visual Studio 2010

weitere Funktionen und Methoden für die teambasierte Anwendungsentwicklung sowie zur Verwaltung des gesamten Entwicklungslebenszyklus einer Applikation.<sup>129</sup> Abbildung 5.39 zeigt die Designansicht von Visual Studio 2010.

### 5.5.3 Testmanagement beim Entwicklungsprozess von mobilen Anwendungen

Das systematische Testen von Software gehört zu einer der wichtigsten Tätigkeiten im Softwareentwicklungsprozess, da dadurch Abweichungen des Softwareproduktes von den Spezifikationen aufgedeckt und Fehler bei der Implementierung identifiziert werden. Durch den Vertrieb mangelhafter Software können immense Kosten verursacht werden. Diese Kosten sind umso höher, je später im Laufe des Entwicklungsprozesses Fehler entdeckt und korrigiert werden.<sup>130</sup>

<sup>129</sup> Vgl. Kotz 2011, S. 33.

<sup>130</sup> Vgl. Grechenig et al. 2010, S. 300.

In der Realität wird oftmals bei der Durchführung ausführlicher Softwaretests gespart. Im Bereich der mobilen Anwendungsentwicklung haben laut der Capgemini-Studie „World Quality Report 2012–2013“ 31 % der befragten Unternehmen Tests durchgeführt. Insgesamt gaben 34 % an, keine passende Vorgehensweise für das Testen mobiler Anwendungen zu haben, weitere 54 % gaben an, nicht über die notwendigen Testdevices zu verfügen.<sup>131</sup>

Softwaretests zielen somit primär darauf ab, das Verhalten einer Software zu validieren und zu überprüfen, ob dieses Verhalten der angedachten zukünftigen Nutzung entspricht. Das Testen von Software ist jedoch nicht immer eine Garantie dafür, dass zum einen ein erfolgreicher Projektablauf garantiert und eine fehlerfreie Software vertrieben wird. Fehler bei der Planung und Durchführung von Tests sowie eine mangelhafte Dokumentation identifizierter Probleme können ebenfalls den Erfolg eines Softwareproduktes erschweren.<sup>132</sup> Gründe hierfür sind bspw. die unzureichende Übertragbarkeit von Testvarianten für PC-Anwendungen oder auch mangelnde Kenntnisse über geeignete Testmethoden.

Im Zusammenhang mit der Entwicklung von Software wird der Begriff „Testen“ unterschiedlich definiert. Eine umfassende Begriffsdefinition lautet wie folgt:<sup>133</sup>

- **Testen** ist der Prozess, der sämtliche (Test-)Aktivitäten umfasst, welche dem Ziel dienen, für ein Software-Produkt die korrekte und vollständige Umsetzung der Anforderungen sowie das Erreichen der festgelegten Qualitätsanforderungen nachzuweisen. Zum Testen gehören in diesem Sinne auch Aktivitäten zur Planung, Steuerung, Vorbereitung und Bewertung, welche der Erreichung der genannten Ziele dienen.

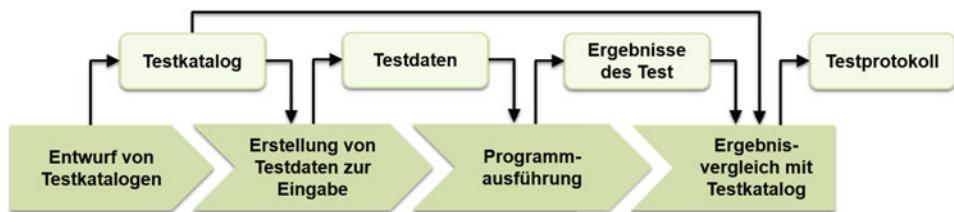
Beim Testen von Software gibt es unterschiedliche Sichten auf bzw. in das betrachtete Testobjekt, die durch sogenannte Box-Tests beschrieben werden. Im Laufe der Zeit haben sich drei Box-Tests etabliert: der Blackbox-, der Whitebox- sowie der Greybox-Test. Als Blackbox-Test wird ein funktionaler oder nichtfunktionaler Test verstanden, der ohne Betrachtung der internen Programmstrukturen die fehlerfreie und vollständige Umsetzung der Anforderungen überprüft. Beim Blackbox-Test werden nur die Eingaben und Ausgaben des Testobjektes analysiert und überprüft. Im Gegensatz hierzu überprüft ein Whitebox-Test, auch als Strukturstest bezeichnet, die innere Struktur eines Softwareprogramms auf Korrektheit und Vollständigkeit. Im Fokus des Whitebox-Tests steht neben den Systemstrukturen und Programmabläufen der Software der zugrunde liegende Programmcode. Der Greybox-Test verbindet die zuvor genannten Box-Tests miteinander und erlaubt somit die Überprüfung der korrekten Umsetzung der Anforderungen, des Programmablaufs und des richtigen Aufbaus der inneren Strukturen.<sup>134</sup>

<sup>131</sup> Vgl. Capgemini 2012, S. 24.

<sup>132</sup> Vgl. Grechenig et al. 2010, S. 300.

<sup>133</sup> Franz 2007, S. 24.

<sup>134</sup> Vgl. Franz 2007, S. 28 f.



**Abb. 5.40** Modell des Vorgehens beim Softwaretest. (Eigene Erstellung, in Anlehnung an Sommerville 2001, S. 581)

Bezüglich der Vorgehensweise zum Testen von mobilen Anwendungen können verschiedene Ansätze und Möglichkeiten in der Praxis verfolgt werden. Im Folgenden werden zwei Ansätze detailliert:<sup>135</sup>

- Gestaltung der Vorgehensweise und des Testrahmens zur Softwarevalidierung durch die Erstellung von Testkatalogen und Testfällen sowie deren Ergebnisprotokollierung
- Testumgebungen und Simulationen bei Nutzung entsprechender Entwicklungsumgebungen

Die Testansätze dienen dazu, eine Applikation so zu testen, wie diese bei korrekter oder fehlerhafter Nutzung funktionieren würde. Die Abb. 5.40 verdeutlicht das Vorgehen zum Test von Applikationen.

Bei dem Vorgehen zur Durchführung von Softwaretests ist es von wesentlicher Bedeutung, einen Testkatalog mit verschiedenen Testfällen vorzubereiten, in dem enthalten ist, welche Funktionen der Software getestet werden sollen. Im nächsten Schritt sind für die einzelnen Testfälle im Testkatalog Testdaten zur Eingabe vorzubereiten, mit denen anschließend eine konkrete Programmausführung vorgenommen wird. Die Testergebnisse aus dem Testvorgehen werden wiederum mit den erwarteten Testergebnissen aus dem Testkatalog verglichen und die positiven bzw. negativen Ergebnisse in einem Testprotokoll vermerkt. Fehlerfälle bedingen Änderungsbedarfe bei der Entwicklung der entsprechenden Applikation und dienen zur Verbesserung des Softwareverhaltens in Hinblick auf die anforderungsgemäße Nutzung.

Der mobilen Anwendungsentwicklung ist somit dem intensiven und fortlaufenden Testen einer mobilen Applikation eine hohe Bedeutung zuzumessen. Das qualitative Testen einer Applikation gewährleistet die Überprüfung der eigenen Entwicklungen auf syntaktische und semantische Korrektheit, um die Funktionalitäten der Applikation zu überprüfen und in Bezug auf die Anforderungen zu validieren. Nachfolgend wird eine Auswahl an Werkzeugen zur Testphase vorgestellt.

<sup>135</sup> Vgl. Sommerville 2001, S. 581.

## 5.5.4 Werkzeuge zur Entwicklungs- und Testphase

Im Laufe der Zeit hat sich für die Entwicklung mobiler Anwendungen eine Vielzahl an nützlichen Werkzeugen etabliert, die dem Entwickler die Programmierung einer Applikation vereinfachen. Nachfolgend wird näher auf den Einsatz von Frameworks und Debugging Tools für die Umsetzung mobiler Anwendungen eingegangen. Bei Frameworks handelt es sich um einen Ordnungsrahmen zur Softwareentwicklung, der Entwicklern vorgefertigte Designstrukturen und Codefragmente zur Verfügung stellt. Debugging Tools werden während des Entwicklungsprozesses, zur Überprüfung und Verbesserung des Programmcodes, eingesetzt.

### 5.5.4.1 Frameworks für die mobile Anwendungsentwicklung

In Bezug auf die Entwicklung mobiler Anwendungen wird unter einem Framework ein Rahmenwerk bzw. ein Programmiergerüst verstanden, das den Entwicklern verschiedene Bibliotheken und Programmstrukturen zur Verfügung stellt und dadurch die Softwarearchitektur vorgibt.<sup>136</sup> Für die mobile Anwendungsentwicklung lassen sich Frameworks unterscheiden in jene, die den gesamten Bereich der Entwicklung abdecken, und in solche, die sich nur auf Teilbereiche beschränken.<sup>137</sup>

Durch die Verwendung von Frameworks können Softwareentwickler auf entsprechende Standardbausteine zugreifen, sodass nicht bei jedem neuen Entwicklungsprojekt das Rad neu erfunden werden muss. Frameworks weisen insbesondere in Bezug auf die Designstruktur Vorteile auf, da bspw. die Programmierung von Buttons, Eingabefeldern oder Statusmeldungen nicht von Grund auf neu erfolgen muss. Durch den Zugriff auf Bibliotheken und vorgefertigte Codebausteine kann durch den Einsatz von Frameworks der Entwicklungsaufwand deutlich verkürzt werden.<sup>138</sup>

Frameworks werden den Entwicklern meist kostenlos zur Verfügung gestellt. Für die Entwicklung nativer, hybrider oder webbasierter Applikationen können unterschiedliche Frameworks zum Einsatz kommen. Die Entwicklung nativer Anwendungen kann durch Frameworks wie PhoneGap oder Titanium Mobile ermöglicht werden. Die Frameworks werden zunächst mittels webbasierter Programmiersprachen, bspw. JavaScript, HTML oder CSS, entwickelt und anschließend in native Komponenten kompiliert.<sup>139</sup> Da es sich bei der Art der hybriden Anwendungen im Prinzip um eine Webanwendung handelt, können beliebige JavaScript-Frameworks, wie bspw. Sencha Touch oder The M-Projekt, verwendet werden.<sup>140</sup> Im Bereich der webbasierten Anwendungsentwicklung kommen insbesondere die Frameworks jQuery oder jQuery Mobile zum Einsatz<sup>141</sup>, wobei

---

<sup>136</sup> Vgl. IT Wissen 2013.

<sup>137</sup> Vgl. Werler 2012, S. 127 f.

<sup>138</sup> Vgl. Vollendorf und Bongers 2010, S. 18.

<sup>139</sup> Vgl. Gerlicher 2012, S. 170.

<sup>140</sup> Vgl. Gerlicher 2012, S. 162.

<sup>141</sup> Vgl. Werler 2012, S. 127 f.



**Abb. 5.41** PhoneGab-Framework. (Eigene Erstellung, vgl. Dunne 2012)

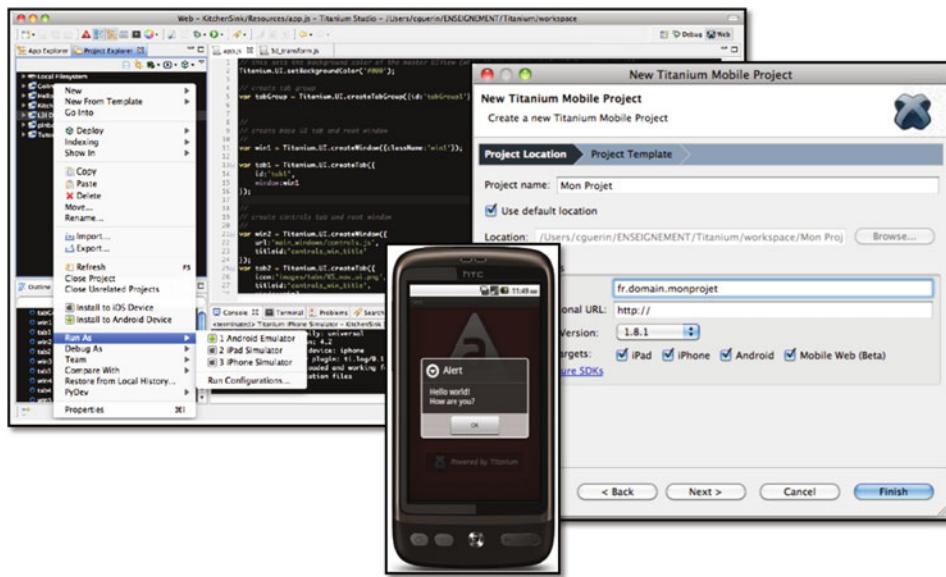
letzteres ebenfalls für die Entwicklung hybrider Anwendungen verwendet werden kann. Nachfolgend werden einige der oben genannten Frameworks kurz vorgestellt.

PhoneGab ist eine Open-Source-Software des Unternehmens „Adobe Systems“ zur Erstellung von plattformübergreifenden Anwendungen für mobile Endgeräte. Wie bereits erwähnt können mittels CSS, HTML oder JavaScript Anwendungen erstellt und durch die von PhoneGab zur Verfügung gestellten plattformspezifischen Anwendungsvorlagen in native Applikationen kompiliert werden. Aktuell werden durch PhoneGab die Betriebssysteme Android, iOS, Windows Phone 7 und 8, BlackBerry OS, Symbian OS, webOS und Bada unterstützt.<sup>142</sup> PhoneGap stellt neben Bibliotheken für die webbasierte Anwendungsentwicklung weiterhin native Funktionen, wie z. B. die Ansteuerung der Beschleunigungssensoren des Endgerätes, zu Verfügung. Ein Nachteil bei der Verwendung von PhoneGap besteht darin, dass die Kompilierung der webbasierten Anwendung in eine native Applikation erst dann erfolgen kann, wenn die jeweilige SDK des Betriebssystems vorhanden ist.<sup>143</sup> Abbildung 5.41 gibt einen kurzen Einblick in das PhoneGab-Framework.

Das Titanium-Mobile-Framework ist ebenfalls ein Open-Source-Projekt, das von der Firma Appcelerator für die Entwicklung mobiler Anwendungen zur Verfügung gestellt wird. Um die Anwendungsinhalte in einen nativen Code umzuwandeln, verwendet Titanium Mobile im Gegensatz zu PhoneGap keine Browserkomponente, sondern einen JavaScript-Interpreter. Die Entwicklung der mobilen Applikation ist damit nur in JavaScript möglich. Weiterhin werden durch Titanium Mobile nur die native Kom-

<sup>142</sup> Vgl. Adobe 2013.

<sup>143</sup> Vgl. Gerlicher 2012, S. 171.



**Abb. 5.42** Titanium-Mobile-Framework. (Eigene Erstellung, vgl. Guérin 2012)

pilierung in Android- und iOS-Anwendungen unterstützt. Hierzu müssen ebenfalls wie bei PhoneGap die entsprechenden SDKs vorinstalliert sein.<sup>144</sup> Ein Einblick in das Titanium-Mobile-Framework wird durch Abb. 5.42 gegeben.

### 5.5.4.2 Einsatz von Debugging Tools für die mobile Anwendungsentwicklung

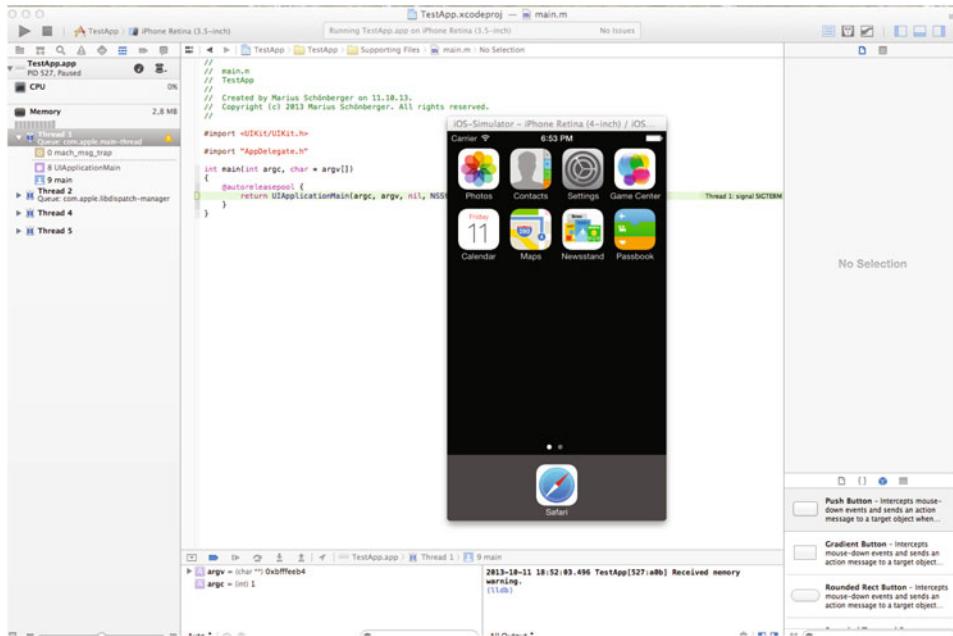
Der Gedanke, die gesamte Programmentwicklung durch geeignete und systemnahe Software zu unterstützen, basiert auf der Forderung, den hierzu benötigten Zeit-, Ressourcen- und Änderungsaufwand zu reduzieren. Generell wird in diesem Zusammenhang ein Softwareentwicklungswerkzeug als Programm definiert, das die Softwareentwicklung vereinfacht sowie beschleunigt und dabei gleichzeitig die Softwarequalität verbessert.<sup>145</sup> Während des Entwicklungsprozesses sowie hinsichtlich der Veröffentlichung fehlerfreier Software bestehen intensive Bemühungen, den Testprozess zu systematisieren und effektiver zu gestalten. Aus Sicht der Entwickler muss der Test:<sup>146</sup>

- alle Programmanweisungen zur Ausführung bringen,
- alle Programmverzweigungen berücksichtigen und
- alle Programmschleifen aktivieren und durchlaufen.

<sup>144</sup> Vgl. Gerlicher 2012, S. 173.

<sup>145</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 292 f.

<sup>146</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 290 f.



**Abb. 5.43** iOS-Simulator. (Rühl und Schenkel 2012)

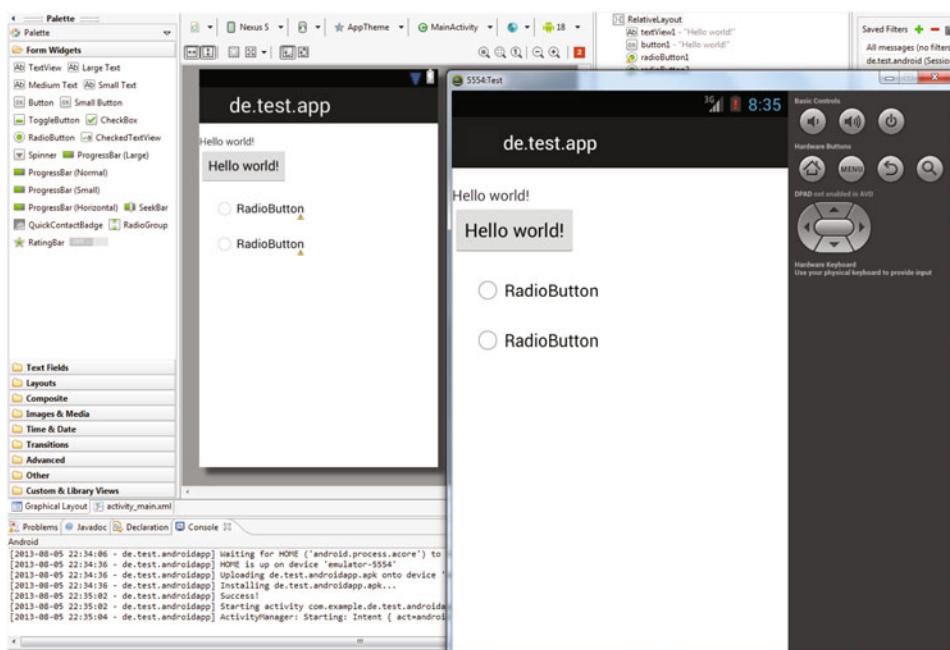
Abhängig von der Anzahl und Qualität der Testhilfen kann der Testbetrieb effizienter gestaltet werden. Zu den Testhilfen zählen bspw. Programme zur Ablaufüberwachung und -protokollierung, die auch als Debugger bezeichnet werden. Nachfolgend werden die in der Eclipse- und Xcode-Entwicklungsumgebung eingebundenen Debugger vorgestellt.

In Hinblick auf die Testdurchführung mobiler Anwendungen ist es von wesentlicher Bedeutung, die Applikationen unabhängig von angeschlossenen Endgeräten zu simulieren und eine konkrete Programmausführung gemäß der zuvor vorbereiteten Testfälle im Testkatalog mit entsprechenden Testdaten zu ermöglichen. Die Xcode-Entwicklungsumgebung bietet für die Simulation des Betriebssystems und damit für das Testen von iOS-Anwendungen die Möglichkeit, iPhone- und iPad-Geräte darzustellen und die Applikation als Desktopanwendung zu emulieren.<sup>147</sup> Abbildung 5.43 zeigt den in Xcode integrierten iOS-Simulator, der ein effizientes Debugging mobiler iOS-Anwendungen ermöglicht.

Für Testfälle im Android-Bereich bietet Eclipse mit dem Android-SDK ebenfalls die Möglichkeit, die entwickelte Applikation auf verschiedenen simulierten Endgeräten zu testen (siehe Abb. 5.44). Zur Testunterstützung bietet Google die Android Development Tools, die folgende Testarten unterstützen:<sup>148</sup>

<sup>147</sup> Vgl. Sadun 2009, S. 22.

<sup>148</sup> Vgl. Rühl und Schenkel 2012.



**Abb. 5.44** Beispiel für das Debugging einer Android-Anwendung

- Der Dalvik Debug Monitor Server (DDMS) dient dem Debugging von mobilen Applikationen auf angeschlossenen Android-Geräten sowie zum Auslesen von Log-Daten und Gerätedaten.
- Das Android Virtual Device (AVD) ist ein Emulator, mit dem sich aus der Entwicklungsumgebung heraus nahezu jedes Android-Gerät mit entsprechenden Spezifika und Hardwareeigenschaften (z. B. Speichergröße, Bildschirmauflösung und Betriebssystemversion) emulieren lässt.

Es ist zu empfehlen, die Testphase bei der Entwicklung bereits in einem frühen Stadium und regelmäßig an einem angeschlossenen Endgerät durchzuführen, um die Tests unter realen Bedingungen durchzuführen. Das Ziel dieser Endgerätestests ist die Kontrolle, ob sich die entwickelten Applikationen auf einem Endgerät anders verhalten als in der Testumgebung bzw. in der Simulation. Der Test auf einem mobilen Endgerät ist insbesondere in Hinblick auf die Speicherverfügbarkeit, den Zugriff auf Internet- und Backenddienste sowie auf die gerätespezifische Hardware zu empfehlen. Bei webbasierten Applikationen, die sich aufgrund der Plattformunabhängigkeit und der unterschiedlichen Laufzeitumgebungen unterschiedlich verhalten, ist des Weiteren das Testen der Anwendung auf verschiedenen Webbrowsern zu empfehlen.<sup>149</sup>

<sup>149</sup> Vgl. Rühl und Schenkel 2012.

## 5.6 Einführungs- und Veröffentlichungsphase

Unter der Einführungs- und Veröffentlichungsphase werden alle Tätigkeiten subsumiert, die nach der Umsetzung des Sollkonzeptes der Inbetriebnahme der neuen Softwarelösung dienen. Durch die bereits genannten verschiedenen Entwicklungsausrichtungen wird der Prozess der mobilen Anwendungsentwicklung entweder durch die Einführung der mobilen Lösung bei einem Kunden bzw. Unternehmen (B2B) oder durch die Veröffentlichung auf einem Onlinemarketplace (B2C) beendet (vgl. Abb. 5.16). In der Regel sind für die Einführung einer neuen Softwarelösung bei einem Kunden umfangreiche Vorarbeiten notwendig, wie z. B. die Durchführung von Schulungen, organisatorischen Maßnahmen in den betroffenen Bereichen sowie die Planung und Bereitstellung benötigter Ressourcen. Hierbei erlangen insbesondere folgende Aktivitäten des Projektmanagements eine besondere Bedeutung:<sup>150</sup>

- Projektorganisation, d. h. Bildung von Projektteams, z. B. zur Durchführung von Schulungen, sowie verstärkte Einbeziehung von Vertretern der zukünftigen Anwender in das Projektteam.
- Projektplanung, d. h. Planung der Einführung und -strategie sowie Ausarbeitung eines Zeit- und Ressourcenplans zur Softwareeinführung.
- Projektcontrolling, d. h. detaillierte und konsequente Überwachung der Projektzeit- und Kostenpläne.

Aus Sicht der Anwender und der Auftraggeber stellt die Einführung und Inbetriebnahme einer neuen Lösung ein entscheidendes Ereignis dar, sodass der Aufwand für die Vorbereitung der Einführung nicht unterschätzt werden darf. Neben der Übergabe des Gesamtproduktes müssen weiterhin alle mit dem Projekt erstellten Dokumentationen an den Auftraggeber weitergeleitet werden. Darüber hinaus müssen im Zuge der Einführung ggf. die Übernahme existierender Datenbestände aus Altsystemen übernommen sowie die neue Lösung über einen gewissen Zeitraum unter realen Bedingungen getestet werden.<sup>151</sup>

In den nachfolgenden Ausführungen wird auf die Einführung und den Betrieb von mobilen Applikationen im B2B-Bereich sowie auf unterschiedliche Vertriebsmöglichkeiten mobiler Anwendungen im B2C-Bereich eingegangen. Weiterhin werden Marketingkonzepte für mobile Applikationen vorgestellt. Das Kapitel endet mit der Darstellung hilfreicher Werkzeuge zur Einführungs- und Veröffentlichungsphase.

---

<sup>150</sup> Vgl. Ammenwerth und Haux 2005, S. 229 f.

<sup>151</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 319.

### 5.6.1 Einführung und Betrieb von mobilen Applikationen im B2B-Bereich

Zu den typischen Aktivitäten der Systemeinführung gehören die Abnahme der erstellten Softwarelösung, die Festlegung einer Einführungsstrategie, die Übernahme von Datenbeständen sowie die Durchführung von Schulungen und die eigentliche Inbetriebnahme der Software. An die Inbetriebnahme knüpft die Betriebs- und Wartungsphase der Software an. Mit der förmlichen Abnahme der erstellten Softwarelösung beginnt der eigentliche Einführungsprozess. Für die erfolgreiche Abnahme sollte geprüft werden, ob<sup>152</sup>

- das System die festgelegten Anforderungen erfüllt,
- das System mit den vorhandenen Systemplattformen und Anwendungssystemen kompatibel ist,
- der Entwicklungsprozess korrekt und fehlerfrei erfolgt ist,
- die Dokumentation vollständig ist und
- IT-Sicherheitsmaßnahmen vorhanden sind.

Durch die Abnahme wird die Validierung des Softwareproduktes sichergestellt. Daher sollte das Abnahmeverfahren dokumentiert werden und einem definierten und zuvor geplanten Ablauf folgen.

Durch die Festlegung der Einführungsstrategie werden der Zeitpunkt und der Zeitraum der Umstellung sowie die gewählte Vorgehensweise zur Einführung der Software bestimmt. Die Einführung kann stufenweise entweder mit Teilen des neuen Anwendungssystems (Step-by-Step-Strategie), an einem bestimmten Stichtag durch die Beendigung des Altsystems und sofortige Inbetriebnahme des neuen Anwendungssystems (Big-Bang-Strategie) sowie als Parallellauf unter gleichzeitiger und zeitlich begrenzter Fortführung des alten Systems erfolgen.<sup>153</sup>

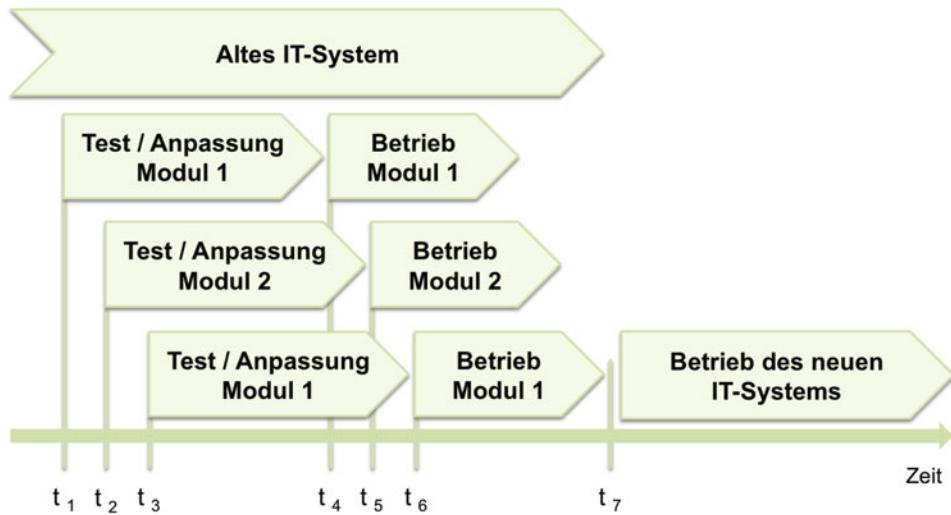
Bei einer stufenweisen Einführung werden schrittweise einzelne Funktionsbereiche oder Abteilungen mit der neuen Softwarelösung produktiv genommen, während andere Prozessbereiche mit dem alten System weiterarbeiten. Somit können Mitarbeiterressourcen eingespart und die neuen Systemfunktionen leichter erlernt werden. Ein weiterer wesentlicher Vorteil dieser Einführungsstrategie besteht in der Berücksichtigung des Sicherheitsaspekts. Einen Nachteil der Einführungsstrategie bildet die zusätzliche Programmierung von Schnittstellen, die nur für kurze Zeit während der Umstellungsphase verwendet werden.<sup>154</sup> Abbildung 5.45 stellt die Step-by-Step-Einführung nochmals grafisch dar.

Bei der Big-Bang-Strategie erfolgt der Betrieb eines neuen Anwendungssystems schlagartig zu einem bestimmten Zeitpunkt. Gleichzeitig wird das bestehende alte System

<sup>152</sup> Vgl. Stahlknecht und Hasenkamp 2005.

<sup>153</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 319.

<sup>154</sup> Vgl. Becker und Schütte 2004, S. 184.



**Abb. 5.45** Step-by-Step-Einführung. (Eigene Erstellung, in Anlehnung an Abts und Mülder 2004, S. 331)

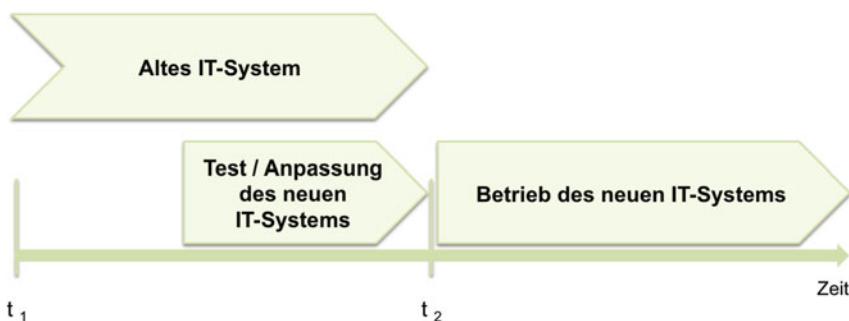
dadurch abgelöst. Im Vergleich zur stufenweisen Einführung ist der potenziell erzielbare Nutzen bei der Big-Bang-Strategie höher. Da die Erstellung von temporären Schnittstellen sowie der Aufwand für prozessübergreifende Tätigkeiten entfällt, können kürzere Einführungszeiträume realisiert werden. Aufgrund des hohen Zeitdrucks können jedoch Fehler bei der Umstellung auftreten und dadurch die Akzeptanz der zukünftigen Nutzer negativ beeinflusst werden. Des Weiteren ist das Einführungsrisiko deutlich höher als bei der Step-by-Step-Strategie, da der Umfang des Projektes höhere Anforderungen in Bezug auf die Beherrschung der Interdependenzen stellt.<sup>155</sup> Nachfolgende Abb. 5.46 stellt die Big-Bang-Einführung grafisch dar.

Für die Step-by-Step- als auch für die Big-Bang-Strategie bestehen die Aufgaben der Einführung in einer Vorbereitung des neuen Softwaresystems für den Echtbetrieb, der Installation sowie der Einspeisung notwendiger Daten in die neue Anwendung. Hinsichtlich der hierbei vorliegenden Daten kann es sich

- um die erstmalige Einrichtung von Daten, bspw. die Übernahme von Kundendaten, die bisher in Karteien geführt wurden, oder
- um die Migration von Datenbeständen, bspw. bei der Umorganisation von einer Datei zu einer Datenbankorganisation, handeln.<sup>156</sup>

<sup>155</sup> Vgl. Becker und Schütte 2004, S. 184.

<sup>156</sup> Vgl. Stahlknecht und Hasenkamp 2005, S. 319.



**Abb. 5.46** Big-Bang-Einführung. (Eigene Erstellung, in Anlehnung an Abts und Mülder 2004, S. 331)

Liegen bereits größere und umfangreichere Datenbestände vor, sollte vor der Umstellung ein Konzept zur Datenübernahme erarbeitet werden, sodass der operative Betrieb unmittelbar nach der Einführung der neuen Software beginnen kann.<sup>157</sup>

Spätestens mit dem Abschluss der Umstellung auf das neue Anwendungssystem sollten alle zukünftigen Anwender bereits eine Einweisung erhalten haben. Die hierzu notwendigen Schulungsmaßnahmen sind somit zu einem frühen Zeitpunkt innerhalb der Einführungsphase einzuplanen. Um die Umstellungsphase für die Benutzer so kurz wie möglich zu halten, sollten in den Schulungen grundlegende Funktionen und Handlungsschritte der neuen Software sowie wichtige Unterschiede zur Vorgängeranwendung aufgezeigt werden.<sup>158</sup> Die Anwender müssen weiterhin die Gelegenheit erhalten, das Gelernte an ihrem Arbeitsplatz umzusetzen. Praktische Erfahrungen lassen sich bspw. durch das Arbeiten mit fiktiven Unternehmensdaten gewinnen.<sup>159</sup>

Hinsichtlich der Einführung mobiler Anwendungen hat die Forschungsgruppe „Kommunikations- und Kollaborationsmanagement“ der Universität Münster anhand einer Onlinebefragung versucht, den Status quo in deutschen Unternehmen zu erheben. Hierzu wurden insgesamt ca. 200 Manager und Entscheidungsträger befragt, wie diese maßgebliche Erfolgsfaktoren für die Integration von mobilen Anwendungen in ihrem Unternehmen einschätzen. Aus den Umfrageergebnissen konnten weiterhin zukünftige Herausforderungen für die Unternehmen abgeleitet werden, die insbesondere im Management der Endgerätevielfalt und in der Bewältigung von Sicherheitsproblemen liegen.<sup>160</sup>

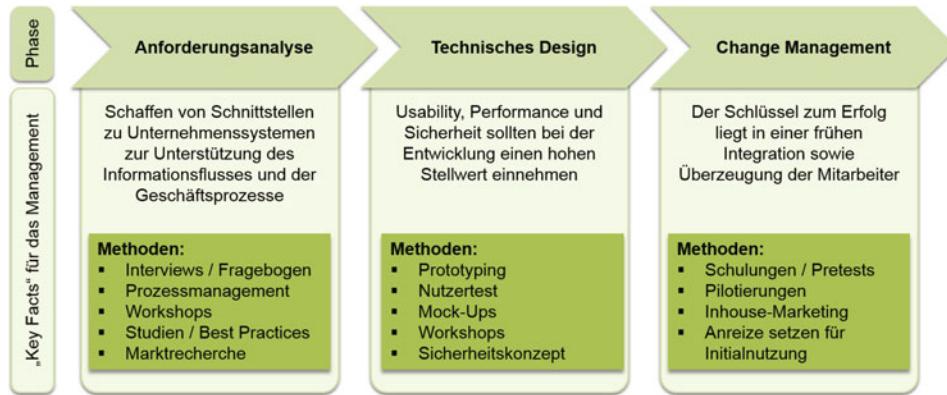
Eine Mehrheit der Befragten gab an (64 %), dass die Gestaltung des Einführungsprozesses der mobilen Anwendung von entscheidender Bedeutung für die erfolgreiche Nutzung

<sup>157</sup> Vgl. Krcmar 2005, S. 184.

<sup>158</sup> Vgl. Krcmar 2005, S. 184.

<sup>159</sup> Vgl. Abts und Mülder 2004, S. 332.

<sup>160</sup> Vgl. Stieglitz und Brockmann 2012, S. 6.



**Abb. 5.47** Gestaltung und Komponenten des Einführungsprozesses von mobilen Anwendungen in Unternehmen. (Eigene Erstellung, in Anlehnung an Stieglitz und Brockmann 2012, S. 10)

ist.<sup>161</sup> Da mobile Anwendungen im Vergleich zu anderen Unternehmensanwendungen oftmals als unterstützende oder ergänzende Dienste eingeführt werden und damit eher freiwillig von der Belegschaft angenommen werden müssen, kann eine geringe Akzeptanz zu einer Nutzungsverweigerung führen. Damit etablierte Verhaltensweisen somit nicht bestehen bleiben, müssen die zukünftigen Anwender bereits bei der Planung und Gestaltung des Einführungsprozesses mit einbezogen werden, damit diese die Vorteile identifizieren können und positiv gegenüber der Nutzung der neuen Anwendung eingestellt sind.<sup>162</sup>

In der bereits angesprochenen Studie konnte in Bezug auf den Einführungsprozess festgestellt werden, dass bei 79 % der befragten Unternehmen die frühe Durchführung von Pretests mit der mobilen Anwendung ein wichtiger Bestandteil der Einführung sei. Als weitere wichtige Faktoren bei der Einführung wurden die Vermittlung des Mehrwertes der Applikation (73 %) sowie die Einbeziehung der Mitarbeiter bei der Konzeption der Anforderung (71 %) genannt.<sup>163</sup> Ein Vorschlag für die Gestaltung und Komponenten des Einführungsprozesses von mobilen Anwendungen in Unternehmen wird in Abb. 5.47 gegeben.

## 5.6.2 Vertriebsmöglichkeiten von mobilen Applikationen im B2C-Bereich

Der Vertrieb mobiler Anwendungen hat sich seit der Veröffentlichung des ersten iPhones im Jahre 2007 und dem damit gleichzeitigen Auftreten erster Applikationen für mobile Endgeräte national als auch international als lukratives Geschäftsmodell entwickelt.

<sup>161</sup> Vgl. Stieglitz und Brockmann 2012, S. 10.

<sup>162</sup> Vgl. Stieglitz und Brockmann 2012, S. 10.

<sup>163</sup> Vgl. Stieglitz und Brockmann 2012, S. 10.



**Abb. 5.48** Geschäftsmodell für den Vertrieb mobiler Anwendungen über einen Online-Store.  
(Eigene Erstellung, in Anlehnung an Holzer und Ondrus 2009, S. 57)

Im Vergleich zu stationären Desktopanwendungen stehen für die Verteilung mobiler Anwendungen nur eine begrenzte Anzahl an offiziellen Vertriebswegen zur Verfügung. Im Fokus der Entwickler stehen primär die plattformeigenen, endverbraucherorientierten Marketplaces, wie bspw. der PlayStore für Android-Anwendungen, der App-Store für iOS-Anwendungen sowie der Windows Phone Marketplace für Windows-Phone-Anwendungen. Die Inhalte auf den Online-Marktplätzen von Apple und Microsoft werden exklusiv vertrieben, d. h. Entwicklern, die ihre Anwendung für die Betriebssysteme iOS oder Windows Phone erstellt haben, stehen im Gegensatz zu Android-Entwicklern keine weiteren Vertriebsmöglichkeiten zur Verfügung.<sup>164</sup>

Allgemeingültig für den Vertrieb mobiler Applikationen über Online-Marktplätze ist das in Abb. 5.48 dargestellte Geschäftsmodell, das die in Verbindung mit der Entwicklung und dem Vertrieb der mobilen Anwendung beteiligten Parteien sowie die damit verbundenen Erlösquellen verdeutlicht. Dem Geschäftsmodell liegt folgendes Vorgehen zugrunde:<sup>165</sup>

1. Der Softwareentwickler programmiert mit Hilfe verschiedener Entwicklungswerkzeuge die mobile Anwendung. Die fertige Anwendung wird anschließend durch den Entwickler auf einem Online-Store veröffentlicht und potenziellen Nutzern zur Verfügung gestellt. Für die Veröffentlichung muss der Entwickler meist eine Gebühr an den Betreiber entrichten.
2. Besucher des Online-Stores wählen aus einer Vielzahl an gelisteten mobilen Anwendungen eine gewünschte Applikation aus und laden diese auf ihr entsprechendes mobiles Endgerät, bspw. Smartphone oder Tablet-PC.
3. Für diesen Download ist im Falle einer kostenpflichtigen Anwendung eine Zahlung an den Betreiber des Online-Stores zu entrichten.
4. Ein kleiner Teil dieser Zahlung muss der Entwickler dem Betreiber für die Bereitstellung der Anwendung in dem Online-Store sowie für die angefallenen Transaktionskosten überlassen.

Damit eine mobile Anwendung über einen Online-Marktplatz veröffentlicht werden kann, muss der Entwickler zuvor einen Zugang zum Online-Store beantragen und einrichten.

<sup>164</sup> Vgl. Koppay 2012, S. 167.

<sup>165</sup> Vgl. Holzer und Ondrus 2009, S. 56.

Für diese Entwicklerzugänge fallen Kosten an, die je nach Betreiber und Leistungsumfang unterschiedlich sein können. Ein Zugang für den Play-Store von Google kann für eine einmalige Gebühr von 25 US \$ erworben werden.<sup>166</sup> Im Vergleich hierzu müssen jährlich 99 US \$ für einen Zugang zum Apple App-Store entrichtet werden.<sup>167</sup> Neben diesen Entwicklerzugängen bietet Apple für die jährliche Zahlung von 299 US \$ spezielle Unternehmenskonten an, über die innerhalb des Unternehmens eigene kostenlose als auch kostenpflichtige Anwendungen günstiger verteilt werden können.<sup>168</sup>

Bevor eine fertige mobile Anwendung im Katalog eines Online-Stores gelistet wird, müssen verschiedene Anforderungen an und Restriktionen für die Applikation erfüllt sein. Applikationen für den App-Store von Apple und den Windows-Phone-Store von Windows müssen hierzu einen Zertifizierungsprozess durchlaufen, in dem die Applikationen nach bestimmten Qualitätskriterien geprüft und mit einer technischen Signatur versehen werden. So werden einerseits die Anwendungen auf Fehler und Mängel überprüft und andererseits ein hohes Sicherheits- und Qualitätsniveau gewährleistet. Für die Veröffentlichung auf dem Google-Play-Store ist das Zertifikat vom Anbieter der mobilen Applikation bereitzustellen, der auch für die Sicherstellung der Qualität verantwortlich ist.<sup>169</sup>

### **5.6.3 Werkzeuge zur Einführungs- und Veröffentlichungsphase**

Wie zu Beginn des Kapitels beschrieben, werden unter der Einführungs- und Veröffentlichungsphase alle für die Umsetzung des Sollkonzeptes und der Inbetriebnahme der neuen Software benötigten Aufgaben und Tätigkeiten subsumiert. Hierzu stehen dem Einführungs- und Veröffentlichungsprozess verschiedene unterstützende Werkzeuge und Methoden zur Verfügung (vgl. Abb. 5.47). Nachfolgend werden zunächst die in Abschn. 5.6.1 angesprochenen Benutzertests und -schulungen genauer betrachtet. Daran anschließend werden für die Vermarktung von mobilen Applikationen nützliche Werbe- und Marketingmöglichkeiten erörtert.

#### **5.6.3.1 Benutzertests und Schulungen**

Unabhängig von der gewählten Einführungsstrategie (vgl. Abschn. 5.6.1) sollte die neue Software vor der Inbetriebnahme ausreichend und unter Alltagsbedingungen getestet werden. Mit diesen Benutzertests wird vor dem eigentlichen Produktivbetrieb überprüft, ob das neue Anwendungssystem die betrieblichen Anforderungen ordnungsgemäß erfüllt. In diesem Zusammenhang werden realistische Aufgaben am System bearbeitet und hierbei auftretende Fehler im System, die bspw. die Ausführung bestimmter Aufgaben erschweren oder behindern, identifiziert und für die spätere Ergebnispräsentation dokumentiert.<sup>170</sup>

---

<sup>166</sup> Vgl. Google 2013.

<sup>167</sup> Vgl. Apple 2013a.

<sup>168</sup> Vgl. Apple 2013b.

<sup>169</sup> Vgl. Knüpffer et al. 2013, S. 26.

<sup>170</sup> Vgl. Prümper 2013a.

Die Durchführung eines ausführlichen und umfangreichen Benutzertests benötigt eine gute Organisation und ist mit einem hohen Zeitaufwand verbunden. In Bezug auf das Vorgehen zur Durchführung der Tests bietet sich folgender Ablauf an:<sup>171</sup>

1. Testteam zusammenstellen
2. Testaufgaben festlegen
3. Tests durchführen
4. Ergebnisse dokumentieren

In einem ersten Schritt sollte ein Testteam zusammengestellt werden, das für eine optimale Durchführung der Tests aus mindestens zwei an der Entwicklung der Anwendung beteiligten Mitarbeitern bestehen sollte. Die Mitarbeiter sollten weiterhin an der Planung der Einführung beteiligt gewesen sein. Dem Testteam sollte zudem eine Person aus dem Unternehmen zur Seite stehen, die mit dem Umgang und den Funktionen des alten Systems vertraut ist und somit mögliche Fragen des Testteams beantworten kann. Das Testteam sollte in einem weiterführenden Schritt Testaufgaben definieren, die den gesamten Anwendungsbereich der Software abdecken. Hierzu zählen neben Standardaufgaben, die oft und von vielen Benutzern gleichzeitig bearbeitet werden, auch Sonder- und Ausnahmefälle, deren Bearbeitung besonders weitreichende Folgen haben kann.<sup>172</sup>

Die Durchführung der Tests durch das Testteam sollte zu einem Zeitpunkt erfolgen, an dem die neue Software möglichst weitgehend an das Unternehmen angepasst wurde und dennoch mögliche Änderungen an der Anwendung vorgenommen werden können. Während des Tests sollten die durchgeführten Testläufe dokumentiert und die identifizierten Mängel protokolliert werden. Folgende Auswahl typischer Problemfelder kann im Zusammenhang mit der Durchführung von Benutzertests erhoben werden:<sup>173</sup>

- Funktionen fehlen
- automatische Berechnungen werden nicht durchgeführt
- Eingabefelder, die nicht benötigt werden
- überflüssige Bearbeitungsschritte
- Funktionen sind schwierig zu finden
- Buttons oder Eingabefelder stehen an der falschen Stelle
- unverständliche Ergebnisdarstellungen

Die Ergebnisse der durchgeführten Benutzertests werden abschließend zusammengefasst und im Team besprochen. Wurden alle Mängel beseitigt, können die Schulungsmaßnahmen für die Mitarbeiter vorbereitet und geplant werden. Hierbei stellt sich zunächst die

<sup>171</sup> Vgl. Prümper 2013a.

<sup>172</sup> Vgl. Prümper 2013a.

<sup>173</sup> Vgl. Prümper 2013b.

Frage, welche Mitarbeiter geschult werden müssen und wie lange die Schulungsmaßnahmen andauern sollen. Die Planung und Durchführung der Schulungsmaßnahmen sollte folgende Schritte beinhalten:<sup>174</sup>

1. Inhaltlichen und personellen Qualifizierungsbedarf ermitteln
2. Schulungsort und -termine festlegen
3. Schulung durchführen
4. Übungsmöglichkeiten bereitstellen

Der notwendige Qualifizierungsbedarf kann aus den Ergebnissen der Anforderungsanalyse abgeleitet werden. Die bei der Anforderungserhebung durchgeführte Definition und Beschreibung wichtiger Geschäftsprozesse liefert hierzu wertvolle Hinweise über den personellen als auch den inhaltlichen Schulungsbedarf. Durch die Geschäftsprozessanalyse wurden einerseits die Fachbereiche und Arbeitsplätze festgelegt, an denen die neue Software eingesetzt werden soll, und andererseits die Geschäftsprozesse beschrieben, die durch das neue Anwendungssystem unterstützt werden sollen. Darüber hinaus müssen die Inhalte der Schulungsmaßnahme an den in der Geschäftsprozessanalyse definierten Aufgaben- und Tätigkeitsbereichen, die zukünftig mit Hilfe des neuen Softwaresystems bearbeitet werden sollen, ausgerichtet werden. Wurden in der Analyse der Geschäftsprozesse unterschiedliche Aufgabenbereiche eingegrenzt bzw. wird die Software für die Bearbeitung verschiedener Aufgaben eingesetzt, sollten die Mitarbeiter dieser Arbeitsbereiche jeweils differenzierte Weiterbildungsmaßnahmen erhalten.<sup>175</sup>

In einem anschließenden Schritt sollte der Schulungsort festgelegt sowie der Zeitraum für die Schulungsmaßnahme terminiert werden. Die Termine für die anfallenden Schulungen sollten möglichst nahe am Beginn des Echtbetriebs liegen, damit die neu gewonnenen Kenntnisse unmittelbar angewendet werden können. Für die zeitlich freigestellten Mitarbeiter sollten entsprechende Vertretungen verfügbar sein. In der Regel sind Inhouseschulungen günstiger und insbesondere für kleine und mittlere Unternehmen praktikabler, da die Kosten für einen Schulungsraum sowie anfallende Reise- und Unterbringungskosten eingespart werden können. Externe Schulungen sind dann zu empfehlen, wenn im Betrieb notwendige Räumlichkeiten fehlen, die Schulungsmaßnahme nur mit Störungen des Arbeitsalltags verbunden wäre oder die Freistellung der Mitarbeiter nur begrenzt möglich ist.<sup>176</sup> In Bezug auf die Einführung mobiler Anwendungen kann eine Mischung aus interner und externer Schulung empfohlen werden. Wird bspw. eine mobile Anwendung für den Außendienst eingeführt, können im Rahmen einer Inhouseschulung zunächst die grundlegenden Funktionen vermittelt und anschließend die einzelnen Abläufe der Applikation anhand realer Bedingungen vertieft werden.

---

<sup>174</sup> Vgl. Prümper 2013c.

<sup>175</sup> Vgl. Prümper 2013c.

<sup>176</sup> Vgl. Prümper 2013c.

Unabhängig davon, ob die Schulung intern oder extern durchgeführt werden soll, muss für die erfolgreiche Durchführung der Weiterbildung sichergestellt werden, dass alle notwendigen Ressourcen zur Verfügung stehen, d. h.:<sup>177</sup>

- die Räumlichkeit muss genügend Platz für die Schulungsteilnehmer bieten,
- die Räumlichkeit muss mit ausreichenden Präsentationsmaterialien, wie z. B. Beamer, Flipchart, Metaplanwände usw., ausgestattet sein,
- im Seminarraum müssen ausreichend Stühle, Tische, Steckdosen usw. vorhanden sein,
- auf den Computern bzw. mobilen Endgeräten sollte die zu schulende Anwendung installiert sein und
- es müssen realistische Nutzerprofile und Testdaten auf den Computern bzw. mobilen Endgeräten vorhanden sein.

Im Sinne des lebenslangen Lernens empfiehlt sich nach dem Produktivstart der Einsatz sinnvoller Übungsmöglichkeiten, damit die Benutzer das Gelernte nicht vergessen, vertiefen und ggf. weiter ausbauen können. In der Praxis haben sich hierzu elektronische Lernsysteme etabliert, auf denen eine Trainingsversion der Software installiert ist, die von den Benutzern für nachträgliche Übungen im Umgang mit der Software verwendet werden können.<sup>178</sup>

#### 5.6.3.2 Werbe- und Marketingmöglichkeiten

Mit der Umsetzung von Werbe- und Marketingmöglichkeiten soll die Realisierung der zuvor in der Planungs- und Konzeptionsphase festgelegten Marketingziele erreicht werden. Jedoch sind die Möglichkeiten mobiler Marketingstrategien, in Bezug auf die sinnvolle Schaltung gezielter Werbemaßnahmen in den Online-Stores, recht eingeschränkt. Auf das Ranking und die Darstellung der mobilen Anwendungen in den einzelnen Online-Stores kann nur bedingt Einfluss genommen werden. Insbesondere das Ranking der mobilen Applikation wird durch deren Downloadanzahl, Qualität und Bewertung durch die Nutzer beeinflusst.<sup>179</sup> Laut einer Studie von Deloitte MCS Limited aus dem Jahr 2011 erreichen nur ca. 20 % alle mobilen Applikationen nennenswerte Downloadzahlen.<sup>180</sup>

Unternehmen bzw. Entwickler mobiler Anwendungen stehen somit vor der Herausforderung, ihre Produkte bestmöglich zu präsentieren und Nutzer von der Applikation zu begeistern. Wichtige Kriterien bezüglich der Präsentation im Online-Stores sind der Name der Anwendung sowie damit verbundene Schlüsselwörter (Keywords) und die allgemeine Beschreibung über den Leistungsumfang der Anwendung. Im bestmöglichen Fall ist der Name der mobilen Anwendung einzigartig, um nicht in der breiten Masse unterzugehen oder mit einer anderen gleichartigen Applikation verwechselt zu werden. Durch

<sup>177</sup> Vgl. Prümper 2013d.

<sup>178</sup> Vgl. Prümper 2013c.

<sup>179</sup> Vgl. Knüppfer et al. 2013, S. 26.

<sup>180</sup> Vgl. Deloitte 2011, S. 2.

den Einsatz von Schlüsselwörtern soll die Funktionalität und der Anwendungsbereich der Anwendung besser herausgestellt werden. Gleichzeitig erleichtern die Schlüsselwörter das Auffinden der Applikation im Online-Store. Die Entscheidung für den Download einer Applikation ist weiterhin davon abhängig, inwieweit die Beschreibung der Anwendung das Interesse des Nutzers geweckt hat. Die Beschreibung sollte auf wesentliche Leistungsmerkmale und Anwendungsbereiche beschränkt sein und einen ersten Eindruck über den Funktionsumfang vermitteln.<sup>181</sup>

Diese hauptsächlich textbasierten Werbemaßnahmen können durch zusätzliche visuelle Werbemittel, wie z. B. dem Icon der Anwendung oder Abbildungen der Applikation, unterstützt werden. Das Icon sollte ebenfalls wie der Name der Anwendung einen Wiedererkennungswert aufweisen und in das Gesamtkonzept der Applikation passen. Ein weiterer wichtiger Faktor für die Präsentation im Online-Store ist die Darstellung von Screenshots der Anwendung. Diese sollten bestenfalls die Leistung der Anwendung demonstrieren und dadurch den Nutzer zum Ausprobieren bewegen. Der Google-Play-Store bietet zudem die Möglichkeit an, ein kurzes Video potenziellen Nutzern zur Verfügung zu stellen. Dadurch kann der Mehrwert der mobilen Anwendungen besser präsentiert werden.<sup>182</sup>

Die Schaltung zusätzlicher Werbung neben den Online-Stores kann dazu führen, in den Markt besser einzudringen und dadurch die kritische Nutzermasse und Downloadzahl schneller zu erreichen. In diesem Zusammenhang können moderne Werbemöglichkeiten, wie z. B. die Schaltung von Werbung über soziale Netzwerke, als auch klassische Werbemaßnahmen, wie bspw. Print-, Radio- oder Fernsehwerbung, zum Einsatz kommen.<sup>183</sup> Eine Auswahl möglicher Marketingmaßnahmen für die Bewerbung einer mobilen Anwendung wird nachfolgend gegeben:

- **Unternehmenswebsite:** Aktuell besitzen eine Vielzahl an Unternehmen eigene Websites, die für die Präsentation des Unternehmens sowie deren Produkte und Leistungen verwendet werden. Die Positionierung von Werbemaßnahmen auf der eigenen Website stellt eine kostengünstige Variante dar, die schnell und ohne großen Zeitaufwand zu realisieren ist.
- **Newsletter:** Newsletter werden dazu eingesetzt, die Kunden oder potenzielle Neukunden eines Unternehmens zeitnah mit aktuellen Informationen zu versorgen. Ist der Versand von Newslettern bereits eingerichtet, stellt dies ebenfalls eine kostengünstige Möglichkeit dar, eine breite Masse über die neue mobile Applikation zu informieren.
- **Soziale Netzwerke:** Die Schaltung von Werbung auf sozialen Netzwerken hat den Vorteil, neue Zielgruppen zu erschließen und damit die Informationen über die neue mobile Applikation auch an Personen zu richten, die bisher noch nicht in Verbindung mit dem Unternehmen standen.

---

<sup>181</sup> Vgl. Knüpffer et al. 2013, S. 26.

<sup>182</sup> Vgl. Knüpffer et al. 2013, S. 27.

<sup>183</sup> Vgl. Knüpffer et al. 2013, S. 27.

- **Radio- und Fernsehwerbung:** Die Vermarktung der mobilen Anwendung über das Radio oder das Fernsehen stellt eine kostenintensive Variante dar. Die Vermarktung über diese Medien bietet sich insbesondere dann an, wenn eine mobile Applikation bereits einen gewissen Bekanntheitsgrad erreicht hat und durch die Werbemaßnahmen eine breite Masse auf Änderungen oder Verbesserungen hingewiesen werden soll.
- **Virales Marketing:** Neben den bisher genannten Werbemaßnahmen besteht weiterhin immer die Möglichkeit, dass eine mobile Anwendung durch Weiterempfehlungen bereits begeisterter Nutzer beworben wird.

Bei der Schaltung zusätzlicher Werbung und der Wahl einer geeigneten Werbemaßnahme sollte ein zielgruppenorientiertes Marketing angestrebt werden, um die entstehenden Kosten zu minimieren und gleichzeitig einen maximalen Nutzen zu erreichen.

---

## Literatur

- Abts, D., Mülder, W.: Grundkurs Wirtschaftsinformatik, Eine kompakte und praxisorientierte Einführung, 5. Aufl. Vieweg & Sohn, Wiesbaden (2004)
- Adizes, I.: Corporate lifecycles: how and why corporations grow and die and what to do about it. Prentice Hall, New York (1988)
- Adobe Systems Inc.: Supported features. <http://phonegap.com/about/feature/> (2013). Zugegriffen 10 Okt. 2013
- Aichele, C.: Intelligentes Projektmanagement. W. Kohlhammer, Stuttgart (2006)
- Albert, K., Stiller, M.: Der Browser als mobile Plattform der Zukunft. Die Möglichkeiten von HTML5-Apps. Chancen und Grenzen der Entwicklung mobiler Anwendungen mit Hilfe von Web-standards. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 147–160. Springer, Berlin (2012)
- Allweyer, T.: BPMN 2.0, Business Process Model and Notation. Einführung in den Standard für die Geschäftsprozessmodellierung, 2. Aufl. Books on Demand, Norderstedt (2009)
- Ammenwerth, E., Haux, R.: IT-Projektmanagement in Krankenhaus und Gesundheitswesen. Einführendes Lehrbuch und Projektleitfaden für das taktische Management von Informationssystemen. Schattauer, Stuttgart (2005)
- Apple Inc.: Apple developer programs. <https://developer.apple.com/programs/> (2013a). Zugegriffen 9 Okt. 2013
- Apple Inc.: iOS developer enterprise program. <https://developer.apple.com/programs/ios/enterprise/> (2013b). Zugegriffen 9 Okt. 2013
- Balzert, H.: Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering, 3. Aufl. Spektrum Akademischer, Heidelberg (2009)
- Balzert, H.: Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb, 3. Aufl. Spektrum Akademischer, Heidelberg (2011)
- Becker, J., Schütte, R.: Handelsinformationssysteme, Redline Wirtschaft, 2. Aufl. MI-Wirtschaftsbuch, Frankfurt a. M. (2004)
- Berekoven, L., Eckert, W., Ellenrieder, P.: Marktforschung. Methodische Grundlagen und praktische Anwendung, 12. Aufl. Gabler Wiesbaden (2009)

- Biskup, H., Fischer, T.: Vorgehensmodelle - Versuch einer begrifflichen Einordnung – Vorstellung erster Ergebnisse einer Arbeitsgruppe der Fachgruppe 5.11, Gesellschaft für Informatik e. V. (GI), Bonn, (2003)
- Blinn, N., Nüttgens, M., Schlicker, M., Thomas, O., Walter, P.: Lebenszyklusmodelle hybrider Wertschöpfung. Modellimplikation und Fallstudie am Beispiel des Maschinen- und Anlagenbaus. In: Bichler, M., Hess, T., Krcmar, H., et al. (Hrsg.) Multikonferenz Wirtschaftsinformatik, S. 711–722. GITO-Verlag, Berlin (2010)
- Boehm, W.B.: A spiral model of software development and enhancement. *IEEE Comput.* **21**(5), S. 61–72 (Ausgabe, Redondo Beach) (1988)
- Brandt-Pook, H., Kollmeier, R.: Softwareentwicklung kompakt und verständlich. Wie Softwaresysteme entstehen. Vieweg+Teubner, Wiesbaden (2008)
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom): Zeitenwende auf dem Handy-Markt, Berlin. [http://www.bitkom.org/de/markt\\_statistik/64046\\_71243.aspx](http://www.bitkom.org/de/markt_statistik/64046_71243.aspx) (2012). Zugegriffen 6 Aug. 2013
- Bunse, C., Knethen von, A.: Vorgehensmodelle kompakt, 2. Aufl. Spektrum Akademischer, Heidelberg (2008)
- Capgemini: World quality report 2012–2013. [http://www.de.capgemini.com/sites/default/files/resource/pdf/World\\_Quality\\_Report\\_2012\\_-\\_2013.pdf](http://www.de.capgemini.com/sites/default/files/resource/pdf/World_Quality_Report_2012_-_2013.pdf) (2012). Zugegriffen 4 Dez. 2013
- Deloitte MCS Limited: Killer apps? Appearance isn't everything. <http://www.deloitte.com/assets/Dcom-UnitedKingdom/Local%20Assets/Documents/Services/Consulting/uk-con-killer-apps.pdf> (2011). Zugegriffen 9 Okt. 2013
- Diederichs, H.: Komplexitätsreduktion in der Softwareentwicklung. Ein systemtheoretischer Ansatz. Books On Demand, Norderstedt (2004)
- Dornberger, R.: Innovationsmanagement. Ideengenerierung. [http://web.fhnw.ch/personenseiten/rolf.dornberger/Documents/Lectures/im/IM3\\_Ideengenerierung.pdf](http://web.fhnw.ch/personenseiten/rolf.dornberger/Documents/Lectures/im/IM3_Ideengenerierung.pdf) (2006). Zugegriffen 6 Okt. 2013
- Dunne, S.: Phonegap from scratch. Introduction. <http://mobile.tutsplus.com/tutorials/phonegap-phonegap-from-scratch/> (2012). Zugegriffen 10 Okt. 2013
- Erlenkötter, H.: C Programmieren vom Anfang an, 10. Aufl. Rowohlt Taschenbuch, Hamburg (2005)
- Felker, D.: Android Apps Entwicklung. Wiley-VCH, Weinheim (2011)
- Feyhl, A.W.: Management und Controlling von Softwareprojekten. Software wirtschaftlich auswählen, einsetzen und nutzen, 2. Aufl. Gabler, Wiesbaden (2004)
- Franz, K.: Handbuch zum Testen von Web-Applikationen. Springer, Berlin (2007)
- Gall, H., Hauswirth, M., Klösch, R.: Objektorientierte Konzepte in Smalltalk, C++, Objective-C, Eiffel und Modula-3. Informatik Spektrum. **4**, S. 195–202 (1995)
- Gerlicher, A.R.S.: Die Grenzen des Browsers durchbrechen. Hybride Anwendungsentwicklung für mobile Endgeräte. In: Smart Mobile Apps, Verclas, S. Linnhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 161–177. Springer, Berlin (2012)
- Gernert, C.: Agiles Projektmanagement. Risikogesteuerte Softwareentwicklung. Carl Hanser, München (2003)
- Google Inc.: Entwickler-Registrierung. [https://support.google.com/googleplay/android-developer/answer/113468?hl=de&ref\\_topic=2365624](https://support.google.com/googleplay/android-developer/answer/113468?hl=de&ref_topic=2365624) (2013). Zugegriffen 09 Okt. 2013
- Grande, M.: 100 Minuten für Anforderungsmanagement. Kompaktes Wissen nicht nur für Projektleiter und Entwickler. Vieweg+Teubner, Wiesbaden (2011)
- Grechenig, T., Bernhart, M., Breiteneder, R., Kappel, K.: Softwaretechnik. Mit Fallbeispielen aus realen Entwicklungsprojekten. Pearson Studium, München (2010)
- Guérin, C.: Introduction to cross-platform mobile development with Appcelerator Titanium. [http://l3i.univ-larochelle.fr/IMG/pdf/cours\\_titanium.pdf](http://l3i.univ-larochelle.fr/IMG/pdf/cours_titanium.pdf) (2012). Zugegriffen 10 Okt. 2013

- Henning, P.A., Hoffmann, D.W., Vogelsang, H.: Grundlagen der Programmiersprachen. In: Henning, P.A., Vogelsang, H. (Hrsg.) Handbuch Programmiersprachen. Softwareentwicklung zum Lernen und Nachschlagen, S. 9–59. Carl Hanser, München (2007)
- Highsmith, J.: Agile project management: creating innovative products, 2. Aufl. Addison-Wesley Longman, Amsterdam (2009)
- Holzer, A., Ondrus, J.: Trends in mobile application development. In: Hesselman, C., Giannelli, C. (Hrsg.) Mobile wireless middleware, operationg systems, and applications – workshops, S. 55–64. Springer, Berlin (2009)
- Horn, C., Kerner, I.O., Forbig, P. (Hrsg.): Lehr- und Übungsbuch Informatik. Grundlagen und Überblick, 3. Aufl. Carl Hanser, München (2003)
- Hruschka, P., Rupp, C., Starke, G.: Agility kompakt: Tipps für erfolgreiche Systementwicklung, 2. Aufl. Spektrum Akademischer, Heidelberg (2009)
- IT Wissen: Framework. <http://www.itwissen.info/definition/lexikon/Framework-framework.html> (2013). Zugegriffen 10 Okt. 2013
- Kloss, J.H.: Android-Apps. Mobile Anwendungen entwickeln mit App Inventor. Markt + Technik, München (2011)
- Knüpfffer, W., Fritsch, M., Matthes, A.: Von der Idee zur eigenen App. Ein praxisorientierter Leitfaden für Unternehmer mit Checkliste, eBusiness-Lotse Metropolregion Nürnberg, Juni. [http://www.nik-nbg.de/fileadmin/redaktion/Hinterlegte\\_Dokumente\\_Homepage/Leitfaden\\_Von\\_der\\_Idee\\_zur\\_eigenen\\_App.pdf](http://www.nik-nbg.de/fileadmin/redaktion/Hinterlegte_Dokumente_Homepage/Leitfaden_Von_der_Idee_zur_eigenen_App.pdf) (2013). Zugegriffen 9 Okt. 2013
- Koord, Y., Krauter, V.: Überblick Vorgehensmodelle im Projektmanagement. [http://winfwiki.wifom.de/index.php/%C3%9Cberblick\\_Vorgehensmodelle\\_im\\_Projektmanagement](http://winfwiki.wifom.de/index.php/%C3%9Cberblick_Vorgehensmodelle_im_Projektmanagement) (2009). Zugegriffen 4 Okt. 2013
- Koppay, H.: Entwicklung und Vermarktung von Handy-Apps. Einstieg in die Welt der mobilen Applikationen. Disserta, Hamburg (2012)
- Kotz, J.: Erfolgreich Visual Basic 2010 programmieren. Addison-Wesley, München (2011)
- Krcmar, H.: Informationsmanagement, 4. Aufl. Springer, Berlin (2010)
- Kruchten, P.: The rational unified process an introduction, 3. Aufl. Pearson Education, Boston (2004)
- Kuassi, L., Bischel, M.: Anwendungssicht mobiler Geschäftsanwendungen. In: Vercales, S., Linhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 125–147. Springer, Berlin (2012)
- Lassmann, W.: Wirtschaftsinformatik. Nachschlagewerk für Studium und Praxis. Gabler, Wiesbaden (2006)
- Litke, H.-D.: Projektmanagement. Methoden, Techniken, Verhaltensweisen. Evolutionäres Projektmanagement, 5. Aufl. Carl Hanser, München (2007)
- Meck, U.: Management komplexer Problemsituationen. Ziele setzen und Informationen nutzbar machen. Passavia Druckservice, Passau (2009)
- Mertens, P., Bodendorf, F., König, W., et al.: Grundzüge der Wirtschaftsinformatik, 9. Aufl. Springer, Berlin (2005)
- Microsoft Corp.: Visual Studio 2013, online. <http://www.microsoft.com/visualstudio/deu/visual-studio-2013> (2013). Zugegriffen 9 Okt. 2013
- Object Management Group: Business process model and notation (BPMN) version 2.0, object management group, BPMN Specification, document number: formal/2011-01-03, Needham, (2011)
- Patzak, G.: Systemtechnik – Planung komplexer innovativer Systeme. Grundlagen, Methoden, Techniken. Springer, Berlin (1982)
- Platz, J., Schmelzer, H.J.: Projektmanagement in der industriellen Forschung und Entwicklung. Einführung anhand von Beispielen aus der Informationstechnik. Springer, Berlin (1986)

- Pohl, K.: Requirements Engineering, Grundlagen, Prinzipien, Techniken, 2. Aufl. dpunkt, Heidelberg (2008)
- Pomberger, G., Dobler, H.: Algorithmen und Datenstrukturen. Eine systematische Einführung in die Programmierung. Pearson Studium, München (2008)
- Pomberger, G., Pree, W.: Software Engineering. Architektur-Design und Prozessorientierung, 3. Aufl. Carl Hanser, München (2004)
- Prümper, C.: Benutzertests. <http://www.seikumu.com/de/inbetriebnahme/I-benutzertests/benutzertests.php> (2013a). Zugegriffen 09 Okt. 2013
- Prümper, C.: Typische Mängel einer Software. Orientierungshilfe für Benutzertests. <http://www.seikumu.com/de/dok/dok-inbetriebnahme/Typische-Maengel-Software.pdf> (2013b). Zugegriffen 09 Okt. 2013
- Prümper, C.: Benutzerschulungen. <http://www.seikumu.com/de/inbetriebnahme/II-benutzerschulung/benutzerschulungen.php> (2013c). Zugegriffen 09 Okt. 2013
- Prümper, C.: Checkliste zur Vorbereitung von Inhouse-Schulungen. <http://www.seikumu.com/de/dok/dok-inbetriebnahme/Checkliste-Inhouse-Schulungen.pdf> (2013d). Zugegriffen 09 Okt 2013
- Rinza, P.: Projektmanagement. Planung, Überwachung und Steuerung von technischen und nichttechnischen Vorhaben, 4. Aufl. Springer, Berlin (1998)
- Ruf, W., Fittkau, T.: Ganzheitliches IT-Projektmanagement. Wissen, Praxis, Anwendungen. Oldenbourg Wissenschaftsverlag, München (2008)
- Rühl, C., Schenkel, T.: Best Practices für die Entwicklung mobiler Unternehmens-Apps. <http://www.heise.de/developer/artikel/Best-Practices-fuer-die-Entwicklung-mobiler-Unternehmens-Apps-1627012.html> (2012). Zugegriffen 04 Okt. 2013
- Sadun, E.: Das iPhone Entwicklerbuch. Rezepte für Anwendungsprogrammierung mit dem iPhone SDK. Addison-Wesely, München (2009)
- Scheer, A.-W.: Wirtschaftsinformatik. Referenzmodelle für industrielle Geschäftsprozesse, 7. Aufl. Springer, Berlin (1997)
- Schmidt, G.: Informationsmanagement. Modelle, Methoden, Techniken, 2. Aufl. Springer, Berlin (1999)
- Scholz, C.: Strategische Organisation. Multiperspektivität und Virtualität, Nachdruck der, 2. Aufl. als Skript im Eigenverlag, Saarbrücken (2007)
- Sommerville, I.: Software Engineering, 6. Aufl. Pearson Studium, München (2001)
- Stahlknecht, P., Hasenkamp, U.: Einführung in die Wirtschaftsinformatik, 11. Aufl. Springer, Berlin (2005)
- Stieglitz, S., Brockmann, T.: Mobile Enterprise. Erfolgsfaktoren für die Einführung mobiler Applikationen. In: Meinhardt, S., Reich, S. (Hrsg.) Mobile Computing, HMD – Praxis der Wirtschaftsinformatik, (Heft 286), S. 6–14. dpunkt, Heidelberg (2012)
- Stolle, R., Herrmann, M.: Angebotsmanagement professionell. Erfolgreich vom Angebot bis zum Vertragsschluss. Erich Schmidt, Berlin (2006)
- Strang, T., Lichtenstern, M.: Programmierung von Smart Mobile Apps. In: Smart Mobile Apps, Verclas, S., Linnhoff-Popien, C. (Hrsg.) Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse, S. 419–429. Springer, Berlin (2012)
- Tremp, H., Ruggiero, M.: Application Engineering. Grundlagen für die objektorientierte Softwareentwicklung mit zahlreichen Beispielen, Aufgaben und Lösungen. Compendio Bildungsmedien, Zürich (2011)
- Versteegen, G.: Das V-Modell in der Praxis. Grundlagen, Erfahrungen, Werkzeuge. dpunkt, Heidelberg (2001)
- Victor, F.: Programmiersprachen. In: Schneider, U., Werner, D. (Hrsg.) Taschenbuch der Informatik, S. 197–220. 6. Aufl. Carl Hanser, München (2007)

- Vogel-Heuser, B.: Systems Software Engineering. Angewandte Methoden des Systementwurfs für Ingenieure. Oldenbourg Industrieverlag, München (2003)
- Vollendorf, M., Bongers, F.: jQuery. Das Praxisbuch. Galileo Computing, Bonn (2010)
- Werler, S.: Best Practices für die plattformübergreifende App-Entwicklung - Einer für alle. iX Developer App-Entwicklung, Ausgabe 3/2012, Heise Zeitschriftenverlag, Hannover, (2012)
- Wieczorek, H.W., Mertens, P.: Management von IT-Projekten, 4. Aufl. Springer, Berlin (2011)

Klaus Knopper

*Grundlagen und Beispiele aus Theorie und Praxis zur  
Systemsicherheit bei der Anwendung mobiler Softwaresysteme.*

## Zusammenfassung

In diesem Buchabschnitt geht es um die Sicherheit im Netzwerk von System- und Anwendersoftware auf mobilen „intelligenten“ Geräten. Es werden die Risiken aufgezeigt, die das Arbeiten in einer schwer oder gar nicht kontrollierbaren vernetzten Umgebung mit sich bringt, die potenziellen Angriffspunkte und Systemschwächen analysiert sowie Härtungs- und Sicherungsmaßnahmen erläutert.

---

## 6.1 Verbreitung mobiler IT-Systeme und Eingriff in unternehmenskritische Bereiche

Um die Jahrtausendwende zeichnete sich ein neues Zeitalter bezüglich der Nutzungsarten für IT-Systeme ab. Zuvor wurden Mainframes (ein Computer, mehrere Anwender) bereits durch Personal Computer (ein oder wenige Benutzer pro PC) zurückgedrängt, mittlerweile sind auch Notebooks als mobile Computer quantitativ im Rückzug und werden durch Tablets, Smartphones und Wearable Computing (Smartwatches, Smart Glasses) ersetzt, wodurch sich auch der Zugriff auf und die Darstellung von sensitiven Daten verändert.<sup>1</sup> Mobile Geräte sind oft in ihren Ein- und Ausgabemöglichkeiten auf eine durch

---

<sup>1</sup> Vgl. Friedewald et al. 2010.

den Formfaktor vorgegebene Bedienung über Touchscreens, Stifte, Gesten und Sprache eingeschränkt, und sie sind im Regelfall permanent mit dem lokalen Netzwerk oder Internet verbunden, um auf extern gelagerte Programm- und Nutzdaten sowie Updates und Kommunikationsfunktionen zugreifen zu können. Eine möglichst intuitive und durch geometrische Randbedingungen definierte, auf das „Wesentliche“ reduzierte Benutzeroberfläche mit sehr eingeschränkten Einstellungsmöglichkeiten steht hier einem komplexen Zusammenspiel verschiedenster Client-Serverdienste in einem heterogenen Netzwerk mit vielen potenziellen Fehlerquellen und Schwachstellen gegenüber.

Bei der Jobsuche von IT-Fachkräften spielt zunehmend als Auswahlkriterium seitens der Arbeitnehmer der Einsatz persönlicher Infrastruktur, also die Nutzung eines eigenen mobilen Gerätes auch für berufliche Zwecke innerhalb des Unternehmens, eine entscheidende Rolle. Der klassische Notizblock sowie Formulare werden durch Tablets mit entsprechender Software und Direktzugriff auf Datenbanken ersetzt, was einerseits die Effizienz steigert und eine Echtzeitverarbeitung aktueller Daten ermöglicht, andererseits erhebliche Gefahren durch Spionageaktivitäten und Angriffsszenarien verschiedener Interessensgruppen mit sich bringt.

Neben dem produzierenden Sektor gilt gleiches auch für Bildung und Forschung, wo zunehmend auf gemeinsame und zentral gewartete PC-Pools verzichtet wird, und sich die an einem Projekt oder einer Veranstaltung teilnehmenden Personen ganz selbstverständlich mithilfe ihres eigenen Gerätes beteiligen möchten. Diese Einbindung persönlicher IT in das Unternehmensnetzwerk hat sich als Begriff „Bring Your Own Device“ (BYOD) etabliert, wird von den Netzeinnehmern als sehr effizient empfunden und oft schon quasi als selbstverständlich vorausgesetzt, gerade bei Konferenzen oder solchen Projekten, bei denen der schnelle, mobile Informationsaustausch erfolgsentscheidend ist.

---

## 6.2 IT-Sicherheitsbegriff

Informationssicherheit besteht allgemein aus den drei Teilgebieten

1. Vertraulichkeit
2. Verfügbarkeit
3. Integrität

Die IT-Sicherheit beschäftigt sich mit den mit elektronisch speicher- und übertragbaren Daten verbundenen Teilespekten.

► **Vertraulichkeit** Semantik: Nur die autorisierten Personen und Zielgruppen sollen Zugriff auf die für sie bestimmten, möglicherweise sensiblen oder persönlichen Daten erhalten.

Technische Hilfsmittel: S. Abschn. 6.5.2

► **Verfügbarkeit** Semantik: Der Zugriff und Zugang zu IT-Infrastruktur als auch Daten soll (für die berechtigten Personenkreise (s. Vertraulichkeit) jederzeit gewährleistet sein. Dies schließt ein, dass Ausfälle durch Fehler oder Angriffe möglichst ohne signifikante Wartezeit behoben werden und die Daten, wenn sie gebraucht werden, sofort wieder zur Verfügung stehen.

Technische Hilfsmittel: S. Abschn. 6.5.3

► **Integrität** Semantik: Konsistenz und Authentizität sollen garantiert sein, d. h. es muss sichergestellt sein, dass Informationen durch Fehler oder gezielte Angriffe nicht verfälscht, zerstört oder unbrauchbar gemacht werden.

Technische Hilfsmittel: S. Abschn. 6.5.4

---

### 6.3 Vorfälle und Bedrohungsszenarien

Im Juni 2013 wurde durch den ehemaligen amerikanischen Geheimdienstmitarbeiter Edward Snowden das Ausmaß internationaler Spionageaktivitäten im Internet bekannt, von gezielter Überwachung von Privatpersonen bis hin zur Infiltrierung von Unternehmen und deren IT-Infrastruktur.<sup>2</sup> Hierbei wurden auch Informationen an Unterseekabeln abgefangen sowie Verbindungsdaten protokolliert, selbst verschlüsselte Kommunikation wurde mithilfe der Vorratsdatenspeicherung langfristig archiviert für einen Zeitpunkt, zu dem man möglicherweise die verschlüsselten Daten technisch dechiffrieren kann.

Die Tatsache, dass viele Internetdienste nicht von vornherein in Hinblick auf Sicherheitsaspekte entwickelt worden sind, mag den Geheimdiensten bei den umfangreichen Abhörprogrammen PRISM (USA) und Tempora (UK) zugutegekommen sein. Der Umstand, dass auch als vermeintlich sicher geglaubte verschlüsselte Dienste von den Abhöraktionen betroffen waren, unter Duldung marktführender internationaler Dienstanbieter, hat in der deutschen IT-Wirtschaft eine große Verunsicherung ausgelöst und zu teilweise panikartigen Strategiewechseln in Bezug auf IT-Sicherheit geführt.

Durch die Enthüllungen von Snowden sensibilisiert, wurden auch diverse sicherheitsrelevante Algorithmen überprüft, die als Referenzimplementation von Standardisierungsgremien herausgegeben wurden, und es wurden tatsächlich Fehler in Zufallsgeneratoren gefunden, die unter anderem zur Erzeugung kryptografischer Schlüssel verwendet werden, sodass Teile der für die Verschlüsselung verwendeten Bitfolgen vorhersagbar waren, was Brute-Force-Attacken auf verschlüsselte Inhalte erleichtert, da nicht mehr der gesamte durch die Schlüssellänge vorgegebene Zahlenraum durchsucht werden muss.

---

<sup>2</sup> Vgl. Wikipedia 2013a.

► **Brute-Force-Attacke** „Durchprobieren“ von Bitfolgen in möglichst hoher Geschwindigkeit (mit entsprechender Rechenleistung), um ein Passwort bzw. einen Schlüssel zum Dekodieren von Datensätzen zu finden. Der Aufwand reduziert sich, wenn bestimmte Randbedingungen vorgegeben sind wie „nur Codes, die Zeichen entsprechen, die sich mit auf einer Tastatur eingeben lassen“.

Durch die Affäre wurde allerdings nicht nur das Ausmaß der Eingriffe, äußerst umfangreiche Datensammlungen und Schwachstellen in der Nutzung einiger Netzzugangsdienste offenbart, sondern auch die Wirksamkeit bestimmter Schutzmechanismen wie mathematisch verifizierter Verschlüsselungsalgorithmen mit hoher Schlüssellänge bestätigt<sup>3</sup>, die richtig eingesetzt tatsächlich einen hohen Grad an Schutz für persönliche und unternehmenskritische Daten selbst angesichts scheinbar grenzenloser Rechenleistung auf der Seite der nach den Daten begehrenden Parteien bieten (s. Abschn. 6.5).

Das Ausspähen oder Manipulieren von Daten ohne gerichtliche Anordnung ist zwar nach nationalem wie internationalem Recht in den meisten Ländern eine Straftat, die Statistiken der Antivirenhersteller zeigen jedoch, dass offenbar eine zunehmende, hohe Motivation zur Programmierung von Schadsoftware und zielgerichtete Angriffe auf IT-Systeme durch entsprechende finanzielle Anreize unter Inkaufnahme der möglichen Aufdeckung und Bestrafung krimineller Aktivität existiert.

► **Virus (Computer-)** Eine Software, die System- und Anwendersoftware verändert mit dem Ziel, sich selbst zu reproduzieren und (ggf. mithilfe des Anwenders beim Datenaustausch mit anderen) zu verbreiten.

► **Wurm (Computer-)** Ein Computervirus, der zur Verbreitung eher auf automatisierbare Mechanismen wie Schwachstellen vernetzter Computersysteme zurückgreift als auf eine unbemerkte Verbreitung durch das Verhalten des Computernutzers.

► **Trojaner (Computer-)** Eine Software, die den Computernutzer durch eine versprochene und von diesem gewünschte Funktionalität dazu verleitet, die Software ungeprüft auszuführen und damit in der Software versteckte Funktionen zu aktivieren, die den Computer unter die Kontrolle des Angreifers bringen und ihn in den meisten Fällen über Kontrollserver im Internet „fernsteuerbar“ machen. Die Abgrenzung zu Viren und Würmern besteht im primären Ziel, weder sofort erkennbaren Schaden anzurichten oder sich unkontrolliert zu verbreiten, sondern so lange wie möglich unerkannt und unauffällig zu bleiben, um den Computer für den Auftraggeber beispielsweise als Teil eines Botnetzes jederzeit verfügbar zu halten, andere Systeme anzugreifen, beliebige Daten zwischenzuspeichern oder auch direkt Serverdienste wie Webshops oder Tauschbörsen mit nicht gesetzeskonformen Inhalten auf dem Computer des Benutzers kostenpflichtig anzubieten.

---

<sup>3</sup> Vgl. Schmidt 2013a.

- ▶ **Malware/Schadsoftware** Sammelbegriff für Viren, Würmer und Trojaner
- ▶ **Exploit** Ausnutzen einer Systemschwäche oder eines Fehlers zum unberechtigten Manipulieren von Daten oder Einschleusen von Schadsoftware in ein Computersystem.
- ▶ **Local Exploit** Ausnutzen einer Systemschwäche, meist zum Erlangen eines privilegierten Benutzerstatus (Administrator) und Installation von Schadsoftware zur Erhaltung dieses Status als bereits angemeldeter, ursprünglich unprivilegierter Benutzer.
- ▶ **Remote Exploit** Ausnutzen einer Systemschwäche über das Netzwerk zum Eindringen in ein System unter Umgehung der regulären Anmeldung mit dem Ziel, Schadsoftware einzuschleusen und den Zugang weiter offen zu halten.

Der Aktivität von Schadsoftwareprogrammierern steht die der Sicherheitsexperten entgegen, die Schadsoftware und die von ihr ausgenutzten Schwachstellen analysieren und Gegenmaßnahmen auf der Ebene des Betriebssystems oder spezieller Schutzprogramme entwerfen, die die Infektion mit Schadsoftware und deren Ausbreitung unterbinden sowie die sensiblen Nutzerdaten schützen sollen. Man könnte von einem „Wettlauf“ sprechen, da die Zeitkomponente durchaus kritisch ist: Angreifer suchen nach einem „Zero Day Exploit“, also die direkte Ausnutzung einer offenen Schwachstelle ohne Verfügbarkeit einer Abhilfe, während sich Sicherheitsbeauftragte damit beschäftigen, ein System entweder von vornherein sicher zu entwerfen oder durch eine „Systemhärtung“ mit restriktiven Rechten und Kontrollen zu verhindern, dass sich Programmfehler für Exploits ausnutzen lassen.

In sicherheitskritischen Systemen im Embeddedbereich wird zudem auf die benutzerdefinierte Installation von Software verzichtet und sowohl das Betriebssystem als auch alle laufenden Dienste und Anwendungen einer Überprüfung unterzogen, um sämtliche Angriffspotenziale auch ohne zusätzliche, aufwendige Schutzmaßnahmen auszuschließen. Im Rahmen dieses Kapitels beschäftigen wir uns jedoch mit benutzerdefinierten Installationen auf mobilen Geräten, Betriebssystemen und Apps.

---

## 6.4 Sicherheitsarchitektur mobiler Betriebssysteme und Anwendungen am Beispiel Android

### 6.4.1 Konzept

Die Android-Plattform hat das Betriebssystem Linux binnen kurzer Zeit an Platz 1 der Verkaufszahlen für Smartphones mit fast 80 % Marktanteil im 3. Quartal 2013 katapultiert.<sup>4</sup> Google veröffentlicht in den Technischen Informationen zu Android einen Abschnitt über

---

<sup>4</sup> Vgl. Llamas et al. 2013.

das im Folgenden zitierte Sicherheitskonzept<sup>5</sup> der Open-Source-Basis und stützt sich hierbei größtenteils auf die bereits unter Unix-Systemen bekannten Datei- und Prozessrechte im Multiuserbetrieb, die eine Beeinflussung von Ressourcen eines Anwenderprogramms durch ein anderes verhindern oder zumindest kontrollierbar halten sollen.

### Zitat aus „Android Security Overview“

#### The Application Sandbox

The Android platform takes advantage of the Linux user-based protection as a means of identifying and isolating application resources. The Android system assigns a unique user ID (UID) to each Android application and runs it as that user in a separate process. This approach is different from other operating systems (including the traditional Linux configuration), where multiple applications run with the same user permissions.

This sets up a kernel-level Application Sandbox. The kernel enforces security between applications and the system at the process level through standard Linux facilities, such as user and group IDs that are assigned to applications. By default, applications cannot interact with each other and applications have limited access to the operating system. If application A tries to do something malicious like read application B's data or dial the phone without permission (which is a separate application), then the operating system protects against this because application A does not have the appropriate user privileges. The sandbox is simple, auditable, and based on decades-old UNIX-style user separation of processes and file permissions.

Was im Zitat als „kernel-level Application Sandbox“ bezeichnet wird, ist also in erster Linie das von Unix-Systemen bekannte Sicherheitskonzept, das Benutzerprozesse sowie -daten voneinander isoliert, sofern die Benutzer-IDs und Dateirechte entsprechend gesetzt sind. Allerdings sind aus dieser Definition noch keine darüber hinausgehenden Schutzmechanismen erkennbar, die einen Vorteil gegenüber einer handelsüblichen Linux-Distribution versprechen. Insbesondere wurde erst mit Android 4.3 SE Linux zur Verfügung gestellt, was bemerkenswerterweise bereits vor 10 Jahren von der NSA als Open-Source-Kernel-Erweiterung zur Erstellung eines vor Angriffen weitestgehend geschützten Betriebssystems entwickelt wurde und das die Linux-Kernel-API um eine sehr fein granulierte Zugriffskontrolle vieler Systemroutinen bereichert.<sup>6</sup> Dies ermöglicht unter anderem einen gezielten, sowohl systemweiten als auch appbezogenen Schutz von Netzwerk und Datenträgern.

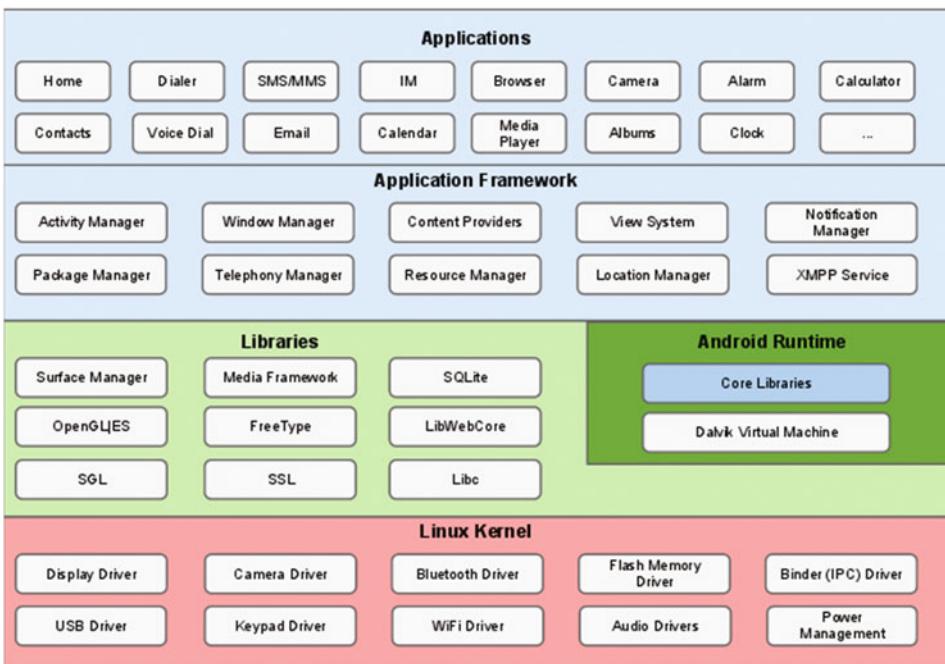
Android-Apps sind, bezogen auf die Syntax, in der Programmiersprache Java geschrieben, werden aber von einer von Google spezialisierten virtuellen Maschine namens „dalvik“ ausgeführt, die ihrerseits eine API zum Zugriff auf Systemressourcen zur Verfügung stellt und damit die Anwendung von der nativ laufenden Systemsoftware entkoppelt (vgl. Abb. 6.1). Allerdings dürfen Apps unter Android auch nativen Code für die verwendete Hardwareplattform, meist ARM-basiert<sup>7</sup>, in Form von dynamisch gebundenen Bibliotheken oder Modulen enthalten, die bei Bedarf geladen und direkt vom Linux-Betriebssystem verarbeitet werden.

---

<sup>5</sup> Vgl. Google 2013a.

<sup>6</sup> Vgl. National Security Agency 2013.

<sup>7</sup> Vgl. Wikipedia 2013b.



**Abb. 6.1** Android software stack. (Vgl. Google 2013b)

Als zusätzlicher Schutz vor unautorisierte Veränderung der Android-Systemsoftware befinden sich der Systemkern und die vom Hersteller als essenziell betrachteten Anwendungen auf einer Partition des Systemdatenträgers, die im Normalbetrieb nur-lesbar eingebunden wird (`/system`). Zum Einspielen von Updates wird diese Partition kurzzeitig auch schreibbar eingebunden, damit die Daten aktualisiert werden können. Da hierzu keine Referenzen auf die Daten durch das laufende System mehr vorhanden sein dürfen, bzw. da Systemdateien, auf die noch zugegriffen wird, nicht während eines Zugriffs bereits überschrieben werden, verlagern die meisten Android-Geräte den eigentlichen Updatevorgang in den Bootloader, der bei Vorhandensein eines Updatepaketes in der Datenpartition zunächst die Signatur des Updates anhand einer Liste gültiger Zertifikate überprüft und anschließend das Update auf die Systempartition schreibt. Nach Abschluss des Schreibvorganges wird der aktualisierte Linux-Kernel von der `/system`-Partition gestartet und diese wieder auf den Status „nur-lesbar“ gesetzt.

Während des Normalbetriebes ist im Sicherheitskonzept von Android vorgesehen, dass jedes Anwenderprogramm (also jede von der dalvik-VM gestartete App) mit einer eigenen Benutzerkennung läuft. So wird der Zugriff auf Dateisystem- und Prozessebene auf die der entsprechenden Benutzer-ID gehörenden Ressourcen eingeschränkt.

### Prozesse unter Android (verkürztes Listing)

USER	PID	PPID	VSIZE	RSS	WCHAN	PC	NAME
root	1	0	596	460	ffffffffff	00000000 S	/init
root	2	0	0	0	ffffffffff	00000000 S	kthreadd
root	3	2	0	0	ffffffffff	00000000 S	ksoftirqd/0
root	6	2	0	0	ffffffffff	00000000 S	migration/0
root	7	2	0	0	ffffffffff	00000000 S	watchdog/0
u0_a40	3652	2754	513924	48596	ffffffffff	00000000 S	com.android.systemui
u0_a72	3727	2754	494660	29152	ffffffffff	00000000 S	org.pocketworkstation.pckeyboard
u0_a36	3739	2754	514456	23480	ffffffffff	00000000 S	com.google.process.location
radio	3758	2754	502852	23148	ffffffffff	00000000 S	com.android.phone
nfc	3768	2754	499620	19120	ffffffffff	00000000 S	com.android.nfc
u0_a36	3828	2754	529472	30048	ffffffffff	00000000 S	com.google.process.gapps
nfc	3887	2754	482320	16576	ffffffffff	00000000 S	com.android.nfc:handover
u0_a49	3931	2754	478888	16396	ffffffffff	00000000 S	com.android.smsspush
u0_a17	3991	2754	491552	24500	ffffffffff	00000000 S	android.process.media
u0_a36	4224	2754	503336	20856	ffffffffff	00000000 S	com.google.android.gsf.login
u0_a13	4387	2754	479760	16452	ffffffffff	00000000 S	com.bel.android.dspmanager
u0_a60	4769	2754	509288	26924	ffffffffff	00000000 S	com.android.vending
u0_a105	6765	2754	512076	46448	ffffffffff	00000000 S	net.osmand.plus
u0_a10	6796	2754	504392	28528	ffffffffff	00000000 S	com.android.calendar
u0_a11	6813	2754	482032	19508	ffffffffff	00000000 S	com.android.providers.calendar
u0_a9	6947	2754	489644	28820	ffffffffff	00000000 S	com.android.calculator2

In der ersten Spalte des Listings ist die Benutzer-ID des Prozesses sichtbar, dessen Name in der letzten Spalte angegeben ist. Die Tatsache, dass die hier mit *com.* beginnenden Prozessnamen über die dalvik-VM gestartet wurden, wird von Android kaschiert.

Systembedingt sind unter Android einige Bereiche des Systems für alle Benutzer-IDs les- oder schreibbar, wozu Temporärverzeichnisse gehören. Als Dateisystem für die Datenpartition /data, die von den Apps als Speicher für eigene Daten genutzt wird, kommt ext4 zum Einsatz, das anders als das bekannte FAT32 auch Unix-spezifische Dateirechte und Spezialdateien wie symbolische Links, Devices (Gerätedateien), Pipes und Sockets erlaubt.

#### 6.4.2 Verschlüsselung der Datenpartition

Android unterstützt ab Version 3.0 die Verschlüsselung der Datenpartition /data über dm-crypt im Linux-Kernel, um programmspezifische Daten wie Zugangsdaten und elektronische Zahlungsverfahren vor Bekanntwerden durch Diebstahl des physischen Gerätes zu schützen. Hierbei werden nicht einzelne Dateien, sondern die gesamte /data-Partition blockweise durch AES mit einer Block- und Schlüssellänge von 128bit chiffriert, wobei der durch eine Zufallszahl generierte symmetrische Schlüssel am Ende der Partition in

einem unbenutzten Bereich abgelegt und zuvor ebenfalls mit dem gleichen Algorithmus verschlüsselt wird. Als „Schlüssel zum Schlüssel“ wird der Hashwert eines vom Benutzer eingegebenen Passwortes verwendet. Entschließt sich der Benutzer später, das Passwort zum Entsperren der Partition zu wechseln, wird die Partition nicht erneut komplett verschlüsselt, sondern lediglich der bereits vorhandene symmetrische Schlüssel mit einem neuen Passworthash verschlüsselt und erneut abgespeichert, sodass das alte Passwort zugunsten des neuen ungültig wird. Im stromlosen Zustand ist die /data-Partition ohne den entschlüsselten symmetrischen Schlüssel weder einzubinden noch ist der Inhalt lesbar. Nach dem Einschalten des Gerätes muss der Benutzer recht früh im Bootvorgang das Passwort eingeben, dessen Hashwert den symmetrischen Schlüssel aktiviert. Das eingegebene Passwort wird dabei nie im Klartext gespeichert und auch nicht direkt für eine Verschlüsselung verwendet.

Die Verschlüsselung der /data-Partition lässt sich bei aktuellen Android-Versionen aus den Einstellungen im Menüpunkt „Sicherheit“ aktivieren, allerdings ist eine Rückwandlung ins unverschlüsselte Format offenbar nicht mehr vorgesehen und erfordert ggf. manuelle Aktivität oder ein vollständiges Zurücksetzen des Gerätes nach Backup der Daten auf einen externen Datenträger.

AES<sup>8</sup> gilt als sehr zuverlässig und auch die vermeintlich kurze Schlüssellänge von 128bit gilt bei dem symmetrischen Verfahren noch als ausreichend sicher gegen Brute-Force-Angriffe. Der AES-Ver- und -Entschlüsselungsalgorithmus ist mittlerweile auf vielen Computerarchitekturen in die Hardware implementiert und ermöglicht das Arbeiten mit verschlüsselten Daten fast ohne Geschwindigkeitsverlust.

#### 6.4.3 APK-Dateien und Rechtesystem von Android-Apps

Während dalvik das .dex-Format mit Binärkode verwendet, ähnlich wie .class-Dateien bei Java, sind .apk-Dateien ZIP-Archive, vergleichbar mit .jar-Dateien in Java, die alle für eine Anwendung (einen Prozess) notwendigen Klassen und Bibliotheken (auch prozessorspezifische, native Objektdateien) enthalten. Eine Manifest-Datei (MANIFEST.MF) stellt ein Inhaltsverzeichnis mit Prüfsummen der im Archiv enthaltenen Dateien zur Verfügung, um Übertragungs- und Speicherungsfehler der .apk-Dateien prüfen zu können.

Bereits bei der Installation einer App wird der Benutzer darauf hingewiesen, dass bestimmte Rechte vom System angefordert werden, und kann sich daraufhin entscheiden, die App gar nicht erst zu installieren. Die App muss über die zuvor vom Benutzer bestätigten Rechte Zugriff auf bestimmte Teile der API freigeschaltet bekommen, wie Internetzugang oder die Möglichkeit, Daten sowie persönliche Einstellungen lesen oder schreiben zu können. Die Angabe, welche Ressourcen verwendet werden, befindet sich in der Datei `AndroidManifest.xml`<sup>9</sup> des zur App gehörenden .apk-Archivs.

<sup>8</sup> Vgl. Wikipedia 2013c.

<sup>9</sup> Vgl. Google 2013c.

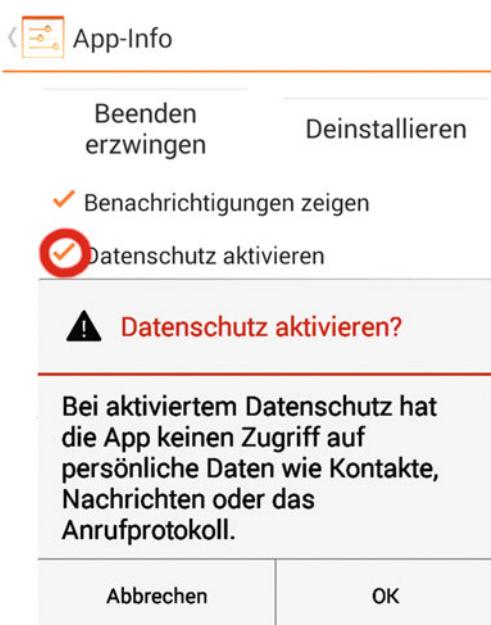
**Abb. 6.2** Angeforderte Rechte einer Android-App vor der Installation



In der gezeigten Anfrage (vgl. Abb. 6.2) werden das Auslesen und Ändern von Kontakt- daten, Lesen und Schreiben auf den Datenbereich, Zugriff auf und Anlegen von Konten, Telefonanrufe und Internetzugriffe in Kombination miteinander verlangt, was im Ergebnis bedeutet, dass persönliche Daten ausgelesen, über das Internet übertragen und lokal auch verändert werden könnten. Der Anwender sollte sich in diesem Fall gewissenhaft überlegen, ob die zur Installation beabsichtigte App diese Rechte wirklich benötigt und ob er dem Hersteller vertrauen kann.

Aus sicherheitstechnischer Sicht ist die Bestätigung der Liste von angeforderten System- rechten vor der Installation zwar ein wichtiges Merkmal, allerdings sind die dargestellten Rechte bei genauerer Betrachtung nicht sehr detailreich formuliert und der Benutzer hat auch nicht die Möglichkeit, einzelne Rechte zu verweigern unter Verzicht auf eine Teil-

**Abb. 6.3** Zusätzliche Aktivierung von Schutzfunktionen bereits installierter Apps unter Android 4.3



funktionalität. Auch hat der Anwender zu diesem Zeitpunkt noch nicht die Möglichkeit erhalten, die Lizenz, unter der ihm die Software überlassen wird, zu lesen und zu akzeptieren, es sei denn, der Softwarehersteller hat dies in der Beschreibung seines Produktes ausdrücklich vorgesehen.

Ab Android Version 4.3 hat der Anwender die Möglichkeit, in den Systemeinstellungen zumindest den Zugriff auf private Daten app-spezifisch einzuschränken (Abb. 6.3).

#### 6.4.4 Herstellerspezifische Restriktionen, Rooting und alternative Firmware

Grundsätzlich sind in Android aufgrund seiner Open-Source-Lizenz Änderungen an der Systemsoftware erlaubt<sup>10</sup>, jedoch scheinen sich viele Hersteller vor einem Supportproblem zu sehen, wenn der Austausch von Systemsoftware oder Modifikationen an ihrem eigenen, oft proprietären, Diagnose- und Update-System vorgenommen werden, und implementieren hard- und softwareseitig Maßnahmen, um nicht von ihnen autorisierte Änderungen am System auf technischer Ebene zu verhindern.

Durch das Umschalten des Flashbausteins auf „read-only“ ist es beispielsweise nicht möglich, vorinstallierte Apps des Herstellers zu entfernen, auch Updates können lediglich auf die Datenpartition gespeichert werden. Und das mitgelieferte Installationsprogramm

<sup>10</sup> Vgl. Wikipedia 2013d.

für .apk-Dateien prüft mitunter, ob die vom Anwender zur Installation vorgesehene Software auch vom Hersteller des Gerätes signiert wurde, und verweigert bei fehlgeschlagener Verifikation den Dienst. Solange der Hersteller den Anwender regelmäßig mit Updates versorgt, können viele Anwender mit den zusätzlichen Einschränkungen leben, aber wenn der Herstellersupport für ein „älteres“ Gerät eingestellt wird oder Sicherheits- und Featureupdates nicht mehr regelmäßig geliefert werden, fühlen sich Anwender, die von Open-Source-Betriebssystemen wie GNU/Linux eine größere Flexibilität gewohnt sind, durchaus benachteiligt.

Der Begriff „Rooten“ oder „Rooting“ stammt aus dem Unix-Jargon und bezeichnet die Aktivierung eines Administratorzuganges, d. h. eines Benutzers ohne Einschränkungen, der in Unix traditionell den Benutzernamen „root“ trägt, bei Bedarf. Es bedeutet hingegen nicht, dass jedes Programm sofort und automatisch mit Administratorrechten laufen soll. Da der Unterbau von Android ein schlankes Linux-System ist, ist für das jederzeit ein- und ausschaltbare Administratorrecht lediglich ein Programm hinzuzufügen, das aufgrund seiner Dateirechte für die Dauer seiner Ausführung von der unprivilegierten User-ID zum Benutzer mit ID 0 wechselt und unter dieser ID Programme starten kann, für die dann die gesetzten Dateirechte aufgrund des Administratorstatus nicht gelten.<sup>11</sup> Um ein zusätzliches Programm zu installieren, dessen Installation von den herstellerbedingten Restriktionen verhindert wird, müsste jedoch bereits ein Administratorzugang existieren, sodass ein Initialproblem entsteht, das sich oft nur durch die Ausnutzung von systematischen Schwächen in der Implementation der jeweiligen Restriktion lösen lässt, d. h. der Anwender, der sein mobiles System von den Restriktionen befreien möchte, muss sich in gewisser Weise als „Angreifer“ betätigen, um die vollständige Kontrolle über sein eigenes Gerät dauerhaft zu erlangen. Je nach Hersteller und Gerät ist der Aufwand hierfür unterschiedlich hoch. Einige Gerätehersteller bieten Hilfestellungen und das Freischalten der Schreibfunktion für Bootlader und Systempartition an, möchten den Benutzer allerdings zuvor vertraglich verpflichten, auf die Herstellergarantie und Gewährleistung sowohl für Soft- als auch Hardware zu verzichten – eine juristisch sehr fragliche Praxis, deren Legalität noch immer überprüft wird.

Das Rooten hat sicherheitstechnisch betrachtet sowohl Vor- als auch Nachteile. Einerseits entsteht durch die Verweigerung der Hersteller bezüglich Gewährleistungsansprüchen ein Risiko des selbst zu tragenden Totalverlustes („bricked“ devices), zumindest ist der Aufwand für eine Wiederherstellung im Fehlerfall oft hoch und erfordert für die Reparatur Expertenwissen. Andererseits stellen ungepatchte Systemfehler, veraltete Benutzeroberflächen mit mangelhafter Bedienbarkeit und die Unmöglichkeit, Software ohne Hilfe des Herstellers auf den aktuellen, sicheren Stand bringen zu können, ebenso ein Risiko dar, weswegen auch viele nicht technisch versierte Anwender mehr und mehr dazu tendieren, ihr System zu rooten bzw. von Fachpersonal rooten zu lassen. Diesbezüglich hat

---

<sup>11</sup> Sudo bzw. su.

sich eine regelrechte Community gebildet, die in Foren Erfahrungen und Ratschläge zu Themen außerhalb der vom Hersteller vorgesehenen Nutzungsmöglichkeiten austauscht.<sup>12</sup>

#### 6.4.5 „Back to the roots“ – Cyanogenmod als Open Source Android

Eine über das Rooting hinausgehende Modifikation stellt die Installation einer alternativen Firmware, bzw. eines nicht vom Hersteller des Gerätes selbst stammenden Android-Derivates dar, um Zugriff auf die aktuellste Entwicklung der Android-Basis sowie Unterstützung neuer Anwendungen zu erhalten. Das von Steve „Cyanogen“ Kondik gegründete Cyanogenmod-Projekt<sup>13</sup> setzt den von Google publizierten unter Open-Source-Lizenz stehenden Android-Code in gerätespezifische Installationspakete mit fast in jedem Fall vollständiger Unterstützung der Hardwareeigenschaften um. Dabei liegt der Fokus der Arbeit auf Sicherheit, Aktualität und für den Anwender leichte Aktualisierbarkeit des Systems. Nach der Hürde der ersten Installation bzw. des Ausschaltens der herstellerbedingten Restriktionsmechanismen lässt sich Cyanogenmod bei Bedarf oder in regelmäßigen Abständen, bei einigen Versionen sogar täglich per „Nightly Build“, aktualisieren. Ein erweitertes Recoverysystem zum Instandsetzen oder Neuaufsetzen der Systempartition im Fehlerfall ist Bestandteil des Systems ebenso wie eine standardmäßig installierte Rootschnittstelle, die programmatisch automatisch aktiviert oder deaktiviert werden kann. Cyanogenmod lässt sich sowohl ohne proprietäre zusätzliche Apps betreiben als auch mit dem Zusatz der Google Apps („gapps“), die eine Anbindung an die von Google zur Verfügung gestellten Netzwerkdienste sowie die Nutzung des Google-Play-Store erlauben, um wie bei den herstellerspezifischen Android-Varianten freie, kostenlose oder kostenpflichtige Software zusätzlich zu installieren. Ende September 2013 gründete sich die *Cyanogen Inc.* als Firma<sup>14</sup> mit dem Ziel, auch kommerziellen Support, Installationshilfen und mobile Geräte mit bereits vorinstalliertem Cyanogenmod liefern zu können. Die communitybasierte Open-Source-Basis von Cyanogenmod mit Anleitungen und installierfähigen Paketen soll jedoch unverändert weiter unterstützt und kostenlos angeboten werden.

Das Risiko des Wegfalls der Herstellergarantie oder eines Totalverlustes ist bei Alternativfirmware ähnlich hoch wie beim Rooten des Gerätes, jedoch bieten die meist im ersten Schritt installierten Recoverysysteme die Möglichkeit des neuen Versuches einer Installation über Tastenkombinationen beim Einschalten und Flashen per USB-Kabel oder sogar das Wiederaufspielen der Originalsoftware zur womöglichen Wiedererlangung der Herstellergarantie des „unmodifizierten“ Gerätes.

---

<sup>12</sup> Bspw. XDA Developers (<http://forum.xda-developers.com>).

<sup>13</sup> Vgl. Cyanogenmod 2013.

<sup>14</sup> Vgl. Diedrich 2013.

## 6.5 Technische Sicherheitsaspekte und Schutzmaßnahmen

### 6.5.1 Fallbeispiel E-Mail

E-Mail gehört auch bei mobilen Geräten heute noch zu den meist genutzten und ältesten Internetdiensten, obwohl zunehmend auch „Echtzeitkommunikation“ in der Art eines Chats als Alternative verwendet wird.

Das SMTP-Protokoll wurde 1982 unter RFC821<sup>15</sup> standardisiert und realisiert nach einem einfachen Schema die Übertragung von Mailnachrichten von einem Client an einen Server, der diese Nachricht dann, ggf. über weitere Server als Zwischenstationen, an den Mailserver der Zieldomain Zustellt. Dort kann die Nachricht, in der nun laut Standard sämtliche beteiligten Zwischenstationen in den Received-Headern verzeichnet sind, mit einem Mailclient nach Authentifizierung über andere Protokolle (POP<sup>16</sup>, IMAP<sup>17</sup>) abgeholt und auf das Endgerät übertragen werden.

Im folgenden Beispiel wird mit dem Verbindungsprogramm `telnet` die Übertragung einer E-Mail von einem Client an den Server simuliert. Die Protokollelemente von SMTP bestehen größtenteils aus „Kommandos“ (hier groß geschrieben) mit Parametern. Der Server antwortet auf jede Eingabe mit einem Statuscode und einer Zusatzinformation, die der Client auswerten kann.

---

#### SMTP-Session

```
HELO localhost
250 knopper.net
MAIL FROM: root@localhost
250 2.1.0 Ok
RCPT TO: knopper@knopper.net
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: niemandhier@localhost
Subject: Test
Test, test...
.
250 2.0.0 Ok: queued as AFB0C50CB
```

Die Statuscodes suggerieren zwar eine Art „Überprüfung“ der Eingaben auf Plausibilität, tatsächlich ist im Mailprotokoll jedoch nur die korrekte Syntax der Kommunikation vor-

---

<sup>15</sup> Vgl. Postel 1982.

<sup>16</sup> Vgl. Myers und Rose 1996.

<sup>17</sup> Vgl. Crispin 2003.

geschrieben. So wird beispielsweise anhand des Zeichens @ erkannt, dass es sich um eine Mailadresse mit Benutzer- und Domainanteil handelt. Restriktiver eingestellte Mailgateways überprüfen noch, ob die Zieladresse (RCPT TO) einen gültigen DNS MX-Record enthält und ob Mail-Relying für diese Adresse erlaubt ist, jedoch wird bei Akzeptanz der E-Mail der mit DATA eingelegte Datenteil unverändert übertragen, insbesondere auch eine als *From:* angegebene, falsche Absenderadresse. Durch die fehlende Prüfung der Identität des Absenders und Empfängers ist es für Versender von Werbemails („Spammer“) sehr einfach, falsche Identitäten vorzutäuschen und Mailserver zum Versand von fast beliebigen Inhalten an eine Anzahl lokaler oder, bei nicht eingeschränktem Relaying, auch an Benutzer auf entfernten Systemen zu verschicken.

Der einzig verlässliche, durch das Protokoll festgelegte Teil einer einfachen E-Mail sind die *Received*-Zeilen im Header, die Herkunft und Weg einer E-Mail nachvollziehbar machen, was oft schon ausreicht, um eine „Falschmeldung“ zu erkennen.

#### Mailheader eines Trojanerangriffs

```
Received: from knopper.net (localhost [127.0.0.1])
          by filter.knoppernet.local (Postfix) with ESMTP id 83730596F
          for <knopper@knopper.net>; Tue, 10 Sep 2013 01:13:55 +0200 (CEST)

Received: from mail-lb0-f171.google.com (mail-lb0-f171.google.com [209.85.217.171])
          by knopper.net (Postfix) with ESMTP ; Tue, 10 Sep 2013 01:13:55 +0200 (CEST)

Received: by mail-lb0-f171.google.com with SMTP id u14so55187071bd.2
          for <knopper@knopper.net>; Mon, 09 Sep 2013 16:13:54 -0700 (PDT)

Received: by 10.114.95.8 with HTTP; Mon, 9 Sep 2013 16:13:54 -0700 (PDT)
Subject: Warnmeldung: Sehr geehrter Bank-Kunde
...
...
```

Die gekennzeichneten Stellen verraten die Herkunft der vermeintlichen „Warnmail“ als durch ein Webformular von einer dynamisch zugewiesenen IP-Adresse aus generiert. Diese Untersuchung der Herkunft wird jedoch von den betroffenen Empfängern selten durchgeführt, obwohl auch Mailprogramme auf mobilen Systemen durchaus die Option „Alle Header anzeigen“ unterstützen.

Die gesamte Kommunikation im einfachen SMTP läuft standardmäßig unverschlüsselt und für alle im gleichen LAN-Subnetz angeschlossenen Computer mit entsprechender Software („Sniffer“) protokollier- und mitlesbar ab. Im Basis-SMTP-Protokoll ist weder eine Authentifizierung des Absenders noch eine Sicherung des Übertragungsweges vorgesehen, der Mailserver muss also alleine aufgrund der beliebig vom Client setzbaren TO- und FROM-Adresse entscheiden, ob die E-Mail lokal zugestellt, weiter transportiert werden darf (Relaying) oder mit einem Fehlerstatus abgelehnt wird (Reject).

Ein häufiges Angriffsszenario für Malwareversender ist es hier, ungültige oder falsche Absenderadressen zu verwenden. Wenn Schadsoftware verschickt wird, wird häufig zunächst vom Angreifer getestet, ob der Mailserver möglicherweise eine als Virus oder Trojaner erkannte E-Mail an den (vermeintlichen) Absender zurückschickt (sog. Backscatter), und als Absenderadresse ebenfalls die Mailadresse des „Opfers“ eingetragen. Die Schadsoftware erreicht dann gerade wegen eines zu aktiv konfigurierten Virensenders den als Absender eingetragenen, beabsichtigten Empfänger.

### **Abhilfemaßnahme gegen Weiterleitung von Schadsoftware per E-Mail**

- Auf der Serverseite: Kein aktives „Zurücksenden“ von Nachrichten, vielmehr sollen schädliche Inhalte, bekannte falsche Absender- oder Empfängeradressen bereits während des SMTP-Dialoges erkannt und der Dialog mit einer Fehlerstatusmeldung abgebrochen werden. Auf diese Weise erhält ein interaktiver Absender direkt eine Fehlermeldung, ein Angreifer mit automatisiert gestarteten Programmen oder Skripten hingegen nicht, und die E-Mail wird auch nicht erneut zugestellt oder in eine „Fehlermail“ eingepackt an andere Adressen weitergeleitet. Virensanner oder Spamfilter müssen entsprechend in einer „Sandwich“-Bauweise konfiguriert werden, der aktive Versand von Fehlermails oder gar Anhängen muss abgeschaltet werden. Dies schützt das Netzwerk auch vor mobilen Geräten mit bereits aktiver Schadsoftware, die sich per Mail über den entsprechend vorbereiteten Server nicht weiter verbreiten kann.

### **Abhilfemaßnahme gegen das Mitlesen von E-Mail im Netzwerk**

- Verwendung von Verschlüsselung auf dem Transportweg: Mithilfe von TLS (Transport Layer Security) kann der SMTP-Dialog vor Abhören geschützt werden, wenn der Client das vom Server präsentierte Zertifikat erfolgreich verifiziert und akzeptiert. Dies schützt sensible Mailinhalte zumindest während der Übertragung vom Client zum Server. Die überwiegende Mehrheit der Mailclients unterstützt den Mailversand über SMTPS oder SMTP + TLS, viele Provider mit eigenen Mailservern stellen derzeit von unverschlüsseltem SMTP auf TLS um und empfehlen ihren Kunden, die entsprechende Konfiguration ihrer Mailclients zum frühestmöglichen Zeitpunkt.

### **Abhilfemaßnahme gegen das unkontrollierte Versenden von E-Mail durch „Gäste“ im Netzwerk**

- Verwendung von Authentifikation im erweiterten Mailprotokoll ESMTP in Verbindung mit Verschlüsselung. Hierbei dürfen, wie beim Mailempfang selbstverständlich, ausschließlich berechtigte Personen eine E-Mail über den Server zustellen. Im Mailclient ist entsprechend ein Login/Passwort oder ein Clientzertifikat zur Autorisierung hinterlegt. Dies funktioniert allerdings nur bei der direkten Zustellung an ein festes Mailgateway, da andere Mailserver den Benutzer nicht kennen bzw. nicht authentifizieren können. Auch ist eine Verschlüsselung hinter dem Mailserver (z. B. beim Relaying über weitere Server) nicht gewährleistet.

## S/MIME: Abhilfemaßnahmen gegen Ausspähen und Manipulieren von E-Mail auf der gesamten Übertragungsstrecke sowie auch nach dem Transport oder in Zwischen speichern

- Ende-zu-Ende (auch: Punkt-zu-Punkt) Verschlüsselung: Hier wird vom Absender bis zum Empfänger durchgehend verschlüsselt, und selbst beim Empfänger wird eine archivierte Mailnachricht nur verschlüsselt auf dem Endgerät gespeichert, hierbei kommen asymmetrische Verschlüsselungsverfahren zum Einsatz (s. a. Abschn. 6.5.2, S/MIME).
- Elektronische Signatur des versendeten Dokuments, ebenfalls mithilfe asymmetrischer Verfahren (s. Abschnitt „Integrität“).

### 6.5.2 Vertraulichkeit

Zur Vertraulichkeit zählt der Zugang zu den Funktionen mobiler IT-Systeme nur für die berechtigten Personenkreise (i. d. R. der Eigentümer des Gerätes) sowie der Schutz gegen Ausspähen von Informationen bei deren Generierung, Übertragung und Speicherung.

#### 6.5.2.1 Zugang

Nicht nur der Zugriff auf standardisierte Internetdienste ist durch Authentifikationsmechanismen geschützt, sondern auch der Zugriff auf die Benutzeroberfläche des mobilen Endgerätes. Als zuverlässig gelten hier manuell einzugebende Passwörter ausreichender Länge und Komplexität.

Generell wird von den Hard- und Softwareherstellern nach Wegen gesucht, die von vielen Anwendern als lästig empfundene (Re-)Autorsierung zu vereinfachen. So werden auf modernen Smartphones oft auch Gesten, Muster, Gesichtserkennung und sogar biometrische Merkmale wie Fingerabdrücke zur Authentifizierung herangezogen. Hierbei gilt, je weniger leicht das persönliche Authentifikationstoken zu erraten oder zu fälschen ist, desto sicherer ist das Anmeldeverfahren. Mustereingaben sind durch den vom Benutzer bevorzugten, einfachen Bewegungsablauf oft nach wenigen Versuchen zu erraten, auch kann durch Spuren auf der Glasoberfläche auf das verwendete Muster geschlossen werden<sup>18</sup>.

Biometrische Verfahren, die vermeintlich eindeutige und für jeden Menschen einmalige Körpermerkmale zur Identifikation heranziehen, haben sich in den letzten Jahren nicht zuletzt durch Aktionen des Chaos Computer Clubs in vielen Punkten als unzuverlässig erwiesen. Auch Fingerabdruckscanner in aktuellen Geräten können mit relativ geringem Aufwand getäuscht werden.<sup>19</sup>

Zwei-Faktor-Authentifizierung ist ein Ansatz, der eine verlässlichere Identifikation und Autorisation des Nutzers durch die Verwendung von mindestens zwei unterschiedlichen Verfahren bieten soll, z. B. das im Onlinebanking verwendete Verfahren, sich zunächst gegenüber der Webseite mit einem einfachen Passwort zu identifizieren, Aufträge jedoch

<sup>18</sup> Vgl. Schmidt 2013b.

<sup>19</sup> Vgl. Schmidt 2013c.

erst nach Eingabe einer nur für die aktuelle Transaktion gültigen Nummer durchzuführen, die nach Anforderung über eine SMS auf das Handy oder Smartphone des Anwenders gesendet wird. Hier dient der Besitz des zuvor bei der Bank registrierten mobilen Gerätes als zusätzliches Authentifikationstoken.

Einen ähnlich hohen Schutz bietet die Anmeldung über Smartcards oder Public-Key-Verfahren, bei denen neben dem Besitz eines eindeutigen persönlichen Schlüssels zusätzlich noch ein Passwort eingegeben werden muss, das ausschließlich die berechtigte Person kennt. Public-Key-Verfahren werden im nachfolgenden Abschnitt im Detail analysiert und sind ebenfalls für den Sicherheitsaspekt der Integrität durch digitale Unterschriften relevant.

### 6.5.2.2 Daten

Im Klartext übertragene Daten sind mit meist geringem technischen Aufwand im Netzwerk abhörbar, wodurch Angreifer auch die Informationen erhalten können, die für den Zugang zu Geräten und Diensten genutzt werden, und so Identitäten von Nutzern übernehmen können. Dabei reicht oft schon die Kenntnis weniger persönlicher Informationen wie Name, Geburtsdatum und Wohnort, um sich telefonisch gegenüber Anbietern von Dienstleistungen erfolgreich als eine andere Person auszugeben und ggf. weitere Zugangsdaten zu erhalten oder umstellen zu lassen.

Bei Smartphones ist der Aspekt des Telefonierens noch zu betrachten. Durch das Design des Verbindungsbaus bei GSM muss sich das Gerät zwar gegenüber der Basisstation mit seiner auf der SIM-Karte gespeicherten International Mobile Subscriber Identity (IMSI) authentifizieren, jedoch ist eine Authentifikation oder Verifikation der Basisstation gegenüber dem Mobilfunkgerät, das sich i. d. R. immer automatisch mit dem stärksten Sender seines bevorzugten Mobilfunkanbieters verbindet, nicht vorgesehen. Diese Schwachstelle führt dazu, dass eine mobile Basisstation mit der Kennung des Mobilfunkanbieters bei Abhöraktionen dazu verwendet werden kann, um einerseits die IMSI aller Geräte in der Umgebung in Erfahrung zu bringen, die sich automatisch einbuchen, andererseits auch Gespräche mitzuschneiden, da die Basisstation nach erfolgreicher Kopplung über eine GSM-Protokolloption das Endgerät anweisen kann, auf die Verschlüsselung der Audiodaten zu verzichten.<sup>20</sup> Dieses Verfahren und entsprechend präparierte Basisstationen werden als IMSI-Catcher bezeichnet und sind von den Endgeräten, die den GSM-Standard für eine Zulassung vollständig implementieren müssen, weder erkennbar noch zu verhindern. Für „abhörsichere“ Telefonate müsste folglich auf GSM verzichtet und beispielsweise auf IP-Telefonie zurückgegriffen werden.

Eine wirksame Maßnahme zur Sicherung von Transportweg und Daten gegen Ausspähen, auch nach deren Speicherung, ist die Verschlüsselung. Neben der im Abschnitt 6.5.1 erwähnten symmetrischen Verschlüsselung von Daten auf einer Partition besitzen bei einer Übertragung zwischen Sender und Empfänger die asymmetrischen Verschlüsselungsver-

---

<sup>20</sup> Vgl. Wikipedia 2013e.

fahren eine höhere Relevanz, da bei diesen das Problem des geheimen Schlüsselaustauschs durch den als potenziell unsicher erkannten Transportweg entschärft wird.

Bei asymmetrischen Verfahren wird für jeden Kommunikationsteilnehmer ein Schlüsselpaar erstellt. Dieses besteht aus

1. einem geheimen (privaten) Schlüssel,
2. einem öffentlichen Schlüssel.

Die beiden Schlüssel werden komplementär zueinander erzeugt. Etwas vereinfacht dargestellt dient der öffentliche Schlüssel zum Verschlüsseln von Daten, die aber mit dem gleichen Schlüssel nicht wieder in die lesbare Form umgewandelt werden können. Um die Daten wieder zu dechiffrieren, muss der zum öffentlichen Schlüssel passende geheime Schlüssel verwendet werden, der nur seinem Besitzer bekannt ist und niemals weitergegeben werden darf. Auf diese Weise können sowohl der öffentliche Schlüssel als auch die mit diesem verschlüsselten Nachrichten über ein unsicheres Netzwerk transportiert werden, ohne dass unerwünschte Mithörer Zugang zu den Originaldaten erhalten. Bei der Kommunikation zwischen mehreren Teilnehmern muss daher der öffentliche Schlüssel des jeweiligen Empfängers dem Absender der Nachricht bekannt sein, mit diesem verschlüsselt er die Daten, die der Empfänger mit seinem geheimen Schlüssel wieder lesbar machen kann.

Auf ähnliche Weise wird bei Netzdiensten vorgegangen, bei denen die Daten verschlüsselt übertragen werden sollen. In diesem Fall ist der Datenempfänger der Webserver, der ein Formular und einen öffentlichen Schlüssel präsentiert, mit dem die Formulardaten vom Browser verschlüsselt an den Server übertragen werden. Auf dem umgekehrten Weg werden die Daten mit dem geheimen Schlüssel verschlüsselt und mit dem öffentlichen Schlüssel entschlüsselt.

In Wirklichkeit ist der Vorgang etwas komplizierter, da asymmetrische Verschlüsselungsverfahren mit großen Schlüssellängen mitunter langsam und für große Datenmengen schlecht geeignet sind. So wird die asymmetrische Verschlüsselung im Regelfall nur beim initialen Handshake dazu verwendet, einen symmetrischen Schlüssel („Session-Key“) zu vereinbaren und sicher zu übertragen, um die beidseitige Kommunikation anschließend mit diesem symmetrischen Schlüssel durchzuführen. Da die eigentlichen Daten hierbei nur mit einem temporären Schlüssel geschützt werden, der bei einem erneuten Verbindungsauftakt durch einen anderen ersetzt werden kann, ist ein Angriff auf die asymmetrische Verschlüsselung selbst bei Kenntnis der Klartextversion der übertragenen Daten deutlich schwieriger.

Die beiden Implementationen, die für asymmetrische Verschlüsselung von E-Mail und Dateien am häufigsten verwendet werden, sind PGP (Pretty Good Privacy bzw. die GNU-Version GPG, GNU Privacy Guard) sowie SSL (Secure Socket Layer). In den meisten Programmen zur Internetnutzung, in Browzern und Mailclients, ist standardmäßig SSL mit einer komfortablen Schlüsselverwaltung integriert. PGP ist als Mailclient-Plug-in verfügbar. Beim Zugriff auf Webseiten mit verschlüsselter Verbindung wird die SSL-geschützte

Variante des http-Protokolls verwendet (https, worauf im WWW-Browser durch ein farblich gekennzeichnetes Schlosssymbol hingewiesen wird).

Eine systematische Schwäche beim Austausch von öffentlichen Schlüsseln wird bei SSL durch eine Integritätssicherung gelöst (s. a. Abschn. 6.5.4): Wird der öffentliche Schlüssel erstmals an die Kommunikationspartner verschickt, so könnte er durch einen Angreifer, der in der Lage ist, den Transportweg sowohl abzuhören als auch übertragene Daten zu manipulieren, durch einen öffentlichen Schlüssel des Angreifers ausgetauscht werden. Der Versender würde dann, ohne es zu bemerken, die Daten an den falschen Empfänger verschlüsseln, und der Angreifer hätte die Möglichkeit, diese mit seinem passenden privaten Schlüssel abzuhören und ggf. um unbemerkt zu bleiben, die entschlüsselten Daten anschließend mit dem öffentlichen Schlüssel des tatsächlich gemeinten Empfängers wieder zu verschlüsseln und an diesen weiterzuleiten.

Um dieses Szenario zu unterbinden, sollte der Versender beim Empfang des öffentlichen Schlüssels seines Gesprächspartners die Prüfsumme, den sogenannten „Fingerabdruck“ des Schlüssels, durch persönliche Kommunikation verifizieren. Dies setzt aber wiederum eine nicht manipulierbare Datenübertragung zwischen den Gesprächspartnern voraus.

Um diesem Dilemma zu entgehen, werden öffentliche Schlüssel durch eine digitale Unterschrift „beglaubigt“, die von einer vertrauenswürdigen Organisation ausgestellt wird und die im Browser oder Mailclient bereits ihrerseits mit dem digitalen Fingerabdruck und Beglaubigungen ihrer Authentizität durch übergeordnete Organisationen gespeichert ist, sodass eine geschlossene Vertrauenskette entsteht.

Ein öffentlicher Schlüssel mit einer oder mehreren digitalen Unterschriften wird als „Zertifikat“ bezeichnet – es wird sozusagen die Authentizität des Schlüsselinhabers zertifiziert. Die Beglaubigung öffentlicher kryptografischer Schlüssel durch anerkannte Institutionen ist eine, je nach Aufwand bei der Identitätsfeststellung und Sicherheitsstufe, unterschiedlich kostspielige Dienstleistung, die den Komfort bei der Nutzung von SSL-geschützten Diensten durch den Wegfall eigener Nachforschungen über die korrekte Herkunft der öffentlichen Schlüssel verbessert, jedoch die Sicherheit der Verschlüsselung selbst nicht erhöht. Vorfälle, die Zweifel am Sinn einer Zertifikatsausstellung durch Dritte wegen mangelnder Sorgfalt beim Umgang mit Zertifizierungsschlüsseln aufkommen lassen, legen es nahe, im Zweifelsfall doch Eigeninitiative zu ergreifen und die Fingerabdrücke von Kommunikationspartnern selbst durch einen Telefonanruf beim Aussteller zu verifizieren und manuell in der Clientsoftware zu bestätigen.

Abbildung 6.4 zeigt den Dialog des Firefox-Webbrowsers beim Empfang eines nicht durch einen bereits gespeicherten Signierer beglaubigten öffentlichen Schlüssels, Abb. 6.5 den gleichen, wenn auch weniger aussagekräftigen, Dialog in der Android-Version von Firefox.

Allgemein | Details |

**Dieses Zertifikat konnte nicht verifiziert werden, da dem Aussteller nicht vertraut wird.**

---

**Ausgestellt für**

Allgemeiner Name (CN)	knopper.net
Organisation (O)	KNOPPER.NET
Organisationseinheit (OU)	Web Security
Seriennummer	00:85:BA:8B:BA:41:36:E3:B7

**Ausgestellt von**

Allgemeiner Name (CN)	knopper.net
Organisation (O)	KNOPPER.NET
Organisationseinheit (OU)	Web Security

**Validität**

Ausgestellt am	23.06.2008
Läuft ab am	02.05.2018

**Fingerabdrücke**

SHA1-Fingerabdruck	1A:9E:08:2C:21:CA:CE:CE:75:43:05:8D:8B:9E:FE:F5:EB:7B:D9:BE
MD5-Fingerabdruck	F1:D3:8F:3C:75:74:15:AB:35:C8:AD:D2:5F:04:D8:51

**Abb. 6.4** Ausgabe von Zertifikatinformationen und Fingerabdruck in Firefox (Desktopversion 24)

Um ein Schlüsselpaar, bzw. den privaten Schlüssel und den selbst signierten öffentlichen Schlüssel, das Zertifikat, zu generieren, ist das Open-Source-Kommandozeilen-Werkzeug `openssl` hilfreich.

### SSL-Schlüsselgenerierung

```
openssl req -new -x509 -nodes -days 3650 -keyout geheim.pem -out public.pem
```

Das Kommando erzeugt den geheimen Schlüssel in der Datei `geheim.pem`, den dazu passenden öffentlichen Schlüssel in der Datei `public.pem`. Die Option `-x509` signiert den öffentlichen Schlüssel mithilfe des geheimen Schlüssels, wobei die Unterschrift zehn Jahre gültig ist (`-days 3650`). Der geheime Schlüssel wird üblicherweise selbst durch ein Passwort verschlüsselt gespeichert, um ihn vor Diebstahl zu schützen. Die Option `-nodes` unterbindet dies, allerdings sollte der geheime Schlüssel dann wenigstens auf einer verschlüsselten Festplattenpartition gespeichert werden, um einen Diebstahl (und damit die Möglichkeit des unrechtmäßigen Besitzers, die Identität des Schlüsseleigentümers vorzutäuschen) auszuschließen.

Während der Schlüsselgenerierung fragt `openssl` interaktiv Informationen ab, die in das Zertifikat aufgenommen und durch die eigene digitale Unterschrift ebenfalls beglaubigt werden (vgl. Abb. 6.6).



Die Android-Version von Firefox (25) ist wenig informativ...

...erlaubt aber auch das Akzeptieren eines selbstsignierten Schlüssels.

**Abb. 6.5** Ausgabe von Zertifikatinformationen und Fingerabdruck in Firefox (Android-Version 25)

Bei Webseiten wird als *Common Name* der vollständige Rechnername (FQDN) eingegeben, bei Personen der Name sowie im Feld *Email Address* die Mailadresse des Schlüsseleigentümers. Für beide Felder ist nur ein einziger Wert zulässig.

Um den selbst signierten öffentlichen Schlüssel sowie den privaten Schlüssel in einen Mailclient zu importieren, werden beide Schlüssel in ein Archiv im pkcs12-Format gespeichert, der geheime Schlüssel wird hierbei wiederum mit einem Passwort geschützt, das interaktiv abgefragt wird.

#### Archivierung von geheimem Schlüssel und selbst signiertem Zertifikat

```
openssl pkcs12 -export -out datei.p12 -inkey geheim.pem -in public.pem -name "Mein Name"
```

Die Datei datei.p12 kann nun in einem Mailprogramm mit SSL-Support integriert werden.

Damit das selbst signierte Zertifikat mit dem dazugehörigen geheimen Schlüssel vom Mailclient akzeptiert wird, muss zunächst der öffentliche Schlüssel (public.pem im obigen Beispiel) als „Root Zertifikat“ importiert und als Autorität zur Beglaubigung von Personen und Mailadressen akzeptiert werden.

```

knopper@eepc: ~
knopper@eepc:~$ openssl req -new -x509 -nodes -days 3650 -keyout private.pem -out public.pem
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'private.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:Rheinland-Pfalz
Locality Name (eg, city) []:Kaiserslautern
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Knopper.Net
Organizational Unit Name (eg, section) []:Personal_Certificate
Common Name (e.g. server FQDN or YOUR name) []:Klaus Knopper
Email Address [knopper@knopper.net]
knopper@eepc:~$ 

```

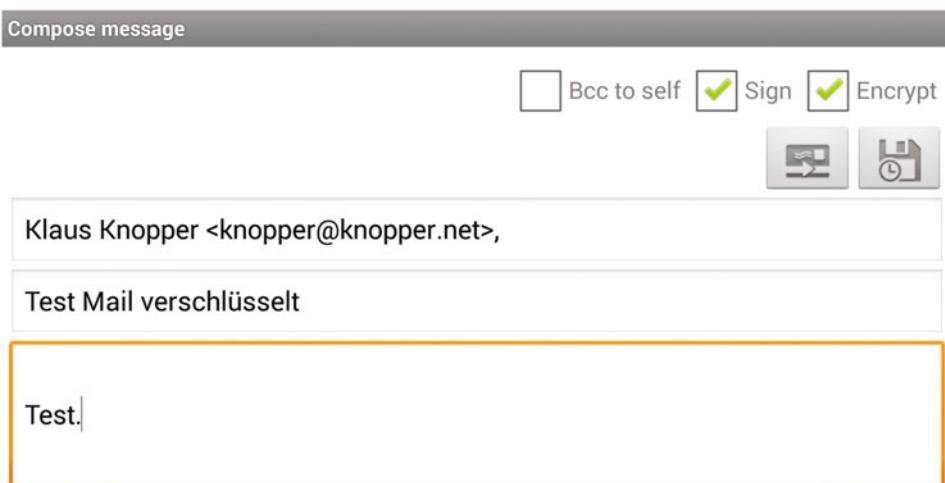
**Abb. 6.6** Aus- und Eingaben (rot) während der Erzeugung des SSL-Schlüsselpaars

Abbildung 6.7 und 6.8 zeigen den Entwurf und die Darstellung einer im S/MIME-Format verschickten, verschlüsselten und signierten E-Mail. Während das Generieren von Schlüsselpaaren und das Einbinden von Zertifikaten und Schlüsseln oft eine größere Herausforderung für den Anwender darstellt, ist die Nutzung von Verschlüsselung und Signatur mit entsprechender Programmunterstützung unter den meisten mobilen Betriebssystemen sehr intuitiv und mit nur geringem Aufwand möglich. Die Nutzung von Verschlüsselung und Signatur in Mailprogrammen wird von Wirtschaftsverbänden und BMI ausdrücklich empfohlen.<sup>21</sup>

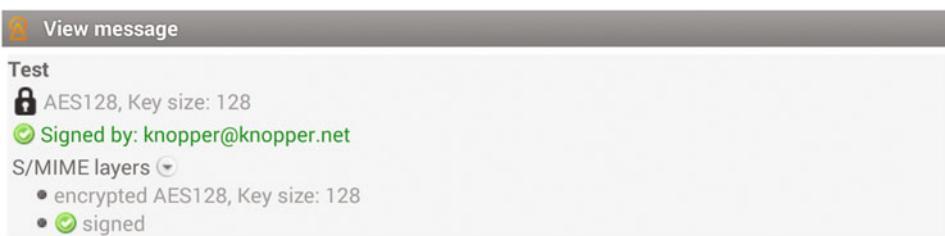
### 6.5.2.3 Speichern von sensiblen Daten in der Cloud

Einige Hersteller mobiler IT-Systeme schränken die Möglichkeiten zur lokalen Datenspeicherung auf Smartphones und Tablets stark ein, wodurch sich die Nutzer gezwungen sehen, auf Netzwerkspeicher auszuweichen. Hierbei sind werbe- oder mehrwertdienstfinanzierte Cloud-Speicher wie Dropbox sehr beliebt, die mit der sicheren, verschlüsselten Übertragung von Daten werben. Dabei wird aber nicht notwendigerweise die Vertraulichkeit gewährleistet, da nach der Datenübertragung Informationen unverschlüsselt oder nur durch entsprechende Gruppenzugriffsrechte geschützt auf der vom Betreiber kontrollierten Hardware gespeichert sind. Oft finden sich auch in den Nutzungsbedingungen von Online-Storage-Anbietern entsprechende Hinweise, dass die Vertraulichkeit und Verfügbarkeit nur eingeschränkt garantiert werden können, weil möglicherweise länderspezifische Gesetze erfordern, dass der Anbieter Verbindungsdaten protokolliert und möglicherweise auch die Entschlüsselung von Daten bei angeordneten Untersuchungen

<sup>21</sup> Vgl. DATEV und Deutschland sicher im Netz e. V. 2013.



**Abb. 6.7** DJIGZO SSL-Mail mit Verschlüsselung und Signatur unter Android



**Abb. 6.8** Empfang einer verschlüsselten und signierten E-Mail mit DJIGZO

durch einen „Generalschlüssel“ (ein weiteres Passwort zum Entschlüsseln des verschlüsselt gespeicherten symmetrischen Schlüssels) möglich ist.

Für sensible persönliche oder unternehmenskritische Daten sind solche Vertragsklau- seln und nicht verifizierbare Sicherheitsmechanismen äußerst ungünstig.

Eine Möglichkeit, sich nicht auf die Vertrauenswürdigkeit des Anbieters bei der Ver- schlüsselung zu verlassen, ist es, die Daten bereits vor der Übertragung selbst mithilfe eines sicheren Verfahrens zu verschlüsseln. So sind Übertragungsweg wie auch die dauerhafte Speicherung zumindest vor dem Ausspähen geschützt, Vertraulichkeit und Integrität blei- ben durch die Eigeninitiative gewährleistet. Der Anbieter kann jedoch die Verfügbarkeit, den Zugang zu den Daten einschränken, wenn die „private“ Verschlüsselung gegen Richt- linien verstößt, die z. B. die stichprobenartige Überprüfung urheberrechtlicher Art von Inhalten, Virenscanner und Filter für unerwünschte Inhalte vorsehen können, was durch die eigenverantwortliche, sichere Verschlüsselung quasi unmöglich gemacht wird.

Durch solche Überlegungen angespornt haben sich providerunabhängige Projekte etabliert, die eine persönlich konfigurier- und absicherbare Cloudlösung zur Integration in Webserver realisieren, mit Verschlüsselung sowohl des Transportweges per SSL/TLS als auch der verschlüsselten Datenhaltung auf den Speichermedien. Hier wäre das Open-Source-Projekt OwnCloud als eine der ersten und aktiv entwickelten Lösungen zu nennen.<sup>22</sup>

### 6.5.3 Verfügbarkeit

Wer sich ein leistungsfähiges Smartphone oder Tablet leistet, wird das Gerät sicher nicht nur zum Telefonieren verwenden, sondern erwartet von diesem persönlichen digitalen Assistenten (PDA) ständige Einsatzbereitschaft und die zuverlässige Speicherung, Verwaltung und Aktualisierung der persönlichen Daten. Schutz vor mechanischen Defekten oder mechanisch bedingten Ausfällen bieten entsprechende stabile Schutzhüllen, jedoch ist zur Funktionserhaltung ebenso ein Schutz der Software und der Daten notwendig. Klassische Bedrohungen der Verfügbarkeit sind

- Diebstahl von Hard- und Software,
- Versehentliches oder von Angreifern beabsichtigtes Löschen oder unbrauchbar Machen von Daten oder Sperrung des Zugriffs auf Daten.

Lösungsansätze gegen Diebstahl der Hardware bieten mittlerweile verschiedene Dienstleister im Zusammenhang mit Wartungs- und Bandbreitenverträgen als Device-Tracking an, d. h. das Gerät kann, wenn es entwendet wurde, durch automatischen Versand seiner GPS-Koordinaten oder WLAN-Adresse beim Einbuchen in ein Netzwerk lokalisiert werden. Eine Möglichkeit, die den Mobilfunkanbietern offen steht, ist auch die Lokalisierung des Gerätes mithilfe einer „Stillen SMS“<sup>23</sup>, die weder für den Benutzer sichtbar ist, noch durch ein Signal registriert wird. Dabei kann sich das Gerät auch im Schlafzustand befinden, da die Sendeeinheit bei eingebuchter SIM-Karte diese Antwort selbsttätig senden kann, ohne dafür die Hilfe des Betriebssystems zu benötigen, welches sich im Stromsparmodus befindet. Diese und andere zum Wiederauffinden von Geräten und Nutzern sicherlich praktische Mechanismen bergen jedoch die Gefahr des Missbrauchs, denn was im Notfall funktioniert, kann auch zu anderen Zwecken genutzt werden, wie dem bei entsprechender Hardwareunterstützung unbemerkten Einschalten des Gerätes mithilfe des Netzbetreibers bis hin zu ebenfalls unbemerkten Film- und Tonaufnahmen im vom Benutzer angenommenen Schlafzustand des Gerätes, was wiederum den Sicherheitsaspekt „Vertraulichkeit“ tangiert.

---

<sup>22</sup> Vgl. Wikipedia 2013f.

<sup>23</sup> Vgl. Wikipedia 2013g.

Oft wird der monetäre Aspekt des Hardwareverlustes gegenüber dem Verlust von wichtigen persönlichen und beruflichen Datensätzen überbewertet. Für die Reproduktion von Arbeitsergebnissen, wenn überhaupt möglich, ist der Zeit- und Kostenaufwand oft um ein Vielfaches höher als der Ersatz des Gerätes. Persönliche Aufnahmen und Erinnerungen haben für den Nutzer ebenfalls einen hohen Stellenwert, und deren Abhandenkommen stellt ebenfalls eine Belastung dar, auch wenn im Zuge von Sicherungsmaßnahmen des Sicherheitsaspektes „Vertraulichkeit“ die entsprechenden Datenpartitionen bereits durch eine starke Verschlüsselung gegen das Ausspähen durch unautorisierte Personen gesichert wurden. Um dem Datenverlust vorzubeugen, greift auch bei Smartphones und Tablets das tägliche (oder in bedarfsgerechten Intervallen durchgeführte) Backup, wofür für alle Betriebssysteme entsprechende Apps zur Verfügung stehen. Firefox bietet eine Online-synchronisation von Lesezeichen, Einstellungen, Zugangsdaten und Cookies mit anderen Firefox-Browsern an. Für die bei den meisten Geräten als „Galerie“ oder „Fotoalbum“ bezeichneten Ordnungs- und Archivierungsprogramme für Bild-, Film- und Tonaufnahmen werden Online-Fotoalben und Cloud-Speicher zur automatischen Übertragung angeboten, die unter dem im vorigen Abschnitt genannten Aspekt der Vertrauenswürdigkeit des Anbieters in Betracht gezogen werden können.

Für einen Ausfall des Betriebssystems sind auf den meisten mobilen Geräten herstellerspezifische Recoverysysteme installiert, die ein Wiederaufspielen der Systemsoftware ermöglichen, sofern hierfür ein Backup existiert oder ein Systemimage beim Hersteller erhältlich ist. Oft ist jedoch mit dem erneuten Aufspielen der Systemsoftware auch ein Verlust der Nutzerdaten verbunden, da Recoverysysteme meist den geräteinternen Flashspeicher neu partitionieren und löschen, um den „Werkszustand“ wiederherzustellen. Die Recoverysoftware „clockworkmod“, die auch vom Projekt Cyanogenmod empfohlen wird, unterstützt auch partitionsweise, inkrementelle Archivierung und Restaurierung von Daten sowie Aktualisierungen der Systemsoftware über, optional signierte, ZIP-Archive.

#### **6.5.4 Integrität**

Schadsoftware (Viren, Würmer, Trojaner) betrifft alle drei Aspekte der Systemsicherheit, beeinträchtigt aber vor allem die Integrität des Systems selbst. Mit zunehmender Verbreitung mobiler Betriebssysteme ist auch die Motivation der Schadsoftwareprogrammierer gewachsen, ihre Software möglichst breit zu streuen. Dabei sind weniger Systemschwächen des Betriebssystemkerns, sondern die Anwendersoftware das Einfallstor. Häufig werden Trojaner in Anwendersoftware versteckt, die kostenlos angeboten wird. Das vermeintliche neue Spiel oder die wesentlich günstiger als im offiziellen App-Store angebotene Produktivsoftware verleiten den Anwender dazu, sich ohne Rückfrage bezüglich der Herkunft und Vertrauenswürdigkeit der App und unter ungelesener Bestätigung aller Sicherheitsabfragen ein Softwarepaket zu installieren, das sein Smartphone oder Tablet unbemerkt zum Teil eines Botnetzes mit Fernsteuerung werden lässt.

Bereits installierte Schadsoftware zu erkennen, ist nicht einfach, da sich die betroffenen Programme bis zu ihrer Aktivierung durch den Angreifer meist passiv und unauffällig verhalten. Lediglich durch nach außen offene Netzwerkports oder unregelmäßigen Verbindungsaufbau zu einem Server im Internet verraten sich Trojaner gelegentlich.

Leider werden diese offenen Ports nicht regelmäßig automatisch überprüft, obwohl es unter allen Unix-Derivaten hierfür systemeigene Hilfsmittel gibt.

#### Netzwerkaktivität: Verbindungen nach außen und laufende Netzdienste auflisten

```
root@android:/ # netstat -an | grep ESTABLISHED | grep -v 127.0.0.1:47470
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	127.0.0.1:47470	0.0.0.0:*	LISTEN
tcp	0	0	10.0.0.121:39020	173.194.70.102:80	ESTABLISHED
tcp	0	0	10.0.0.121:49497	173.194.112.99:80	ESTABLISHED

Im gezeigten Beispiel des im Funktionsumfang etwas reduzierten `netstat`-Kommandos unter Android ist lediglich ein intern (127.0.0.1) erreichbarer Serverdienst (Status LISTEN) auf Port 47470 erkennbar, die anderen Verbindungen über WLAN (IP-Adresse 10.0.0.121) sind Verbindungen zu http-Servern von Google über Port 80. Die Update- und Synchronisationsfunktionen der Smartphone-Betriebssysteme bauen in regelmäßigen Abständen Verbindungen zu Servern des Herstellers auf, was nicht in jedem Fall durch das Abschalten der entsprechenden Funktionen in den Einstellungen zumindest reduziert werden kann.

Bei ständig offenen Netzwerkverbindungen nach außen wäre die Zieladresse zu überprüfen, da Trojaner sich fast immer mit einem Steuerungsrechner verbinden, um Code nachzuladen oder Aufträge für neue Angriffe von dort entgegenzunehmen. Auch die Verbindung zu „außergewöhnlichen“ Ports (die Ports 1–1000 sind für standardisierte Internetdienste vorgesehen, höher liegende Ports hingegen für herstellerspezifische Dienste) oder Verbindungen über das eher für Streaming gebräuchliche UDP-Protokoll und entsprechender Datenverkehr können auf eine Infektion des Systems hinweisen.

Eine Möglichkeit, unerwünschte Netzwerkzugriffe von Apps zu verhindern, ist ein Paketfilter oder Firewall, der nur Verbindungen zu bestimmten IP-Adressen oder Netzwerkports zulässt oder bestimmte nicht erwünschte Adressen sperrt. Die Einrichtung eines Paketfilters ist je nach Sicherheitsanspruch eine komplexe Aufgabe. Die meisten mobilen Betriebssysteme haben zumindest einen einfachen Filter als Voreinstellung, der Verbindungen von außen nur zu bestimmten Diensten zulässt.

### Iptables-Paketfilter-Konfiguration in Cyanogenmod (Ausschnitt)

```
root@android:/ # iptables -L -v

Chain INPUT (policy ACCEPT 145K packets, 202M bytes)
pkts bytes target        prot opt in     out      source          destination
147K  203M bw_INPUT    all  --  any    any     anywhere       anywhere
147K  203M fw_INPUT    all  --  any    any     anywhere       anywhere

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target        prot opt in     out      source          destination
0     0 oem_fwd         all  --  any    any     anywhere       anywhere
0     0 fw_FORWARD      all  --  any    any     anywhere       anywhere
0     0 bw_FORWARD      all  --  any    any     anywhere       anywhere
0     0 natctrl_FORWARD all  --  any    any     anywhere       anywhere

Chain OUTPUT (policy ACCEPT 63921 packets, 7790K bytes)
pkts bytes target        prot opt in     out      source          destination
66716 8084K oem_out     all  --  any    any     anywhere       anywhere
66716 8084K fw_OUTPUT   all  --  any    any     anywhere       anywhere
66716 8084K bw_OUTPUT   all  --  any    any     anywhere       anywhere
```

INPUT bezeichnet die Regelkette für eingehende, OUTPUT für ausgehende und FORWARD für an andere Geräte weitergeleitete Netzwerkpakete. Die Verwendung von weiteren Regelketten als Ziele (die meisten davon sind standardmäßig leer) erleichtern die schnelle Konfigurationsänderung ohne Neustart, beispielsweise bei der Aktivierung von Tethering zur Weitergabe des eigenen Internetzugangs als WLAN-Accesspoint, hier werden bei Bedarf nur Regeln eingefügt, die Forwarding und Masquerading (dynamische Adressumwandlung der Clients, Network Address Translation) einschalten. Aufgrund der von einigen Apps benötigten variablen Netzwerkports wird jedoch auf restriktivere Einstellungen wie Sperre von Netzwerzkugriffen bis auf erlaubte Ports verzichtet.

Einen bereits hohen Schutzfaktor hat die Überprüfung von digitalen Signaturen. Das Verfahren basiert auf der asymmetrischen Verschlüsselung wie im Abschnitt „Vertraulichkeit“ bereits in der Funktionsweise erklärt. Bei der digitalen Signatur wird die Verschlüsselung jedoch in anderer Weise genutzt:

1. Bilden einer Prüfsumme (Hashfunktion) über die zu signierenden Daten.

Eine Hashfunktion erzeugt eine Zahl, die für ein Dokument oder eine Datei eindeutig ist, d. h. es ist eine sehr schwer lösbar kryptografische Aufgabe, zu einem Hashwert eine zweite Datei mit unterschiedlichem Inhalt aber mit dem gleichen Wert zu finden. Beispiel: Der Text „Hallo, Welt“ (ohne Zeilenumbruch) ergibt als SHA256-Hashwert die Zahl 48aad0dacb88f8ec5ce254c7d539f7117a78cd30afa65c74b2e50caf057f27fd, während sich beim gleichen Text mit einem Punkt dahinter („Hallo, Welt.“) der Hashwert zu 192b9f7eadb6f1c566ccb0ccb84f1465437e12eece73a78c16154b1e167833f3 ergibt.

2. Der von den zu signierenden Daten gebildete Hashwert wird nun mit dem privaten (!) Schlüssel des Autors verschlüsselt und zusammen mit dem öffentlichen Schlüssel bzw. Zertifikat in einer Datei gespeichert und beispielsweise bei der Übertragung per E-Mail als zusätzlicher Anhang zu den Daten mitgeliefert. Die von den eigentlichen Daten getrennte Lieferung der verschlüsselten Prüfsumme und des Zertifikats wird als „detached signature“ (losgelöste Unterschrift) bezeichnet.
3. Der Empfänger der Daten kann nun mit dem öffentlichen Schlüssel des Autors, dessen Zugehörigkeit zum Autor er zuvor zweifelsfrei festgestellt haben muss, die verschlüsselte Prüfsumme wieder entschlüsseln. Wenn die Entschlüsselung erfolgreich ist und der Hashwert der übermittelten Daten immer noch mit dem entschlüsselten Hashwert aus der Signatur identisch ist, wurden die Daten nicht verändert.

Die Softwareverwaltung bei mobilen Betriebssystemen kann mit dem gleichen Verfahren die Inhalte von Softwarepaketen überprüfen, wobei hier meist aus Gründen der einfacheren Handhabung die digitale Signatur im gleichen Archiv untergebracht ist wie die überprüften und mit Hashwert versehenen Daten. Der Benutzer wird beim Abweichen der Prüfsummen darauf hingewiesen werden, dass der Inhalt des Softwarepaketes entweder unvollständig ist oder manipuliert wurde. Restriktiv eingestellte Systeme verhindern sogar die Installation von Software ganz, die nicht per digitale Signatur verifiziert werden kann oder bei der die öffentlichen Schlüssel der als zulässig betrachteten Softwareherausgeber nicht im System hinterlegt sind. Dabei ist die digitale Signatur alleine noch kein Garant für die Vertrauenswürdigkeit des Autors oder die Qualität der Software, sondern dient nur der Erkennung von Veränderungen gegenüber der vom Autor signierten Version.

Die über die Updatefunktion des jeweiligen mobilen Betriebssystems oder die in App-Stores erhaltenen Softwarepakete sind i. d. R. mit Signaturen versehen, deren zugehörige öffentliche Schlüssel zur Verifikation den Installationsprogrammen bekannt sind. Möchte der Nutzer selbst erstellte Softwarepakete oder solche installieren, die mit keinem vertrauenswürdigen Schlüssel signiert sind, so muss er ggf. in den Einstellungen die entsprechende Überprüfung abschalten (s. Abb. 6.9).



## Sicherheit

---

### GERÄTEVERWALTUNG

---

#### Geräteadministratoren

Geräteadministratoren abrufen oder deaktivieren

---

#### Unbekannte Herkunft

Installation von Apps aus unbekannten Quellen zulassen

---

#### Apps verifizieren

Installation schädlicher Apps blockieren oder Warnung senden

---

**Abb. 6.9** Einstellung der Signaturprüfung in Android

---

## 6.6 Zertifizierungen

Um Sicherheit planbar und verifizierbar zu gestalten, arbeiten Gremien neben den rein technischen Aspekten und Entwicklungen der IT-Sicherheit auch an der Standardisierung in Aufbau und Betrieb sicherheitsrelevanter Organisations- und Infrastruktur.

Bei Zertifizierungen nach einer Norm, die für das zu zertifizierende Unternehmen oft sehr arbeits- und kostenintensiv sind, werden nur die in der Norm definierten Anforderungen mithilfe von Nachweisen und ggf. Audits vor Ort überprüft.

Einige Zertifikate enthalten keine konkreten zu prüfenden Sachverhalte, sondern es wird zunächst zwischen dem evaluierenden Institut und dem zu evaluierenden Unternehmen eine Liste von Anforderungen und Produkten aufgestellt, welche zu prüfen sind, und nur für diese werden Zusicherungen erteilt. So sind die nach Common Criteria („Gemeinsame Kriterien“) erworbenen Zertifikate keineswegs untereinander vergleichbar, noch enthalten sie eine Aussage über Datenschutz und die generelle Behandlung von Sicherheitsaspekten, da zunächst Sicherheitsvorgaben für Produkte oder Dienstleistungen des Kandidaten individuell erarbeitet und anschließend überprüft werden (Abb. 6.10).<sup>24</sup>

Für den Anwender mobiler Geräte sind Sicherheitsnormen weniger interessant, für die Hersteller der Geräte hingegen sind sie mitunter Voraussetzung für die Erteilung einer Handelsgenehmigung. Zu erwähnen sind hier auch die IT-Grundschutz-Kataloge (ehemals: „IT Grundschutzhandbuch“) des Bundesamtes für Sicherheit in der Informationstechnik als Leitfaden für die Umsetzung von Standardverfahren zur IT-Sicherheit im Unternehmen.<sup>25</sup>

---

<sup>24</sup> Vgl. Common Criteria 2013.

<sup>25</sup> Vgl. BSI 2013.

Norm	Inhalt
ISO/IEC 15408	Common Criteria
ISO/IEC 27001	Informationssicherheits-Managementsysteme (ISMS)
ISO/IEC 27002	Informationssicherheits-Managementsysteme (ISMS)
BSI-Standard 100-1	Managementsysteme für Informationssicherheit
BSI-Standard 100-2	IT-Grundschutz-Vorgehensweise [BSI2]
BSI-Standard 100-3	Risikoanalyse auf der Basis von IT-Grundschutz [BSI3]

**Abb. 6.10** Ausschnitt aus „Normen zur IT-Sicherheit“. (Vgl. Bitkom, 2013)

## 6.7 Handlungsempfehlungen

Aus der Gefahrenanalyse und den technischen Maßnahmen zur Verbesserung der Sicherheit in den vorigen Abschnitten lassen sich allgemeine Verhaltensregeln im Umgang mit sensiblen Daten in unsicheren Netzwerken extrahieren.

- Vermeidung von potenziellen Angriffspunkten hat Priorität vor nachträglicher Schadsoftwareerkennung und -entfernung → alle nicht benötigten Dienste und Widgets abschalten oder entfernen.
- Gefahren für die Sicherheit gespeicherter Daten durch die Nutzung von Apps lassen sich durch Prüfung der Zugriffsrechte und Kombinationen von Zugriffsrechten erkennen und durch Vermeiden nichtvertrauenswürdiger Apps reduzieren.
- Verschlüsselte Zugangsmethoden und Dienste sowohl client- als auch serverseitig sollen gegenüber unverschlüsselten Diensten bevorzugt werden, d. h. Verwendung SMT-PS/SMP + TLS statt SMTP für den Mailversand, IMAPS statt IMAP und POP3S statt POP3 für den Mailempfang, HTTPS statt HTTP für die Formularübermittlung auf Webseiten, SSH statt TELNET für den Remotezugang, Ende-zu-Ende-Verschlüsselung und digitale Signatur von E-Mail.
- Die eigenverantwortliche Prüfung der Herkunft und das Erkennen von fragwürdigen Klauseln im Lizenzvertrag sind entscheidend für die Bewertung der Vertrauenswürdigkeit neu zu installierender Apps. Popularität und hohe Verbreitung einer App sind hingegen kein Kriterium für die Sicherheit.
- Der Zugang zum mobilen Gerät ist ebenso zu schützen wie die darauf gespeicherten Daten, mithilfe eines zuverlässigen und nicht leicht zu umgehenden Authentifikationsverfahrens.
- Regelmäßige Backups auf einen externen, möglichst verschlüsselten oder für andere unzugänglichen Datenträger sorgen für die Datenverfügbarkeit auch bei Verlust oder Defekt des Gerätes.
- Quelloffene Systeme und Anwendungen mit Analyse- und Anpassungsmöglichkeit schaffen Vertrauen in Wartbarkeit und Nachhaltigkeit.

## Literatur

- Bundesamt für Sicherheit in der Informationstechnik (BSI): IT-Grundschutz-Kataloge. [https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKataloge/Inhalt/\\_content/kataloge.html](https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKataloge/Inhalt/_content/kataloge.html) (2013). Zugriffen 11. Okt. 2013
- Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Bitkom): Kompass der IT-Sicherheitsstandards. <http://www.kompass-sicherheitsstandards.de/> (2013). Zugriffen 11. Okt. 2013
- Common Criteria: The Common Criteria. <http://www.commoncriteriaportal.org/> (2013). Zugriffen 11. Okt. 2013
- Crispin, M.: Internet message access protocol, RFC3501. <http://tools.ietf.org/html/rfc3501> (2003). Zugriffen 11. Okt. 2013
- Cyanogenmod: Community-Seite. <http://www.cyanogenmod.org/> (2013). Zugriffen 11. Okt. 2013
- DATEV, Deutschland sicher im Netz e. V.: Verschlüsselung von Emails, Leitfaden zur E-Mail-Sicherheit für Unternehmen. <https://www.sicher-im-netz.de/files/documents/Leitfaden-E-Mail-Verschlüsselung.pdf> (2013). Zugriffen 11. Okt. 2013
- Diedrich, O.: CyanogenMod ist jetzt eine Firma. <http://www.heise.de/open/meldung/CyanogenMod-ist-jetzt-eine-Firma-1960895.html> (2013). Zugriffen 11. Okt. 2013
- Friedewald, M., Raabe, O., Georgieff, P., Koch, D. J., Neuhäusler, P.: Ubiquitäres Computing. Das „Internet der Dinge“. Grundlagen, Anwendungen, Folgen. Edition sigma, Berlin (2010) (Studien des Büros für Technikfolgen-Abschätzung beim Deutschen Bundestag; 31, 2010)
- Google Inc.: Android security overview. <http://source.android.com/devices/tech/security/> (2013a). Zugriffen 11. Okt. 2013
- Google Inc.: Android software stack. <http://source.android.com/devices/tech/security/images/image00.png> (2013b). Zugriffen 11. Okt. 2013
- Google Inc.: Android Manifest. <https://developer.android.com/guide/topics/manifest/manifest-intro.html> (2013c). Zugriffen 11. Okt. 2013
- Llamas, R., Reith, R., Shirer, M.: Apple cedes market share in smartphone operating system market as android surges and windows phone gains, according to IDC. <http://www.idc.com/getdoc.jsp?containerId=prUS24257413> (2013). Zugriffen 11. Okt. 2013
- Myers, J., Rose, M.: Post office protocol – version 3, RFC1939. <http://tools.ietf.org/html/rfc1939> (1996). Zugriffen 11. Okt. 2013
- National Security Agency (NSA): Security enhanced Linux. <http://www.nsa.gov/research/selinux/index.shtml> (2013). Zugriffen 11. Okt. 2013
- Postel, J. B.: Simple mail transfer protocoll, RFC821. <http://tools.ietf.org/html/rfc821> (1982). Zugriffen 11. Okt. 2013
- Schmidt, J.: NSA und GCHQ: Großangriff auf Verschlüsselung im Internet. <http://www.heise.de/security/meldung/NSA-und-GCHQ-Grossangriff-auf-Verschlüsselung-im-Internet-1950935.html> (2013a). Zugriffen 11. Okt. 2013
- Schmidt, J.: Entsperrmuster mal eben gehackt. <http://www.heise.de/security/artikel/Androids-Entsperrmuster-mal-eben gehackt-1830073.html> (2013b). Zugriffen 11. Okt. 2013
- Schmidt, J.: Apples Touch ID des iPhone 5S schon gehackt. <http://www.heise.de/newstickermeldung/Apples-Touch-ID-des-iPhone-5S-schon-gehackt-1964077.html> (2013c). Zugriffen 11. Okt. 2013
- Wikipedia: Edward Snowden. [https://de.wikipedia.org/wiki/Edward\\_Snowden](https://de.wikipedia.org/wiki/Edward_Snowden) (2013a). Zugriffen 11. Okt. 2013
- Wikipedia: ARM Architektur. <http://de.wikipedia.org/wiki/ARM-Architektur> (2013b). Zugriffen 11. Okt. 2013

Wikipedia: Advanced encryption standard. [http://de.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard](http://de.wikipedia.org/wiki/Advanced_Encryption_Standard) (2013c). Zugegriffen 11. Okt. 2013

Wikipedia: Android software development. [http://en.wikipedia.org/wiki/Android\\_software\\_development](http://en.wikipedia.org/wiki/Android_software_development) (2013d). Zugegriffen 11. Okt. 2013

Wikipedia: IMSI catcher. <http://de.wikipedia.org/wiki/IMSI-Catcher> (2013e). Zugegriffen 11. Okt. 2013

Wikipedia: OwnCloud. <http://de.wikipedia.org/wiki/OwnCloud> (2013f). Zugegriffen 11. Okt. 2013

Wikipedia: Stille SMS. [http://de.wikipedia.org/wiki/Stille\\_SMS](http://de.wikipedia.org/wiki/Stille_SMS) (2013g). Zugegriffen 11. Okt. 2013

---

# Business Case I: Mobile Applikationen zur persönlichen Finanzplanung

7

Marius Schönberger

*Konzeption und Entwicklung einer mobilen Applikation zur Baufinanzierung*

---

## Zusammenfassung

Das vorliegende Kapitel beschäftigt sich allgemein mit der Entwicklung mobiler Applikationen und im Speziellen mit den Potenzialen und Herausforderungen einer mobilen Applikation im Markt für Baufinanzierungen. Das Hauptziel des Kapitels besteht in der Entwicklung einer lauffähigen mobilen Applikation zur Baufinanzierung. Die Beschreibung des Softwareentwicklungsprozesses, die Auswahl notwendiger Entwicklungswerkzeuge, die Betrachtung und der Vergleich gegenwärtiger mobiler Endgeräte, die Analyse bestehender Konkurrenzprodukte, die Erklärung finanzmathematischer Grundbegriffe sowie die Darstellung des Ablaufs einer Immobilienfinanzierung bilden die weiteren Nebenziele der nachfolgenden Ausführungen. Demnach liegen folgende Fragestellungen zugrunde:

- Wie sieht der Projektlebenszyklus bei der mobilen Applikationsentwicklung aus?
- Welche Herausforderungen bestehen im Markt für Baufinanzierungen?
- Wie kann eine Applikation zur Baufinanzierung Kreditinstitute oder private Personen bei finanzpolitischen Fragestellungen unterstützen?
- Welche Potenziale entstehen durch eine Applikation zur Baufinanzierung für Unternehmen oder private Personen?
- Wie kann eine Applikation zur Baufinanzierung einen Mehrwert gegenüber der konventionellen Beratung für Baufinanzierungen darstellen?

---

M. Schönberger (✉)  
Fachbereich Betriebswirtschaft, Fachhochschule Kaiserslautern,  
Zweibrücken, Deutschland  
E-Mail: marius.schoenberger@fh-kl.de

Auf die Beantwortung der oben aufgeführten Forschungsfragen wird in den nachfolgenden Kapiteln detailliert eingegangen.

---

## 7.1 Motivation und Relevanz

Die technologische Weiterentwicklung mobiler Endgeräte sowie die fallenden Kosten für die Nutzung mobiler Datendienste haben dazu geführt, dass Mobile Banking in den vergangenen zwei Jahren an Kundenakzeptanz gewonnen hat. Zur Verbesserung der Kommunikationsmöglichkeiten sowie zur Stärkung der Kundenbindung planten im Jahr 2010 mehr als 40 % der Kreditinstitute in Deutschland, in mobile Technologien zu investieren.<sup>1</sup> Obwohl das Serviceangebot durch die Bereitstellung neuartiger Technologien in den letzten Jahren stark erweitert wurde, schöpfen viele Finanzunternehmen die Möglichkeiten des Mobile Banking noch nicht in vollem Umfang aus.<sup>2</sup> Banken und Versicherungsinstitute bieten zwar zahlreiche mobile Applikationen an, jedoch können nicht in allen Fällen umfangreiche Bankingfunktionen genutzt werden. Der Leistungsumfang beschränkt sich eher auf das Aufzeigen nahe gelegener Banken oder Geldautomaten oder die Auflistung verschiedener Ansprechpartner.<sup>3</sup>

Diese Erkenntnisse werden ebenso durch die Schweizer Studie „Mobile Apps for Banking“ der Research Firma „My Private Banking“ bestätigt. Die Studie umfasst fast 200 analysierte und bewertete mobile Applikationen von weltweit 50 führenden Banken. Ergebnisse der Studie zeigten, dass nur eine geringe Anzahl von Banken ein umfassendes Angebot von aufeinander abgestimmten Applikationen für Privatkunden anbieten. Insgesamt bieten zwei Drittel der betrachteten Bankinstitute nur rudimentäre Anwendungen mit begrenztem Funktionsumfang an. Die Studie ergab weiterhin, dass einige globale Banken keine mobilen Applikationen in ihr Produktpotfolio aufgenommen haben.<sup>4</sup> Die Forscher stellten weiterhin folgende Schwachstellen am Angebot mobiler Applikationen fest:<sup>5</sup>

- Die Mehrheit der Applikationen ist nur für das iPhone erhältlich.
- Viele Applikationen bieten nur Basisfunktionen an.
- Nur eine Minderheit des Angebotes bietet nützliche Inhalte an.
- Banken bieten nur selten Funktionen und Werkzeuge zur persönlichen Finanzplanung und Bewertung an.
- Viele Applikationen weisen Mängel in Bezug auf Sicherheit und Datenschutz auf.

---

<sup>1</sup> Vgl. Schilling und Reuter 2010, S. 34–37.

<sup>2</sup> Vgl. Capgemini 2012 S. 32.

<sup>3</sup> Vgl. Schilling und Reuter 2010, S. 34–37.

<sup>4</sup> Vgl. My Private Banking GmbH 2013.

<sup>5</sup> Vgl. My Private Banking GmbH 2013.

Eine Vielzahl der in der Studie analysierten Applikationen können nur auf dem iPhone oder iPad genutzt werden. Zudem mangelt es hauptsächlich an spezifischen Applikationen für das Android- oder Windows-Betriebssystem. Viele Applikationen bieten im Vergleich zu einer klassischen Finanzberatung keinen Mehrwert. Die untersuchten Applikationen dienten oftmals nur dem Auffinden eines nahe gelegenen Kreditinstitutes oder Geldautomaten. Ebenso wird die Darstellung umfassender Produktinformationen oder die Möglichkeit zum Abruf aktueller Finanznachrichten nur von wenigen Applikationen unterstützt. Eines der größten Probleme der betrachteten Applikationen bildet die schlechte oder kaum realisierte Datensicherheit. Vor allem in Bezug auf die Übertragung persönlicher Finanzdaten müssen angemessene Sicherheitsmechanismen entwickelt werden.<sup>6</sup>

Vor diesem Hintergrund beschreibt das vorliegende Kapitel die Entwicklung und Planung einer mobilen Schnittstelle zwischen Kreditinstituten und Kunden zur Ermittlung einer Baufinanzierung. Bei der Umsetzung wird versucht, die zuvor genannten Schwachstellen bestehender Applikationen zu beheben.

---

## 7.2 Planungs- und Analysephase

Der Prozess zur Entwicklung einer Software ist ein komplexes Verfahren, dem eine aufwendige und zeitlich intensive Planungs- und Analysephase vorhergehen muss. In diesem Sinne erstreckt sich der Lebenszyklus eines Softwareproduktes von der initialen Idee über die Planung, Entwicklung und Einführung der Software, über die Wartung oder eine etwaige Weiterentwicklung der Anwendung bis hin zur abschließenden Abschaffung. Vor diesem Hintergrund sowie in Zusammenhang mit der Erstellung einer mobilen Baufinanzierungs-Applikation stellen sich folgende Fragen:

1. Welches Ziel soll mit der Entwicklung der Applikation erreicht werden?
2. Können bereits ähnliche oder identische Softwareprodukte am Markt registriert werden?
3. Welche Rahmenbedingungen bestehen während der Applikationsentwicklung?
4. Welche Methoden sollen für die Planung der Applikation eingesetzt werden?
5. Welchen Aufgaben- bzw. Funktionsumfang soll die Applikation aufweisen und wie können diese umgesetzt werden?
6. Für welche Endgeräte und Betriebssysteme soll die Applikation entwickelt werden?
7. Welches Vorgehensmodell zur Applikationsentwicklung soll eingesetzt werden?
8. In welcher Programmiersprache und auf welcher Entwicklungsumgebung soll die Applikation realisiert werden?

---

<sup>6</sup> Vgl. My Private Banking GmbH 2013.

9. Welche Aspekte sind während des operativen Betriebs wichtig?
10. Welche Methoden sollen für die Validierung und Evaluation der Applikation eingesetzt werden?

Die Entwicklung von Softwareanwendungen zielt darauf ab, für Lösungen bestimmter Aufgabenbereiche computergestützte Werkzeuge bereitzustellen. Ziel der Analysephase ist es, den zukünftigen Aufgabenbereich abzugrenzen, die relevanten Problemstellungen zu identifizieren sowie eine entsprechende Lösung für das Problem zu entwickeln.<sup>7</sup> Aufbauend auf dem Ergebnis der Analysephase werden Anforderungen definiert, welche an die Software gestellt werden (vgl. Abschn. 7.3). Die problemorientierte Erörterung des abgegrenzten Aufgabenbereichs umfasst weiterhin die Analyse der Istsituation, die Modellierung der Problembeschreibung sowie die Auswahl geeigneter Instrumente und Werkzeuge zur Softwareentwicklung.

Der Planungsprozess zur Softwareentwicklung dient der gedanklichen Vorbereitung und Strukturierung des zu entwickelnden Softwareprojektes. Ziel der Planungsphase ist die Erstellung eines Projektplanes, welcher Auskunft über die zeitliche Terminierung des Gesamtprojektes sowie über die Abhängigkeiten der einzelnen Projektphasen innerhalb des Entwicklungsprozesses gibt. Entsprechend der DIN 69901 enthält ein Projektplan Zielvorgaben sowie personelle, sachliche, finanzielle und zeitliche Abgrenzungen und lässt sich mittels verschiedener Methoden und Instrumente aus dem Bereich des Projektmanagements abbilden.<sup>8</sup>

Im Zuge der Istanalyse werden in einem ersten Schritt aktuelle Herausforderungen im Markt zur Baufinanzierung genannt, aus denen Gründe für die Entwicklung einer mobilen Applikation zur Baufinanzierung resultieren sollen. Daran anknüpfend werden in einem weiteren Schritt ausgewählte Vorgehensmodelle zur Softwareentwicklung einander gegenübergestellt und analysiert sowie ein der Problemstellung entsprechendes Modell ausgewählt. Im Rahmen der Planungsphase wird auf Grundlage des ausgewählten Vorgehensmodells die inhaltliche und zeitliche Gliederung des Entwicklungsprozesses abgeleitet und terminiert. Die Bewertung potenzieller Entwicklungsumgebungen zur Applikationsentwicklung sowie die Bestimmung einer geeigneten Alternative bilden den Abschluss des vorliegenden Kapitels. Innerhalb der nachfolgenden Ausführungen werden die oben aufgeführten Fragen nochmals aufgegriffen und beantwortet.

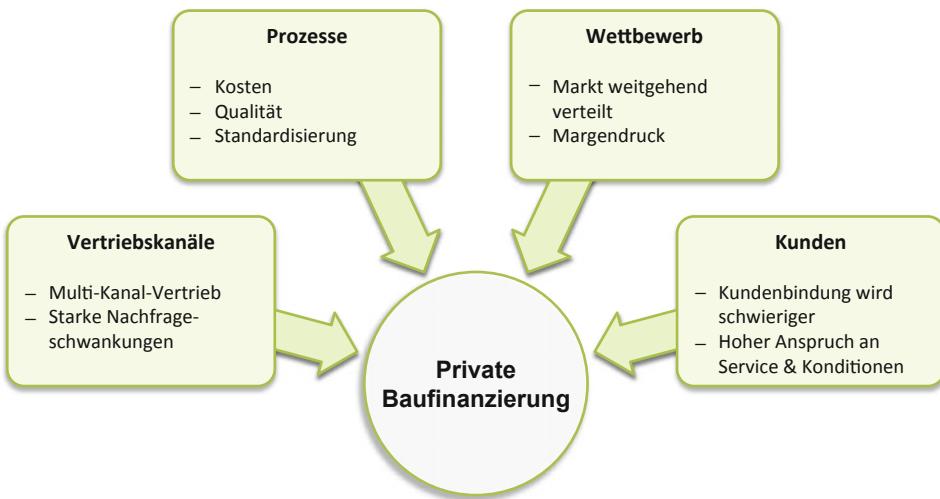
### **7.2.1 Aktuelle Herausforderungen im Markt für Baufinanzierung**

Für einen Großteil der Kreditinstitute in Deutschland zählt die private Baufinanzierung zum alltäglichen Geschäft. Insbesondere Bausparkassen haben sich auf dem Gebiet der Baufinanzierung stark positioniert und werden als zuverlässige und kompetente

---

<sup>7</sup> Vgl. Schmidt 1999, S. 60.

<sup>8</sup> Vgl. Schmidt 1999, S. 41.



**Abb. 7.1** Aktuelle Herausforderungen im Markt für private Baufinanzierung. (In Anlehnung an Müller 2011, S. 6)

Ansprechpartner mit hoher Kompetenz wahrgenommen. Weiterhin ermöglichen Baufinanzierungen vor allem regional verankerten Kreditinstituten den Zugang zum Kunden mit der Aussicht auf eine langfristige Bindung an das Institut. Jedoch sind eine kompetente Beratung sowie ein hoher Anteil an zufriedenen Kunden längst kein Indikator mehr für den Markterfolg eines Institutes. Etablierte Bankhäuser verlieren vor allem durch den Einstieg von Direktbanken und Vermittlungsplattformen in den Markt für Baufinanzierungen wertvolle Marktanteile. Zudem wirken klassische Beratungsfunktionen im Gegensatz zu immer innovativeren Online-Banking-Funktionalitäten oftmals zu langsam und wenig flexibel.<sup>9</sup>

Im Rahmen der Trendstudie „Bank & Zukunft“ hat das Fraunhofer Institut für Arbeitswirtschaft und Organisation (IAO) im Jahr 2007 eine Befragung über den zukünftigen strategischen Ausblick im Bankensektor für das Jahr 2015 durchgeführt, an der sich 460 Bankmanager beteiligten. Demnach erwarten rund 58,8 % der Befragten neue Konzepte im Vertriebsbereich, wie bspw. die Einführung einer Kreditbörsen oder den Direktvertrieb von Baufinanzierungen über das Internet. Dabei erhoffen sich die Befragten den Wegfall der klassischen Servicekraft und die Einführung von Online-Shops für den beratunglosen Vertrieb von Anlage- und Kreditprodukten.<sup>10</sup>

Bevor diese Zukunftsvisionen der Realität entsprechen können, stehen die Kreditinstitute zuerst vor aktuellen Herausforderungen im Markt für die private Baufinanzierung, welche in Abb. 7.1 dargestellt werden.

<sup>9</sup> Vgl. Müller 2011, S. 33.

<sup>10</sup> Vgl. Engstler et al. 2007, S. 2.

Die Positionierung auf dem dynamischen Wettbewerbsfeld, die Reaktion auf unterschiedliche Marktentwicklungen sowie das Erwirtschaften angemessener Erträge reflektieren nur einige Herausforderungen und Bedingungen am Immobilienmarkt, die von den Kreditinstituten bewältigt werden müssen. Zahlreiche Institute und Banken haben daher bereits Maßnahmen eingeleitet und zum Teil umgesetzt, um die Faktoren Effizienz, Qualität und Flexibilität in ihren Geschäftsprozessen entsprechend zu optimieren. Parallel hierzu liegen weitere Herausforderungen in der sinnvollen Gestaltung ansprechender Vertriebskanäle und Kundenbeziehungen, um angesichts der zunehmenden Diversifikation der Absatzwege eine adäquate Lösung zu finden.<sup>11</sup>

Banken und Kreditinstitute müssen erkennen, dass sich der Mehrwert des eigenen Angebotsportfolios in den Kundenprozessen widerspiegeln muss. Dies bedeutet, dass Kreditinstitute ihren Kunden Dienstleistungen für verschiedene Lebenssituationen anbieten, vielfältige Kontakt- und Interaktionsmöglichkeiten zur Verfügung stellen sowie individuell zugeschnittene Leistungen anbieten müssen. Letztlich müssen Banken und Institute über verschiedene Schnittstellen für den Kunden immer erreichbar sein, um einerseits frühzeitig Kunden zu akquirieren und andererseits über diese Schnittstelle Dienstleistungen und Produkte zu offerieren.<sup>12</sup> Der Einsatz von mobilen Applikationen bietet sich an, diese Schnittstellen zu repräsentieren und damit die Interaktion mit den Kunden zu verbessern. Weiterhin können mobile Applikationen dem Anspruch gerecht werden, zeit- und ortsunabhängig den Kunden ganzheitlich zu unterstützen.

Viele Banken, Versicherungen und Kreditinstitute haben das Potenzial mobiler Anwendungen bereits erkannt und arbeiten in Bezug auf ihr eigenes Angebotsspektrum an passenden mobilen Lösungen. Die Betrachtung des Marktes an Banking-Applikationen für mobile Endgeräte zeigt, dass innerhalb der letzten zwei Jahre ein enormer Zuwachs an verschiedenen Bank-, Kredit- oder Versicherungsanwendungen zu verzeichnen ist. So konnten im März 2010 weltweit insgesamt 83 Bankingapplikationen identifiziert werden, im Juli 2010 bereits 274 Applikationen.<sup>13</sup>

### **7.2.2 Analyse und Auswahl geeigneter Werkzeuge zur Applikationsentwicklung**

Für die planvolle und systematische Vorgehensweise bei der Softwareentwicklung muss, je nach vorliegender Aufgabenstellung, ein geeignetes Vorgehensmodell ausgewählt werden. In einem ersten Schritt werden daher ausgewählte Vorgehensmodelle analysiert und eine Vorgehensweise zur Entwicklung der mobilen Applikation ausgewählt. Daran anknüpfend erfolgt die Auswahl eines mobilen Betriebssystems sowie einer geeigneten Entwicklungs-umgebung zur Herstellung der Baufinanzierungs-Applikation. Hierfür werden Vor- und

---

<sup>11</sup> Vgl. Müller 2011, S. 33.

<sup>12</sup> Vgl. Moormann und Schaefer 2012, S. 52.

<sup>13</sup> Vgl. Moormann und Schaefer 2012, S. 47.

Nachteile der zuvor genannten mobilen Betriebssysteme und Entwicklungsumgebungen gegenübergestellt.

### 7.2.2.1 Auswahl eines Vorgehensmodells für das Projektmanagement

Die nachfolgenden Ausführungen befassen sich mit der Auswahl eines Vorgehensmodells zur Anwendungsentwicklung. Hierfür wurden in einem ersten Schritt klassische, moderne und agile Modelle voneinander abgegrenzt sowie deren grundlegende Eigenschaften und Ausprägungen betrachtet. Das nachfolgende Vorgehen zur Auswahl eines geeigneten Modells orientiert sich an einzelnen Modellkriterien. Entscheidende Faktoren bilden hierbei die jeweiligen Ziele, Konzepte sowie Projektzyklen der betrachteten Vorgehensmodelle. Zur Analyse sowie qualitativen Bewertung der jeweiligen Modelleigenschaften und -bedingungen erfolgte die Aufstellung eines Kriterienkatalogs. Anhand der im Katalog enthaltenen Basis- und Modellkriterien wurde die Beurteilung der Modelle unter Verwendung einzelner Bewertungsmatrizen vorgenommen. Die betrachteten Modellkriterien wurden jeweils durch eine Punktzahl bewertet. Die Vergabe von drei Punkten bedeutet, dass ein jeweiliges Kriterium gut ausgeprägt oder vorhanden ist. Die Angabe von zwei Punkten bedeutet, dass ein Kriterium mittelmäßig, bei einem Punkt nur mäßig ausgeprägt ist. Die Vergabe von keinem Punkt bedeutet, dass ein Kriterium nicht ausgeprägt oder nicht vorhanden ist.

Ein wichtiges Merkmal jedes konkreten Vorgehensmodells ist die Ausrichtung an eine gegebene Problemsituation. Erstes Kriterium bei der Wahl eines Vorgehensmodells ist somit die Anwendbarkeit für die gegebene Aufgabenstellung. Weiterhin müssen Vorgehensmodelle eine gewisse Flexibilität aufweisen, um außerplanmäßige Änderungen bei der Softwareentwicklung berücksichtigen zu können. Werden bereits zu Beginn der Softwareentwicklung wiederkehrende Aufgabenpakete registriert, bietet es sich an, ein Vorgehensmodell anzuwenden, welches die iterative Abwicklung gleichartiger Aktivitäten ermöglicht. Ein Vorgehensmodell muss weiterhin leicht verständlich und einfach strukturiert sein.<sup>14</sup> Die betrachteten Vorgehensmodelle lassen sich hinsichtlich ihrer Grundprinzipien charakterisieren (vgl. Abb. 7.2).

Unabhängig von den jeweiligen Grundprinzipien und der vorliegenden Problemstellung müssen Vorgehensmodelle die einzelnen Phasen des Softwareentwicklungsprozesses unterstützen, d. h. Anforderungsermittlung, Analyse, Entwurf, Implementierung, Test und Qualitätssicherung. Des Weiteren müssen die Modelle Aktivitäten zum Projektmanagement sowie zur Dokumentation des Projektverlaufes aufweisen.<sup>15</sup> Die Ausprägungen zur Phasenunterstützung der genannten Vorgehensmodelle werden in Abb. 7.3 aufgezeigt.

Neben der Unterstützung der Phasen des Softwareentwicklungsprozesses erhoffen sich Softwarehersteller durch den Einsatz von Vorgehensmodellen, nützliche Werkzeuge bereitgestellt zu bekommen. Hierbei ist der Einsatz verschiedener Werkzeuge denkbar, bspw. für die Modellierung von Anwendungsfällen, die Erstellung lauffähiger Prototypen oder

<sup>14</sup> Vgl. Bunse und von Knethen 2008, S. 101.

<sup>15</sup> Vgl. Bunse und von Knethen 2008, S. 101.

Grundprinzipien	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Anwendbarkeit, Praxisnähe	3	2	2	3	3	3
Flexibilität des Modells	1	2	1	3	2	2
Iterationen vorgesehen	0	3	3	3	3	0
Verständlichkeit, Einfachheit	3	2	3	2	3	3
<i>Summe</i>	7	9	9	11	11	8

**Abb. 7.2** Bewertung der Vorgehensmodelle hinsichtlich ihrer Grundprinzipien

Phasenunterstützung	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Anforderungsermittlung	3	3	3	3	3	3
Analyse	3	3	3	3	3	3
Entwurf	3	3	3	3	3	3
Implementierung	3	3	3	3	3	3
Test	3	3	3	3	3	3
Qualitätssicherung	3	0	0	0	0	3
<i>Summe</i>	18	15	15	15	15	18

**Abb. 7.3** Bewertung der Vorgehensmodelle hinsichtlich ihrer Phasenunterstützung

zur Evaluierung des Projekterfolges.<sup>16</sup> Nachfolgende Abbildung stellt die Möglichkeiten zur Werkzeugunterstützung durch die genannten Vorgehensmodelle dar (vgl. Abb. 7.4).

Ein weiterer wichtiger Punkt bei der Auswahl von Vorgehensmodellen ist deren Größe bzw. Komplexität. Die Komplexität eines Vorgehensmodells kann sich hierbei durch verschiedene Aspekte ausdrücken, bspw. die Anzahl der durchzuführenden Phasen, den Anteil an benötigten Mitarbeitern zur Projektrealisierung sowie den Umfang des Softwareprojektes. Insbesondere in Bezug auf die Projektgröße und die verfügbare Anzahl an Mitarbeitern bildet die Komplexität ein entscheidendes Auswahlkriterium. Oftmals können kleinere Projektteams die hohen Anforderungen komplexer Vorgehensmodelle nicht meistern, was letztendlich zum Scheitern des gesamten Softwareprojektes führen kann.<sup>17</sup> In Bezug auf die Anwendbarkeit auf Softwareprojekte ergibt sich für die genannten Vorgehensmodelle nachfolgendes Bild (Abb. 7.5).

Obwohl Softwareentwickler gegenwärtig auf eine Vielzahl an unterschiedlichen Vorgehensmodellen zur Softwareentwicklung zurückgreifen können, zeigte die Analyse, dass bisher noch kein einheitliches und standardisiertes Vorgehensmodell für Softwareprojekte vorhanden ist.<sup>18</sup> Ziel der Analyse bestand somit weniger in der Identifikation eines allgemeingültigen und universalen Vorgehensmodells sondern vielmehr in der Unter-

<sup>16</sup> Vgl. Bunse und von Knethen 2008, S. 121.

<sup>17</sup> Vgl. Bunse und von Knethen 2008, S.119.

<sup>18</sup> Vgl. Bunse und von Knethen 2008, S. 1 f.

Werkzeuge	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Erstellung von Modellen	0	0	0	3	0	0
Entwicklung von Prototypen	0	3	0	0	3	3
Evaluierung und Validierung	3	3	3	3	0	3
<b>Summe:</b>	<b>3</b>	<b>6</b>	<b>3</b>	<b>6</b>	<b>3</b>	<b>6</b>

**Abb. 7.4** Bewertung der Vorgehensmodelle hinsichtlich ihrer Werkzeugunterstützung

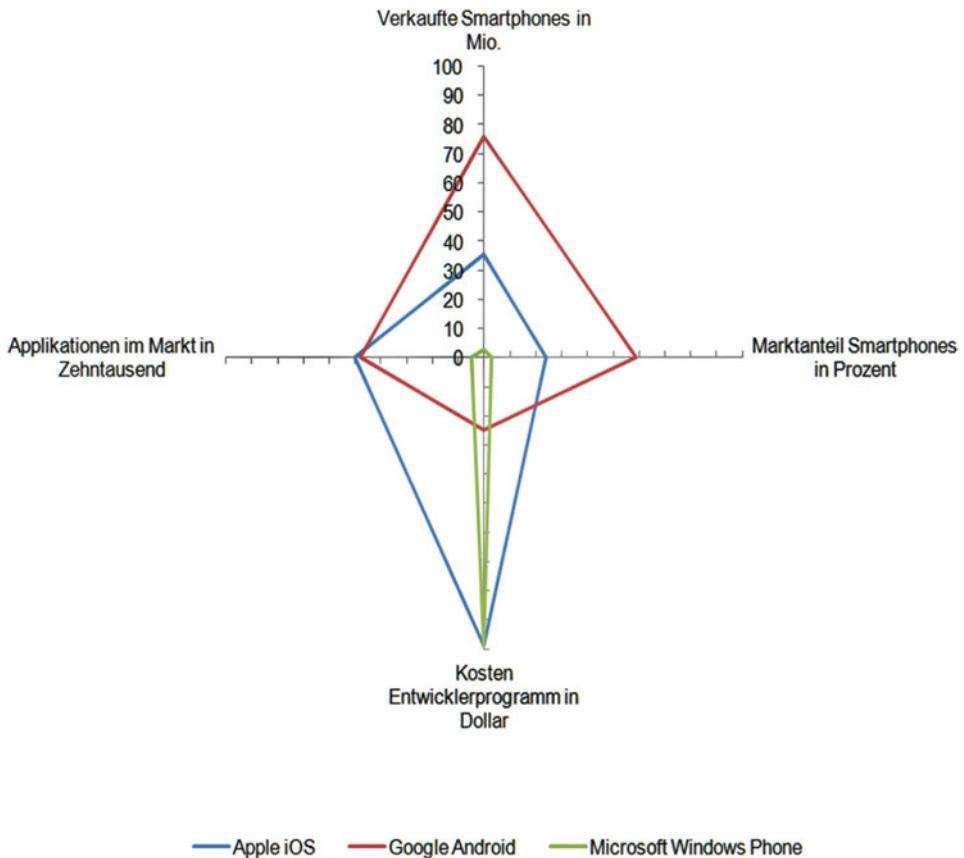
Komplexität	Wasserfall-Modell	Spiral-Modell	V-Modell	RUP-Modell	Extreme Programming	UCAN-Modell
Kompakt, für kleinere Projekte geeignet	3	1	2	1	3	3
Mittel	2	2	2	1	2	3
Umfangreich, für große Projekte geeignet	1	3	3	3	1	2
<b>Summe:</b>	<b>6</b>	<b>6</b>	<b>7</b>	<b>5</b>	<b>6</b>	<b>8</b>

**Abb. 7.5** Bewertung der Vorgehensmodelle hinsichtlich ihrer Komplexität

suchung und Auswahl eines etablierten klassischen, modernen oder agilen Modells, zur bestmöglichen Unterstützung bei der Entwicklung einer mobilen Applikation zur Baufinanzierung.

Das RUP-Modell sowie das Extreme Programming stellten bestmögliche Alternativen bzgl. der Betrachtung der Grundprinzipien dar. Beide Modelle weisen eine hohe Praxisnähe sowie Flexibilität auf. Die bestmögliche Phasenunterstützung bieten das Wasserfall-Modell sowie das UCAN-Modell. Im Gegensatz zu den anderen Vorgehensmodellen betrachten beide Modelle Aktivitäten und Aufgaben der Qualitätssicherung. Des Weiteren bieten das Spiral-, RUP- und UCAN-Modell eine optimale Unterstützung des Entwicklungsprozesses durch nützliche Werkzeuge zur Modellierung, zum Prototyping sowie zur Evaluierung. Die Analyse in Bezug auf die Komplexität und Anwendbarkeit auf unterschiedlich ausgeprägte Softwareprojekte von Vorgehensmodellen ergab weiterhin, dass insbesondere das UCAN-Modell die beste Alternative für kleine, mittlere und große Softwareprojekte darstellt. Vor diesem Hintergrund wird im Folgenden die Planung und Entwicklung der mobilen Applikation zur Baufinanzierung mithilfe des UCAN-Vorgehensmodells durchgeführt.

Die Auswahl des Vorgehensmodells bildet nun die Grundlage für den Entwurf des Projektplans, für die Auswahl eines mobilen Betriebssystems sowie für die Auswahl einer



**Abb. 7.6** Netzdiagramm zur Analyse mobiler Betriebssysteme

entsprechenden Entwicklungsumgebung. Nachfolgend wird zunächst das Verfahren zur Analyse und Auswahl eines geeigneten Betriebssystems zur mobilen Applikationsentwicklung beschrieben und die Ergebnisse der Analyse vorgestellt.

### 7.2.2.2 Auswahl eines mobilen Betriebssystems zur Applikationsentwicklung

Die nachfolgend durchgeföhrte Analyse mobiler Betriebssysteme bezieht sich in erster Linie auf Verkaufszahlen, Marktanteile, Anzahl vorhandener Applikationen im Markt sowie anfallende Registrierungsgebühren bei der Verbreitung einer mobilen Anwendung. Die Angaben zu Marktanteilen, Applikationen im Markt sowie den Kosten für das Entwicklerprogramm beziehen sich hierbei jeweils auf das erste Quartal im Jahr 2012. Die Angaben zu den Verkaufszahlen beziehen sich auf das letzte Quartal im Jahr 2011. Weiterhin erfolgt bei den Angaben jeweils der Bezug auf Smartphones, die mit einem der betrachteten Betriebssystemen ausgeliefert wurden. Die Angaben basieren zudem auf weltweiten Erhebungen. Das Ergebnis der Analyse wird in Abb. 7.6 dargestellt.

	Apple iOS	Google Android	Microsoft Windows Phone
Hardwarestandards	ja	nein	ja
Programmiersprachen	Objective C	Java, C	Visual Basic, C#
Benötigtes stationäres Betriebssystem	Apple OSX	Windows, OSX, Linux	Windows
Entwicklungsumgebung	XCode	Eclipse, App Inventor	Visual Studio
Multitasking	ja	ja	nein
Eigene Distribution möglich	nein	ja	nein

**Abb. 7.7** Gegenüberstellung technologischer Ausprägungen mobiler Betriebssysteme

Im vierten Quartal des Jahres 2011 konnten insgesamt 75,9 Mio. verkaufte Smartphones mit einem Android-Betriebssystem registriert werden. Damit ist Google vor Apple mit 35,4 Mio. und Microsoft mit 2,7 Mio. führend.<sup>19</sup> Ebenso ist Google mit einem weltweiten Marktanteil von 59 % vor Apple mit 23,8 % und Microsoft mit insgesamt 3 %.<sup>20</sup> In Bezug auf das Angebot an mobilen Anwendungen hat Apple einen kleinen Vorsprung aufzuweisen. Im ersten Quartal des Jahres 2012 konnten über 500.000 Applikationen im Apple-Markt erfasst werden, dahinter folgt der Google-Play-Markt mit 480.000 und Microsofts Marketplace mit 50.000 registrierten Applikationen.<sup>21</sup> Die Registrierungsgebühren bei Apple und Microsoft sind identisch und belaufen sich auf 99 US-\$ pro Jahr. Im Gegensatz hierzu fordert Google für die Registrierung einmalig 25 US-\$.<sup>22</sup>

Für die Auswahl eines mobilen Betriebssystems müssen neben der zuvor beschriebenen Kennzahlenanalyse weiterhin technologische Ausprägungen von Betriebssystemen betrachtet werden. Hierzu zählen bspw. Programmiersprachen und -umgebungen sowie Entwicklerwerkzeuge, welche durch das Betriebssystem unterstützt werden. In Abb. 7.7 werden technische Anforderungen an mobile Betriebssysteme gegenübergestellt. Die aufgeführten Kriterien erheben jedoch nicht den Anspruch, das gesamte Spektrum relevanter Eigenschaften mobiler Betriebssysteme abzudecken.

Hellgrün schattierte Flächen weisen auf besonders gute Ausprägungen in Bezug auf eine möglichst kostengünstige Entwicklung von mobilen Applikationen hin, bspw. bei den Ausprägungen „Benötigtes stationäres Betriebssystem“ und „Entwicklungsumgebung“. Da unter monetären Gesichtspunkten kein Vergleich in Bezug auf einsetzbare Programmiersprachen erfolgen kann, wurden diese von der Bewertung ausgeschlossen, jedoch aus Gründen der Vollständigkeit angegeben.

<sup>19</sup> Vgl. Gartner Inc. 2012.

<sup>20</sup> Vgl. Parbel 2012.

<sup>21</sup> Vgl. Microsoft Corp. 2012a; Apple Inc. 2012a.

<sup>22</sup> Vgl. Google Android Developer Portal 2012a; Apple Inc. 2012b; Microsoft Corp. 2012b.

Neben monetären Gesichtspunkten können ein besonderer Mehrwert und Alleinstellungsmerkmale erkannt werden. So ist die eigenständige Distribution der entwickelten Applikation nur bei Google möglich. Da Android selbst der Open-Source-Lizenz unterliegt, kann jeder Entwickler eigenständige Android-Versionen entwickeln und vermarkten. Daher können im Gegensatz zu Apple und Microsoft keine Hardwarestandards für Android gesetzt werden. Microsofts Betriebssystem verfügt des Weiteren derzeit nicht über die Funktion des Multitasking. Dies bedeutet, dass der Wechsel zwischen zwei oder mehreren laufenden Anwendungen nicht gegeben ist. Die Vorteile in Bezug auf die Unterstützung von Multitasking-Anwendungen liegen somit eindeutig bei Apple und Google.

Die Auswertung der Ergebnisse hat gezeigt, dass Android ein modernes und umfangreiches mobiles Betriebssystem darstellt, welches die kostengünstige Entwicklung mobiler Anwendungen unterstützt. Android stellt zudem ein weitverbreitetes Betriebssystem mit hohem weltweiten Marktanteil dar (vgl. Abb. 7.6). Auf Grundlage der Analyseergebnisse wird für die Entwicklung einer mobilen Applikation zur Baufinanzierung das mobile Betriebssystem Android ausgewählt. Im nachfolgenden Schritt wird eine hierfür geeignete Entwicklungsumgebung ausgewählt.

### **7.2.2.3 Auswahl einer Entwicklungsumgebung für Android-basierte Applikationen**

Vor dem Hintergrund der im vorhergehenden Abschnitt durchgeföhrten Analyse können nachfolgend die Programmiersprachen Objective-C, C# sowie HTML 5 für weitere Betrachtungen ausgeschlossen werden, da diese hauptsächlich zur Programmierung von Applikationen für die Betriebssysteme von Apple und Microsoft verwendet werden. Des Weiteren werden Entwicklungsumgebungen, die auf Apple- und Windows-Betriebssysteme ausgerichtet sind, nicht weiter betrachtet. Schwerpunkt und Ziel der Analyse bildet somit die Auswahl einer Entwicklungsumgebung für Android-basierte Applikationen. Im Fokus der Analyse stehen die Entwicklungsumgebungen Eclipse sowie der von Google gehostete App Inventor. Beide Umgebungen unterstützen die Entwicklung Java-basierter Applikationen. Die Analyse der Entwicklungsumgebung beinhaltete neben der Betrachtung technischer Ausprägungen, wie z. B. vorhandener Programmierbibliotheken oder ausreichender Testmöglichkeiten, wirtschaftliche Faktoren in Bezug auf die Distribution der entwickelten Applikation sowie daraus entstehender Entwicklungskosten. Die Ergebnisse der Analyse werden in Abb. 7.8 zusammengefasst.

Die in Abb. 7.8 dargestellten Ergebnisse der Analyse zeigen, dass beide Entwicklungsumgebungen fast annähernd gleich sind und einen ähnlichen Funktionsumfang aufzuzeigen. Beide Umgebungen weisen identische Ausprägungen in Bezug auf vorhandene Online-Hilfen und Java-Bibliotheken sowie auf die Nutzung interner Hardwareressourcen auf. Hinsichtlich der Distribution entwickelter Anwendungen sowie der Portierung dieser auf andere Betriebssysteme sind bei der Entwicklungsumgebung Eclipse deutliche Vorteile gegenüber dem App Inventor nachzuweisen. App Inventor erfordert jedoch einen geringen Programmieraufwand und bietet die Möglichkeit des Vertriebs der Applikation über eigene Distributionswege. Da der App Inventor über einen beliebigen Internetbrowser

	Eclipse Foundation Eclipse	Google App Inventor
Programmiererkenntnisse notwendig	ja	nein
Installation der Umgebung notwendig	ja; Windows, OSX, Linux	nein
Java-Bibliotheken vorhanden	ja	ja
Zugriff auf Hardwareressourcen möglich	ja	ja
Programmieraufwand	hoch	gering
Testen der Applikation	ja; Android Emulator	ja; Android Emulator sowie in Echtzeit über angeschlossenes Endgerät
Distribution über Google Play-Store	ja	nein; eigene Distribution möglich
Entwicklerkosten	ja; fallen an bei der Distribution der Applikation	nein
Portierung auf andere Betriebssysteme möglich	ja	nein
Tutorials / Online-Hilfen vorhanden	ja	ja

**Abb. 7.8** Analyse und Bewertung Android-basierter Entwicklungsumgebung

geöffnet werden kann, wird für die Nutzung der Entwicklungsumgebung keine Installation vorausgesetzt. Ebenfalls fallen bei der Herstellung durch den App Inventor keine Entwicklungskosten an. Des Weiteren werden keine Programmiererkenntnisse vorausgesetzt. Aufgrund dieser Vorteile gegenüber Eclipse wird nachfolgend für die Entwicklung der mobilen Applikation zur Baufinanzierung die Entwicklungsumgebung App Inventor verwendet.

Die Gegenüberstellung mobiler Endgeräte wird nachfolgend nicht weiter betrachtet, da zu Beginn der Implementierungsphase ein geeignetes Android-basierendes Endgerät zu Entwicklungs- und Testzwecken zur Verfügung gestellt wurde. Im nächsten Abschnitt wird die Terminierung und Entwicklung des Projektplans anhand der im UCAN-Modell vorgesehenen Projektphasen durchgeführt. Weiterhin wird Aufschluss über die zeitliche Abschätzung des Gesamtprojektes sowie wichtige Meilensteine innerhalb der Entwicklungsphase gegeben.

### 7.2.3 Zerlegung und zeitliche Abschätzung des Projektes

Die zeitliche Abgrenzung eines Gesamtprojektes ermöglicht die Terminierung eines spezifischen Anfangs- und Endergebnisses sowie die Festlegung angestrebter Zwischener-

Nr.	Vorgangsname	Dauer	Anfang	Ende
1	Initiale Idee	0 Tage	Mo 09.04.2012	Mo 09.04.2012
2	Entwurf, Design und Layout	7 Tage	Mo 09.04.2012	So 15.04.2012
3	Erstellung Papier-Prototyp	7 Tage	Mo 09.04.2012	So 15.04.2012
4	Evaluierung der initialen Idee	0 Tage	So 15.04.2012	So 15.04.2012
5	Erstellung Klick-Dummy	7 Tage	Mo 16.04.2012	So 22.04.2012
6	Evaluierung Klick-Dummy	0 Tage	So 22.04.2012	So 22.04.2012
7	Erstellung funktionsfähiger Prototyp	14 Tage	Mo 23.04.2012	So 06.04.2012
8	Evaluierung funktionsfähiger Prototyp	0 Tage	So 06.04.2012	So 06.04.2012
9	Erstellung finale Applikation	7 Tage	Mo 07.05.2012	So 13.05.2012
10	Evaluierung finale Applikation	0 Tage	So 13.05.2012	So 13.05.2012
11	Testphase	5 Tage	Mo 14.05.2012	Fr 18.05.2012
12	Fehlerkorrektur	5 Tage	Mo 14.05.2012	Fr 18.05.2012
13	Projektende	0 Tage	Fr 18.05.2012	Fr 18.05.2012

**Abb. 7.9** Projektplan zur Applikationsentwicklung

gebnisse in Form von Meilensteinen. Durch die Aufgliederung des Softwareentwicklungsprozesses in einzelne Arbeitspakete kann weiterhin eine detaillierte Projektplanung sowie die Gestaltung der Projektorganisation realisiert werden. Des Weiteren werden durch die Erstellung des Projektplanes einsetzbare Ressourcen, wie bspw. Personal, Sachmittel oder Finanzmittel, den zeitlich abgegrenzten Aufgaben und Aktivitäten zugewiesen.<sup>23</sup> Die im Projektplan enthaltenen Projektmethoden und -werkzeuge richten sich in erster Linie an den Vorgaben des UCAN-Modells aus. Im Zuge der Projektvorbereitung müssen weiterhin folgende Besonderheiten berücksichtigt werden:

- Die Entwicklung der mobilen Applikation zur Baufinanzierung sowie deren Dokumentation war für den Zeitraum von März 2012 bis August 2012 vorgesehen.
- Eine ausführliche Personalplanung wurde nicht berücksichtigt, da die Planung, Umsetzung und Evaluierung der Applikation durch eine Person allein durchgeführt wurde.
- Ebenfalls wurde die Planung von Sach- und Finanzmitteln nicht berücksichtigt, da die zur Implementierung der Applikation benötigten Geräte und Unterlagen im Voraus zur Verfügung standen.

Auf Basis der Vorgaben des UCAN-Modells sowie der Projektvorbereitung erfolgte die Erstellung eines Projektplans, dessen Bestandteile in Abb. 7.9 aufgelistet sind. Ergebnis des Projektplans bildete die Bestimmung eines Zeitraums für die Entwicklung der mobilen

<sup>23</sup> Vgl. Aichele 2006, S. 23 f.

Applikation. Vorgänge zur Dokumentation der Applikation, bspw. die Durchführung einer Zwischen- und Abschlusspräsentation, wurden bei der Planung nicht berücksichtigt.

Der Projektplan betrachtet die Beschreibung und Dauer der durchzuführenden Vorgänge sowie den Zeitraum für die Durchführung des Vorgangs. Vorgänge mit einer Dauer von null Tagen kennzeichnen Meilensteine des Projektes und somit zu erreichende Zwischenergebnisse innerhalb des Entwicklungsprozesses. Der Projektplan umfasst 13 Vorgänge und definiert den Start des Entwicklungsprozesses auf den 9. April 2012 sowie das Ende auf den 18. Mai 2012. Insgesamt wurden für die Durchführung des Entwicklungsprozesses 52 Tage kalkuliert. Die einzelnen Vorgänge werden nachfolgend kurz erläutert.

Auslöser des Projektes bildet die initiale Idee zur Entwicklung einer Applikation. Die daran anknüpfende Entwurfsphase wurde mit insgesamt sieben Tagen kalkuliert. Zu den Aktivitäten dieser Phase gehört die grafische Darstellung der Benutzeroberflächen, die Auswahl ansprechender Farben, die Strukturierung der Bedienelemente sowie die Analyse benötigter finanzmathematischer Rechenvorschriften zur Berechnung einer Baufinanzierung. Parallel zum Entwurf der Applikation erfolgt die Anfertigung eines Papierprototyps zur Validierung der Entwurfsergebnisse. Die Anfertigung eines ersten Klickdummym startet nach erfolgreicher Evaluierung der initialen Idee. Dieser Prototyp soll die in der Entwurfsphase festgelegten Strukturen, Steuerelemente und Farben in ein erstes Versuchsmodell überführen. Im Anschluss an die Evaluation des Versuchsmodells erfolgt durch die Implementierung der finanzmathematischen Funktion die Erstellung eines lauffähigen Prototyps. Die daran anknüpfende Evaluierung soll zur ersten Fehleranalyse dienen und Schwachstellen des Prototyps aufzeigen. Die Eliminierung dieser Schwachstellen führt zur finalen Version der Applikation, welche in einem nachfolgenden Schritt nochmals auf Fehler geprüft wird. Innerhalb der Testphase wird die mobile Applikation an externe Testpersonen weitergegeben. Ein speziell auf die Applikation ausgerichteter Evaluationsbogen soll die Auffassungen und Anmerkungen der Testgruppe zur mobilen Anwendung erfassen. Für den Fall der Identifikation weiterer Schwachstellen durch die Testpersonen dient eine abschließende Fehlerkorrekturphase zur Verbesserung und Optimierung der Applikation.

Die erzielten Ergebnisse der Planungs- und Analysephase werden nachfolgend nochmals zusammengefasst.

## 7.2.4 Zusammenfassung

Die erfolgreiche und ökonomische Entwicklung von Anwendungssoftware setzt eine qualitative und quantitative Planung und Strukturierung des Softwareentwicklungsprozesses voraus. Dabei reicht es nicht aus, die durch das Projektmanagement zur Verfügung stehenden Methoden, Techniken und Werkzeuge anzuwenden. Wesentlich bedeutender sind die Analyse der vorliegenden Problemsituation sowie die Planung eines konkreten Vorgehens zur Lösung der Aufgabenstellung. Hierzu müssen zu bearbeitende Aufgaben und Tätigkeiten identifiziert und in eine zeitliche Reihenfolge gebracht werden. Vor diesem

Hintergrund wurden im vorliegenden Kapitel mehrere Planungs- und Analyseprozesse beschrieben, die zur Bestimmung der gegebenen Rahmenbedingungen erforderlich waren.

Ziel der ersten Phase des Analyseprozesses bildete die Untersuchung des Marktes für Baufinanzierungen. In diesem Zusammenhang konnten aktuelle Herausforderungen für Kreditinstitute lokalisiert werden, welche sich in den Begriffen Vertriebskanal, Prozess, Wettbewerb und Kunde widerspiegeln. Banken und Institute müssen möglichst qualitative und kostengünstige Finanzierungsprozesse aufweisen, um einerseits die Kundenbindung zu fördern sowie dem hohen Anspruch der Kunden gerecht zu werden und andererseits im Wettbewerb zu bestehen sowie dem wachsenden Margendruck standzuhalten. Ein weiterer wichtiger Schritt vor Beginn der Entwicklertätigkeit ist die Analyse potenzieller Konkurrenzprodukte. Hierzu wurden am Markt vorhandene Baufinanzierungs-Applikation betrachtet, deren Funktionsumfang untersucht und aufgetretene Fehler oder Schwachstellen dokumentiert. Das Ergebnis der Analyse zeigte, dass gegenwärtig keine optimalen sowie kostenlosen Applikationen zur Ermittlung einer Baufinanzierung angeboten werden. Anknüpfend an die Konkurrenzanalyse erfolgte die Auswahl eines Vorgehensmodells zur Applikationsentwicklung. Die Bewertung der betrachteten Vorgehensmodelle erfolgte anhand eines Kriterienkatalogs, der sich aus den jeweiligen Funktionen und Zielen der Modelle ableitete. Das UCAN-Modell wurde im Zuge der Beurteilung des Auswahlverfahrens als bestmögliche Alternative für das weitere Projektvorgehen ausgewählt. Weitere Analyse- und Auswahlverfahren in Bezug auf die Wahl eines mobilen Betriebssystems sowie einer geeigneten Entwicklungsumgebung wurden analog unter Berücksichtigung eines Punktesystems durchgeführt. Hierbei wurden das Betriebssystem Android sowie die Entwicklungsumgebung App Inventor zur Erstellung der mobilen Applikation ausgewählt. Abschließend erfolgte die Vorstellung des Projektplans sowie die Darstellung und Beschreibung der einzelnen Arbeitspakete.

Im nachfolgenden Kapitel wird ausführlich auf die Erhebung notwendiger Anforderungen an die Applikation eingegangen.

---

### 7.3 Entwurfs- und Implementierungsphase

Das Ziel der Entwurfsphase besteht in der Konstruktion, Analyse und Dokumentation technisch adäquater Lösungen in Bezug auf die vorliegenden Planungs- und Analyseergebnisse der vorherigen Phase. Hierfür werden zu Beginn der Entwurfsphase die Ziele des Entwicklungsprozesses bestimmt, deren Inhalte und Ausrichtungen maßgeblich den Projekterfolg bestimmen. Für die Beschreibung von Sollzuständen zukünftiger Softwaresysteme müssen auf Grundlage der erfassten Ziele Anforderungen an die Software erhoben werden. Die Anforderungsphase beginnt mit der Analyse potenzieller Anforderungsquellen und endet mit der Fertigstellung eines vollständigen und konsistenten Anforderungsdokuments sowie der Modellierung des Softwaresystems zum

**Abb. 7.10** Aktivitäten der Entwurfs- und Implementierungsphase

Entwurf	Implementierung
<ul style="list-style-type: none"> <li>- Zielbestimmung</li> <li>- Anforderungs-ermittlung</li> <li>- Anforderungs-spezifikation</li> <li>- Darstellung des Programmablaufs</li> </ul>	<ul style="list-style-type: none"> <li>- Umsetzung der Anforderungen</li> <li>- Entwicklung der Datenstrukturen und Algorithmen</li> <li>- Dokumentation der Problemlösung</li> </ul>

besseren Verständnis des Programmablaufes.<sup>24</sup> Aufbauend auf dem Entwurf der Softwarearchitektur erfolgt im Verlauf der Implementierungsphase die Transformierung der Modelle und Anforderungen zu lauffähigen Softwareprogrammen. Die Umsetzung des Softwarekonzeptes, die Entwicklung von Datenstrukturen und Algorithmen sowie die Dokumentation der Problemlösung bilden typische Aktivitäten während der Realisierung von Softwaresystemen.

Das vorliegende Kapitel befasst sich umfassend mit den in Abb. 7.10 dargestellten Aktivitäten für den Entwurf und die Realisierung der mobilen Applikation zur Immobilienfinanzierung. In Bezug auf die Anforderungserhebung werden in einem ersten Schritt allgemeine und projektspezifische Ziele zur Entwicklung der mobilen Applikation ermittelt. Hinsichtlich der festgelegten Ziele werden anschließend Anforderungen an die mobile Anwendung eruiert. Die erhobenen Kriterien an die Software werden nachfolgend durch die Visualisierung des Programmablaufs in einen kausalen Zusammenhang gebracht. Die Visualisierung des allgemeinen Programmablaufs dient weiterhin der anknüpfenden Umsetzung der Anforderungen. In diesem Zusammenhang wird die Gestaltung der Benutzeroberflächen aufgezeigt sowie die Entwicklung der notwendigen Softwarefunktionen und Datenstrukturen erläutert. Das Kapitel endet mit der Beschreibung, Auflistung und Lösung aufgetretener Probleme während des Entwicklungsprozesses.

### 7.3.1 Anforderungserhebung

Das Erheben und Verwalten von Anforderungen an eine Software bildet den zentralen Schwerpunkt des Anforderungsmanagements. Anforderungen beschreiben Funktionalitäten und Eigenschaften einer Software, die entweder aus Kundenwünschen oder gegebenen Aufgabenstellungen abgeleitet werden. Aufgabe der Software- bzw. Systementwickler ist, die festgelegten Anforderungen zu verstehen und entsprechend umzusetzen, damit

<sup>24</sup> Vgl. Krcmar 2010, S. 166 f.



**Abb. 7.11** Unterschiedliche Anforderungstypen. (In Anlehnung an Krcmar 2010, S. 165)

die Kunden des Produktes zufriedengestellt oder Problemlösungen umgesetzt werden können.<sup>25</sup> Pohl definiert den Begriff Anforderung wie folgt:<sup>26</sup>

1. Eine Bedingung oder Eigenschaft, die ein System oder eine Person benötigt, um ein Problem zu lösen oder ein Ziel zu erreichen,
2. Eine Bedingung oder Eigenschaft, die ein System oder eine Systemkomponente aufweisen muss, um einen Vertrag zu erfüllen oder einem Standard, einer Spezifikation oder einem anderen formell auferlegten Dokument zu genügen,
3. Eine dokumentierte Repräsentation einer Bedingung oder Eigenschaft, wie in (1) oder (2) definiert.

In der Softwareentwicklung werden unterschiedliche Typen von Anforderungen differenziert. Anforderungen an ein Softwaresystem werden in funktionale und nichtfunktionale Anforderungen unterschieden. Funktionale Anforderungen beschreiben das Verhalten sowie die Funktionen der Software. Ergebnis der Erhebung dieser Anforderungen bildet die Abgrenzung des Leistungsumfangs einer Software. Im Gegensatz hierzu beschreiben nichtfunktionale Anforderungen Möglichkeiten zur Realisierung funktionaler Anforderungen und geben hierfür feste Rahmenbedingungen vor, bspw. Benutzbarkeit, Zuverlässigkeit oder Wartbarkeit. Eine Unterteilung in Entwickler- und Anwendersicht dient der besseren Strukturierung funktionaler und nichtfunktionaler Anforderungen.<sup>27</sup> Abbildung 7.11 stellt die unterschiedlichen Anforderungstypen grafisch dar.

<sup>25</sup> Vgl. Krcmar 2010, S. 164 f.

<sup>26</sup> Vgl. Pohl 2008, S. 13.

<sup>27</sup> Vgl. Krcmar 2010, S. 164 f.

Nachfolgend werden im Rahmen der Anforderungsermittlung und -analyse notwendige funktionale und nichtfunktionale Anforderungen an die mobile Baufinanzierungs-Applikation genannt. Hierzu werden in einem vorhergehenden Schritt die zu erreichenden Ziele der Applikation bestimmt. Aufbauend auf der Zielbestimmung und der Anforderungserhebung werden die Funktionen und Daten der mobilen Applikation genannt und erläutert. Abschließend erfolgt die Darstellung des Programmablaufs anhand der BPMN-Modellierungsmethodik.

### 7.3.1.1 Zielsetzung und Zielbestimmung

Die Einhaltung konkreter Ziele ist eine fundamentale Voraussetzung für die Entwicklung erfolgreicher Softwareprojekte. Ziele müssen insbesondere unter der Berücksichtigung von Kosten, Qualität und Terminen definiert und ausreichend charakterisiert werden. Die Zielbestimmung unterscheidet hierbei globale und projektspezifische Ziele. Letztere lassen sich anhand der Phasen des jeweils eingesetzten Vorgehensmodells zur Softwareentwicklung ableiten. Projektbezogene Ziele beziehen sich auf das Herleiten inhaltlicher Vorgaben aus der Aufgabenstellung und definieren somit das zu entwickelnde Softwareprodukt.<sup>28</sup> Nachfolgend werden die Ziele der mobilen Anwendung zur Baufinanzierung dargestellt sowie Anforderungen an die Applikation aufgezeigt.

Die mobile Applikation verfolgt das Hauptziel, allgemeine Fragen zum Thema Baufinanzierung zu beantworten. Hierzu zählen z. B.:

- Reicht mein Eigenkapital für eine Finanzierung aus?
- Wie viel Geld benötige ich für die Finanzierung eines Immobilienobjekts?
- Wie viel darf das Immobilienobjekt insgesamt kosten?
- Welche Mehrbelastung habe ich bei der Finanzierung eines Hauses im Vergleich zur aktuellen Kaltmiete?
- Welche Nebenkosten fallen bei der Finanzierung des Immobilienobjekts an?

Die mobile Applikation zur Baufinanzierung soll eine Schnittstelle zwischen Kunden, Kreditinstituten sowie Immobilienmaklern darstellen. Die Zielgruppe setzt sich aus mobilen Endnutzern zusammen, welche die Finanzierung einer Immobilie vorab simulieren oder zwischen unterschiedlichen Finanzierungsmöglichkeiten zur Immobilienanschaffung wählen möchten. Weiterhin soll die Applikation die Möglichkeit bieten, unter Berücksichtigung der aktuellen privaten Finanzsituation, Aussagen über die Finanzierung einer oder mehrerer Immobilien zu tätigen. Hierbei sollen durch die Applikation ungefähre maximale Objektkosten ermittelt werden.

Zur Berechnung dieser Objektkosten müssen Daten über das Immobilienobjekt und dessen Nebenkosten, das Eigenkapital, das Einkommen und die Ausgaben sowie Daten über evtl. vorhandene Finanzierungsmöglichkeiten in die dazu vorgesehenen Eingabefelder eingetragen werden. Auf dieser Informationsbasis können Zwischenergebnisse

<sup>28</sup> Vgl. Gernert 2003, S. 57.

Nr.	Musskriterien an die Baufinanzierungs-Applikation
1	Bereitstellung von Funktionen zur Ermittlung einer Baufinanzierung
2	Ausgabe korrekter und wahrheitsgetreuer Ergebnisse
3	Möglichkeit der Eingabe benutzerindividueller Daten
4	Gewährleistung der Datensicherheit
5	Komplette Eingabefehler-Prüfung
6	Anwendbarkeit auf verschiedenen ausgeprägten Android-Endgeräten

**Abb. 7.12** Musskriterien an die Baufinanzierungs-Applikation

ermittelt und an den Benutzer ausgegeben werden. Die Ergebnisse sollen des Weiteren zu einem späteren Zeitpunkt zur Verfügung stehen. Aufgrund der Funktionalität sowie des gegebenen Aufgabengebietes erhält die mobile Anwendung die Bezeichnung „BauFinanz“. Im Folgenden werden Anforderungen an die mobile Applikation zur Baufinanzierung aufgezeigt.

### 7.3.1.2 Spezifikation der Applikationsanforderungen

In Zusammenhang mit dem Immobilienverkauf zielen Beratungsdienstleistungen darauf ab, die persönliche Finanzsituation des Beratenen aufzuzeigen und mögliche finanzielle Freimittel offenzulegen. Die Reinvestierung dieser Freibeträge in verschiedene Formen der Finanzierung ermöglicht den Erhalt finanzieller Mittel zum Kauf oder zum Bau von Immobilien. Das Ziel der Beratungsdienstleistungen besteht somit in der Ermittlung maximaler Kaufpreise oder Objektkosten auf Grundlage mathematischer Berechnungsvorschriften.

Die Anforderungen an eine mobile Baufinanzierungs-Applikation müssen sich an den Prinzipien der Beratungsdienstleistungen orientieren, um zum einen wahrheitsgetreue und allgemeingültige Ergebnisse zu errechnen und zum anderen den Ansprüchen einer klassischen Beratung gerecht zu werden. Für die Erstellung eines exakten Anforderungsprofils müssen im Voraus unterschiedliche Kriterien bestimmt werden. Innerhalb der Betrachtung und Festlegung der Kriterien wurden diese in Muss-, Wunsch- und Ausschlusskriterien untergliedert. Musskriterien beinhalten diejenigen Anforderungen, die von der mobilen Applikation auf jeden Fall erfüllt und umgesetzt werden müssen. Sie stellen somit die Basis für den Erfolg des Projektvorhabens. In Abb. 7.12 sind die Musskriterien, die die Applikation erfüllen soll, gelistet.

Wunschkriterien beschreiben Anforderungen, die nach zeitlicher und technischer Möglichkeit realisiert werden können (siehe Abb. 7.13).

Ausschlusskriterien beschreiben Leistungen einer Anwendung, die bereits im Voraus unberücksichtigt bleiben (siehe Abb. 7.14).

Eine Validierung der Muss-, Wunsch- und Ausschlusskriterien wurde nicht durchgeführt. Die zuvor aufgelisteten Anforderungen wurden zudem ohne Rücksprache mit potenziellen Endnutzern festgelegt.

Die Rolle des Anforderungsmanagements in der Softwareentwicklung darf nicht unterschätzt werden. Eine Vielzahl an Fehlern und Problemen, die während des Entwicklungs-

Nr.	Wunschkriterien an die Baufinanzierungs-Applikation
1	Bereitstellung von Funktionen zur Ermittlung eines Baubudgets
2	Möglichkeit der Druckausgabe
3	Speicherung der Benutzerdaten
4	Angabe eines Impressums
5	Integration einer Hilfefunktion

**Abb. 7.13** Wunschkriterien an die Baufinanzierungs-Applikation

Nr.	Ausschlusskriterien an die Baufinanzierungs-Applikation
1	Keine Berücksichtigung von Sondertilgungen
2	Keine Ermittlung von Tilgungsplänen
3	Keine Berücksichtigung von Nach- und Anschlussfinanzierungen
4	Keine Berücksichtigung sonstiger abweichender Baufinanzierungsmodule, bspw. die Betrachtung von Lebensversicherungen

**Abb. 7.14** Ausschlusskriterien an die Baufinanzierungs-Applikation

prozesses auftreten können, beruhen auf einem mangelhaften Anforderungsmanagement. Die Kosten der Softwareentwicklung werden dadurch in die Höhe getrieben und die Auslieferungszeit der Software muss neu festgelegt werden. Im Zuge der Anforderungserhebung wurde die initiale Idee weiter verfeinert, sodass erste Lösungswege für das vorliegende Problem konstruiert werden können. In diesem Zusammenhang erfolgt die Erstellung eines Programmablaufplans, welcher die Interaktion des Nutzers mit der mobilen Applikation grafisch darstellt. Der Ablaufplan wird im folgenden Kapitel dargestellt und erläutert.

### 7.3.1.3 Darstellung des Programmablaufplans

Die Beschreibung von Algorithmen, Daten und Funktionen in natürlicher Sprache ist oftmals nur schwer realisierbar. Daher verwenden Softwareentwickler geeignete Modellierungsmethoden, um die zu beschreibenden Programmkonstrukte und Elemente anschaulich darzustellen. Im Laufe der Zeit haben sich hierzu mehrere Modellierungsmethoden etabliert, bspw. das ERM oder die EPK. Die Darstellung des Programmablaufs ermöglicht das Aufzeigen kausaler Zusammenhänge zwischen den einzelnen Schritten des Verarbeitungsprozesses. Für Softwareentwickler bieten die Modelle eine Möglichkeit, das weitere Projektvorgehen dem Kunden besser mitteilen zu können sowie die Schnittstellen zum System allgemein verständlicher präsentieren zu können. Die Modellierung von Programmablaufplänen wird vor allem bei der auftragsbezogenen Softwareentwicklung oftmals vorausgesetzt und als Bedingung in das Pflichtenheft mit aufgenommen.

Zur Modellierung des Programmablaufs der mobilen Applikation zur Baufinanzierung wird im nachfolgenden die Methode der BPMN verwendet. Der Ablaufplan stellt eine vereinfachte Sichtweise auf den Gesamtprozess der Ermittlung einer Baufinanzierung dar.

Bei der Erstellung des Ablaufplans wurden hauptsächlich die aufgestellten Musskriterien aus Abb. 7.12 berücksichtigt. Der Ablaufplan wird in Abb. 7.15 dargestellt.

Kernaussage des Ablaufplans besteht in der Beschreibung der Interaktion eines Benutzers mit der Applikation. Der Prozess wird durch den Benutzer ausgelöst, indem die Applikation über das mobile Endgerät ausgewählt wird. Die Applikation initialisiert den Programmstart und öffnet die Sicht „Hauptmenü“. Der Benutzer hat nun die Möglichkeit, zwischen mehreren Optionen zu wählen. Aus Gründen der Einfachheit wurden in Abb. 7.15 nur die Optionen „Baufinanzierung ermitteln“ und „Beenden“ aufgeführt. Die Ermittlung einer Baufinanzierung setzt die Eingabe von Benutzerdaten voraus, die zur weiteren Verarbeitung an die Applikation weitergeleitet werden. Die Applikation verarbeitet die Benutzerdaten und übergibt die ermittelten Ergebnisse an den Benutzer. Dieser steht nun vor der Wahl, entweder eine erneute Finanzierung unter anderen Konditionen zu ermitteln oder die Applikation zu beenden.

Die Darstellung des Programmablaufplans in Abb. 7.15 ist sehr abstrakt gehalten. Es ist daher anzumerken, dass die Aktivitäten „Baufinanzierung ermitteln“ und „Benutzerdaten verarbeiten“ aus mehreren Subprozessen mit weiteren zahlreichen Aufgaben bestehen. Die detaillierte Erläuterung dieser Aufgaben und Prozesse erfolgt in Abschn. 7.3.2. Nach der Konkretisierung von Projektzielen und Anforderungen an die Software werden in den nachfolgenden Kapiteln die Maßnahmen zur Umsetzung der mobilen Applikation beschrieben sowie Herausforderungen und Probleme innerhalb der Realisierungsphase erläutert.

### 7.3.2 Umsetzung der Anforderungen

Die Umsetzung einer gegebenen Anforderungsspezifikation stellt den Ausgangspunkt für die Erarbeitung einer fachlichen Lösung dar. Bevor diese jedoch durch die Entwicklung von Funktionen und Algorithmen verwirklicht werden kann, müssen Softwareunternehmen die erhobenen Anforderungen validieren und ggf. mit dem Auftraggeber abstimmen. Ausgehend der in Abschn. 7.3.1.2 gegebenen Anforderungsspezifikation erfolgte die Entwicklung der mobilen Anwendung zur Baufinanzierung.

Die Konzeption und Entwicklung des Designs bildete den Ausgangspunkt der Realisierungsphase. Auf Grundlage eines vorher konzipierten Papierprototyps erfolgte die Strukturierung und Gestaltung der Benutzeroberfläche. Die hierfür verwendeten Java-Steuerelemente wurden in einem weiteren Schritt durch die Entwicklung von Funktionen und Datenstrukturen erweitert. Die Datenstrukturen unterstützen die Realisierung von Funktionen, die in der Anforderungsspezifikation niedergeschriebene Kriterien an die Applikation repräsentieren. Anhand ausgewählter Beispiele werden nachfolgend implementierte Datenstrukturen und Funktionen näher betrachtet. Herausforderungen und Probleme während der Umsetzungen der Anforderungen sowie deren Lösung werden abschließend detailliert erläutert.

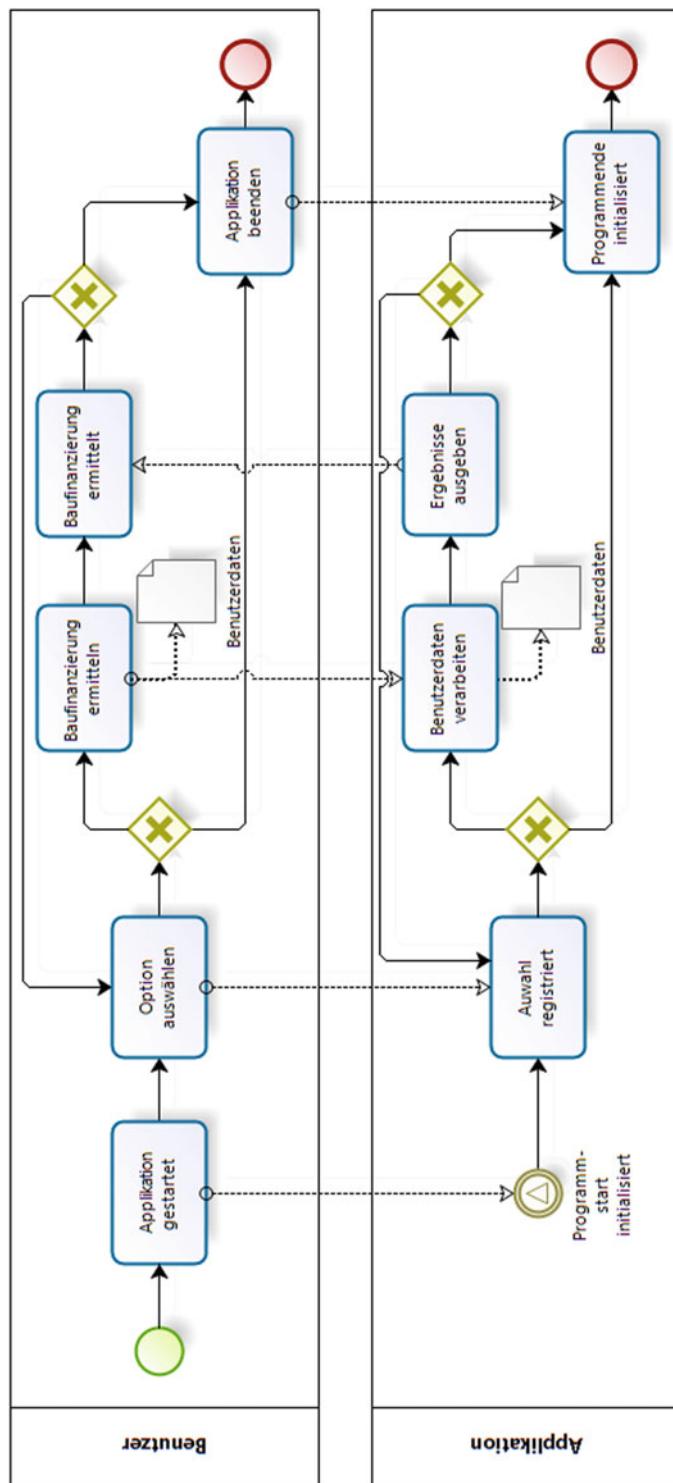


Abb. 7.15 Vereinfachte Darstellung des Programmablaufs

### 7.3.2.1 Anlegen der Applikation in der Entwicklungsumgebung

Softwareentwickler müssen bei der Herstellung von Anwendungssoftware verschiedene Programmierrichtlinien des Auftraggebers oder des Unternehmens berücksichtigen, bspw. die Vergabe eindeutiger Projektnamen, die einheitliche Dokumentation des Projektverlaufs aber auch die Vermeidung komplexer Programmfragmente sowie die Einhaltung festgelegter Entwicklungszeiten. Die Aufstellungen von firmen- sowie projektspezifischen Rahmenbedingungen soll die Gewährleistung einer hohen Softwarequalität sicherstellen. Aus technischer Sichtweise bildet das Anlegen eines Projektes in der jeweiligen Entwicklungsumgebung den Grundstein der Softwareentwicklung. Im Folgenden wird die Erstellung der mobilen Applikation mit App Inventor sowie erste Schritte im Umgang mit der App-Inventor-Entwicklungsumgebung beschrieben. Für das bessere Verständnis des Realisierungsvorgangs wird nachfolgend der Aufbau des App Inventors kurz erläutert.

Der App Inventor stellt eine onlinebasierende Entwicklungsumgebung dar, die in Verbindung mit einem gültigen Google-Account aufgerufen und verwendet werden kann. Die Entwicklungsumgebung ist eine Kombination aus Drag-and-drop-Editor (Design-Editor) sowie grafischer Programmierschnittstelle und erlaubt die Entwicklung mobiler Applikationen durch die Zusammenstellung festgelegter Java-Codeblöcke. Der Aufbau des Design-Editors gliedert sich für die Gestaltung der Applikation in fünf Kategorien auf:<sup>29</sup>

#### 1. Palette

Die Kategorie „Palette“ beinhaltet standardisierte Steuerelemente für die Erstellung von Textfeldern, Schaltflächen sowie Funktionen für die Kommunikation mit den gegebenen Hardwareressourcen, bspw. Kamera- oder Videoaufnahmen, Bewegungssensoren oder GPS-Funktionen.

#### 2. Viewer

Die Kategorie „Viewer“ beinhaltet den eigentlichen Designbereich, der in Form eines Smartphone-Displays das spätere Aussehen der Applikation simuliert. Die aus der Kategorie „Palette“ beinhalteten Steuerelemente werden per Maus beliebig auf dem Designbereich platziert.

#### 3. Components

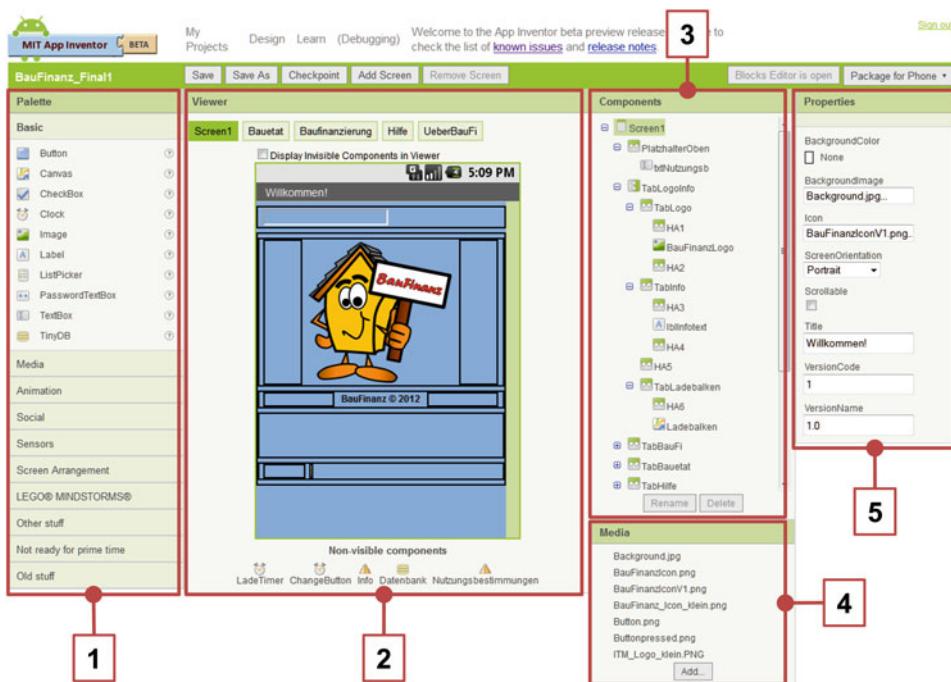
Die Kategorie „Components“ bildet alle im „Viewer“ abgebildeten Steuerelemente in einer hierarchischen Baumstruktur ab. Ziel ist die Darstellung über- und untergeordneter Komponenten mit identischen Eigenschaften.

#### 4. Media

Die Kategorie „Media“ listet alle durch den Entwickler auf die Entwicklungsumgebung hochgeladenen Mediendateien auf, bspw. Audio- oder Videodateien.

---

<sup>29</sup> Vgl. Kloss 2011, S. 64 ff.



**Abb. 7.16** Aufbau und Struktur des Design-Editors der App-Inventor-Entwicklungsumgebung

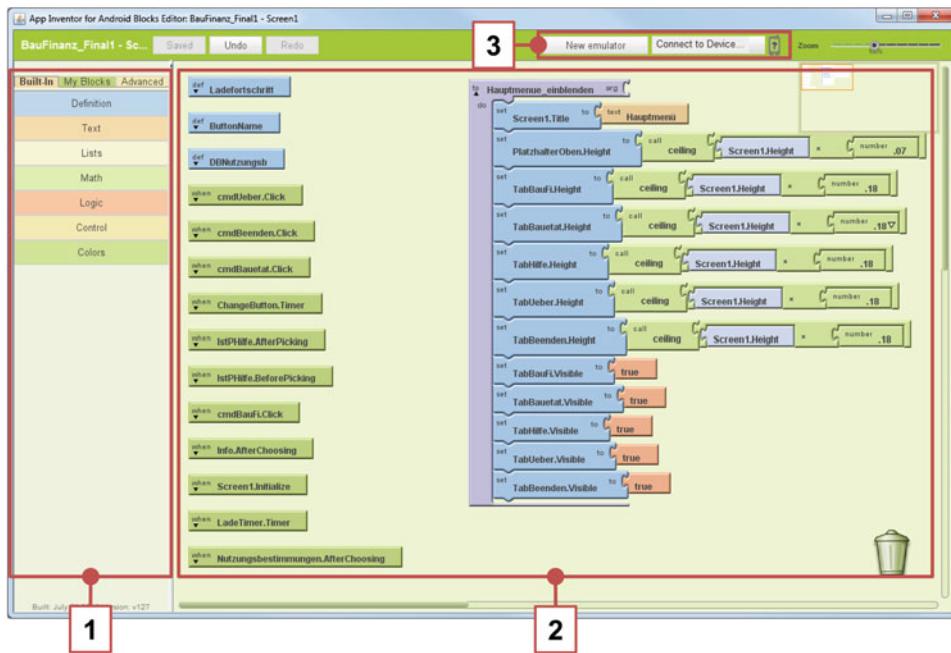
## 5. Properties

Die Kategorie „Properties“ ermöglicht den Zugriff auf die Eigenschaften eines ausgewählten Steuerelements, bspw. die Hintergrundfarbe oder Schriftart und -größe.

Der App Inventor stellt weiterhin Optionen zur Speicherung der Applikation auf einem stationären PC oder zur Übertragung der Anwendung an ein mobiles Endgerät zur Verfügung. Abbildung 7.16 zeigt den Aufbau des App Inventors aus der Sicht des Design-Editors.

Zur Fertigstellung der mobilen Anwendung müssen die im Blocks-Editor enthaltenen Java-Codeblöcke den jeweiligen Steuerelementen aus dem Design-Editor zugewiesen werden. Die Codeblöcke werden innerhalb des Blocks-Editors durch Puzzleteile repräsentiert, durch deren Zusammenfügen die Konfiguration der Steuerelemente vorgenommen wird. Da der Editor auf einer Java-Web-Start-Umgebung basiert, wird eine auf dem PC vorinstallierte Java-Konfiguration benötigt. Der Aufruf des Blocks-Editors erfolgt aus dem Design-Editor heraus und weist ebenfalls mehrere Funktionsbereiche auf.<sup>30</sup>

<sup>30</sup> Vgl. Kloss 2011, S. 75 ff.



**Abb. 7.17** Aufbau und Struktur des Blocks-Editors der App-Inventor-Entwicklungsumgebung

### 1. Blockauswahl

In der linken Spalte des Blocks-Editors werden unter den Reitern „Built-In“, „My Blocks“ und „Advanced“ verschiedene Java-Codefragmente in Form von Blockgruppen aufgelistet. Die Kategorie ist durch keine prägnante Überschrift gekennzeichnet und wird daher im Folgenden als Blockauswahl bezeichnet.

### 2. Arbeitsbereich

Der Arbeitsbereich stellt den eigentlichen Editor dar, in dessen Wirkungsbereich die Kombination mehrerer Codeblöcke aus der Blockauswahl zur Definition von Variablen, Verzweigungen oder Wiederholungen führt. Weiterhin dient ein Papierkorb zum Entfernen nicht mehr benötigter Codeblöcke aus dem Arbeitsbereich.

### 3. Android-Emulator

Zur Kontrolle und Validierung des bisherigen Entwicklungsprozesses bietet der Blocks-Editor die Möglichkeit, einen Android-Emulator zu starten. In Form eines virtuellen Smartphones können dadurch die bisherigen Ergebnisse, mit kleineren Einschränkungen in Bezug auf die Hardwarenutzung, simuliert werden.

Der Blocks-Editor bietet des Weiteren Funktionen zur Speicherung der aktuellen Entwicklung an. Der Aufbau des App Inventors aus der Sicht des Blocks-Editors wird in Abb. 7.17 dargestellt.

Das Hauptmenü des App Inventors listet die bisherigen Entwicklungsprojekte auf und sortiert diese chronologisch nach ihrem Erstellungszeitpunkt. Entwickler können über die Funktionen des Hauptmenüs neue Projekte erstellen sowie bisherige Projekte bearbeiten



**Abb. 7.18** Prototypen zur Simulation der Benutzeroberfläche der mobilen Applikation

oder verwerfen. Weiterhin besteht die Möglichkeit, externe Projekte in die Bibliothek aufzunehmen und zu editieren.

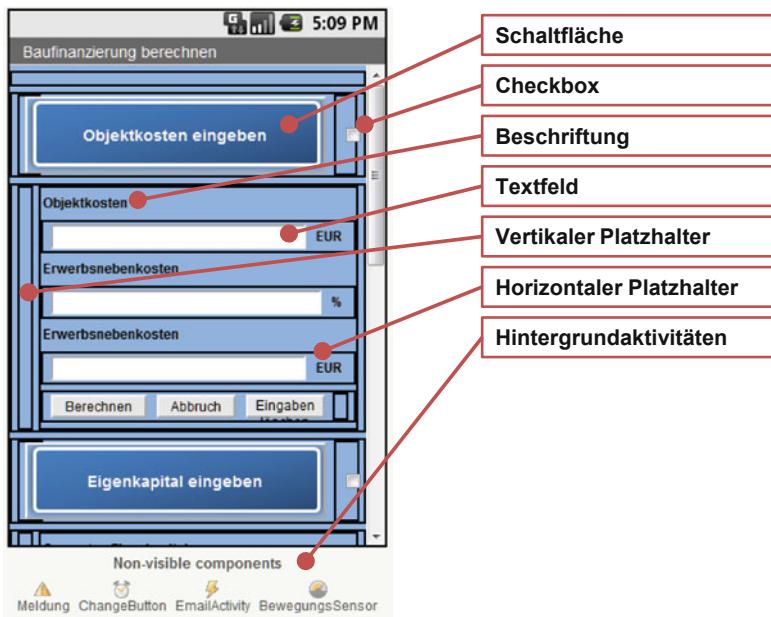
Anknüpfend an den Aufbau und die Struktur der App-Inventor-Entwicklungsumgebung wird nachfolgend das Vorgehen bei der Gestaltung der mobilen Applikation zur Baufinanzierung beschrieben.

### 7.3.2.2 Gestaltung und Entwicklung der Applikation

Die Strukturierung und Gestaltung der Benutzeroberfläche für mobile Endgeräte stellt Softwareentwickler vor neue Herausforderungen, da mobile Anwendungen im Gegensatz zu Desktopanwendungen besondere Rahmenbedingungen, bspw. unterschiedliche Bildschirmgrößen, berücksichtigen müssen. Die Strukturierung der Benutzeroberfläche ist somit für den Erfolg der Applikation entscheidend und muss daher benutzerfreundlich und ergonomisch gestaltet werden. Vor diesem Hintergrund werden durch die Hersteller Empfehlungen und Vorgaben für die Gestaltung mobiler Applikationen ausgesprochen und im Internet zum Download angeboten.<sup>31</sup>

Das UCAN-Modell sieht vor der eigentlichen Entwicklung der Applikation die Realisierung eines Papierprototyps vor, an dem die Gestaltung der Benutzeroberfläche simuliert sowie die Anzahl benötigter Steuerelemente ermittelt werden kann. Hinsichtlich dieser Vorgabe durch das UCAN-Modell wurde zu Beginn der Entwurfsphase ein Storyboard entwickelt, welches eine umfassende Sicht auf die einzelnen Benutzeroberflächen der Applikation sowie den gesamten Ablauf der mobilen Baufinanzierung darstellt. Abbildung 7.18 zeigt eine Auswahl der Prototypen zur Simulation der Struktur der Benutzeroberflächen.

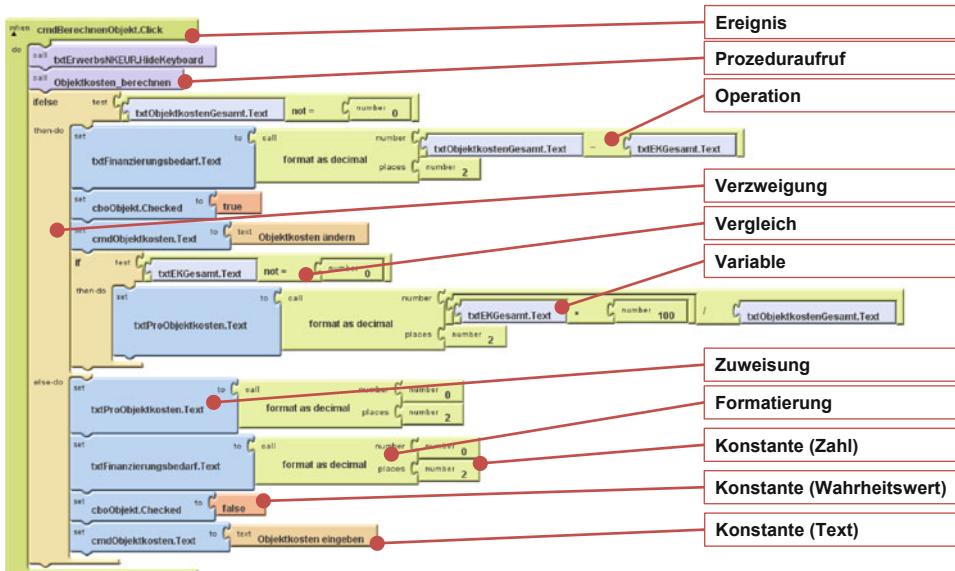
<sup>31</sup> Vgl. Kloss 2011, S. 103.



**Abb. 7.19** Umsetzung der Oberflächengestaltung in App Inventor

In einem weiterführenden Schritt erfolgte die Umwandlung der Papierprototypen aus dem Storyboard in erste dynamische Prototypen. Die zur Umsetzung benötigten Steuerelemente und Benutzeroberflächen wurden durch die Entwurfsphase ermittelt und mittels des Design-Editors digital umgesetzt. Anhand der nachfolgenden Abb. 7.19 werden die zur Erstellung der Benutzeroberflächen eingesetzten Steuerelemente dargestellt und deren Einsatz in der mobilen Anwendung kurz erläutert.

Die Schaltfläche bildet eine der am häufigsten eingesetzten Steuerelemente in der Softwareentwicklung und dient als grafische Benutzerschnittstelle zur Auslösung von Aktionen oder zur Bestätigung von Abfragen. Aufgrund der in App Inventor gegebenen Gestaltungsmöglichkeiten konnten Schaltflächen facettenreich in der Baufinanzierungs-Applikation eingesetzt werden, bspw. für den Aufruf der Baufinanzierungsfunktion oder die Berechnung unterschiedlicher Finanzierungsmöglichkeiten. Die Einstellung einer Software, bzw. die Kontrolle eines bestimmten Zustands einer Software, wird normalerweise durch den Einsatz einer Checkbox (Kontrollkasten) realisiert. Innerhalb der Applikation zur Baufinanzierung werden Checkboxen als Indikator für eine getätigte Aufgabe eingesetzt, bspw. den erfolgreichen Abschluss der Eigenkapitalermittlung. Beschriftungen haben die Aufgabe, einen beliebigen Text an den Benutzer der Applikation auszugeben. In der Baufinanzierungs-Applikation werden Beschriftungen zur Gestaltung des Textes sowie für die Vergabe von Überschriften oder für die Angabe von metrischen Einheiten verwendet. Neben den bereits angesprochenen Schaltflächen bilden Text- oder Eingabefelder eine weitere bedeutende Rolle innerhalb der Softwareentwicklung. Durch den Einsatz



**Abb. 7.20** Umsetzung der Funktionen im Blocks-Editor

von Textfeldern wird die Möglichkeit zur Entgegennahme und Verarbeitung von Texteingaben durch den Benutzer ermöglicht. In der mobilen Anwendung zur Baufinanzierung werden Textfelder einerseits für die Eingabe persönlicher Finanzdaten und andererseits für die Ausgabe ermittelter Ergebnisse verwendet.

Neben den gerade besprochenen aktiven Steuerelementen, die dem Benutzer zu jeder Zeit sichtbar zur Verfügung stehen, beinhaltet die mobile Applikation unsichtbare Strukturierungselemente sowie im Hintergrund operierende Funktionen. Horizontale und vertikale Platzhalter dienen der räumlichen Anordnung und Abgrenzung von anderen Steuerelementen der Benutzeroberfläche. Nichtsichtbare Komponenten, wie bspw. die Ausführung einer Zeitfunktion (vgl. Abb. 7.19, „ChangeButton“) oder der Aufruf einer externen Anwendung (vgl. Abb. 7.19, „EmailActivity“), werden nicht im sichtbaren Bereich des „Viewers“ aufgenommen, jedoch im Blocks-Editor für die Zuordnung von Funktionen zusammen mit den sichtbaren Komponenten aufgelistet.

Anhand des in Abb. 7.20 aufgeführten Beispiels soll repräsentativ die Zuweisung der Codeblöcke aus dem Blocks-Editor zu den Steuerelementen aus dem Design-Editor beschrieben werden. Die Beschreibung der Realisierung notwendiger Baufinanzierungsfunktionen erfolgt in Abschn. 7.3.2.3 Der Aufruf sowie die Ausführung einer Funktion innerhalb einer Applikation benötigt das Eintreten eines Ereignisses, bspw. durch Betätigen einer Schaltfläche, aber auch durch bestimmte Systemzustände. Im dargestellten Beispiel werden beide Alternativen verwendet. Das Ereignis „cmdBerechnenObjekt“ wird durch Betätigung der Schaltfläche „Objektkosten eingeben“ (vgl. Abb. 7.19) aufgerufen sowie die dem Ereignis zugehörigen Programmschritte der Reihe nach bearbeitet. In gleicher

Weise können Programmschritte durch den Prozedurauftrag gestartet werden. Im obigen Beispiel werden unmittelbar nach dem Start des Ereignisses zwei Prozeduren aufgerufen. Nach Beendigung dieser wird die Bearbeitung der restlichen Programmschritte und Methoden des ursprünglichen Ereignisses fortgesetzt. Weitere typische Programmfragmente, die bei der Implementierung der Applikation eingesetzt wurden, sind Verzweigungen, Zuweisungen, Formatierungen, Vergleichs- und Rechenoperationen sowie Variablen und Konstanten.

Neben der Entwicklung der eigentlichen Problemlösung ist der Entwurf einer ansprechenden sowie benutzerfreundlichen Anwendung entscheidend für den Erfolg des Projektes sowie die Akzeptanz beim Endnutzer. In unterschiedlichen Einsatzgebieten von mobilen Endgeräten und Applikationen hat sich gezeigt, dass Benutzungsfehler häufig auf eine unpassende Gestaltung der Mensch-Maschine-Schnittstelle zurückzuführen ist. Die gebrauchstaugliche Entwicklung dieser Schnittstellen für die sichere und fehlerfreie Nutzung mobiler Applikationen ist somit unverzichtbar. Im nachfolgenden Kapitel wird genauer auf die Umsetzung der Funktionen und Datenstrukturen eingegangen und werden ausgewählte Entwicklungsbeispiele aufgezeigt.

### 7.3.2.3 Funktionen und Datenstrukturen der Applikation

Funktionen und Daten repräsentieren die Leistungsfähigkeit der mobilen Applikation in Bezug auf ein bestimmtes Problem oder eine Problemsituation und werden hinsichtlich der im Rahmen der Anforderungserhebung ermittelten Ziele und Anforderungen entwickelt. Im Folgenden werden die Funktionen der Baufinanzierungs-Applikation genannt und anhand mehrerer Beispiele ein Einblick in die Übertragung der Anforderungen und Ziele in Programmcode gegeben. Eine grobe Gesamtsicht auf die entwickelten Basisfunktionen wird in Abb. 7.21 in Form eines Funktionsdiagramms dargestellt.

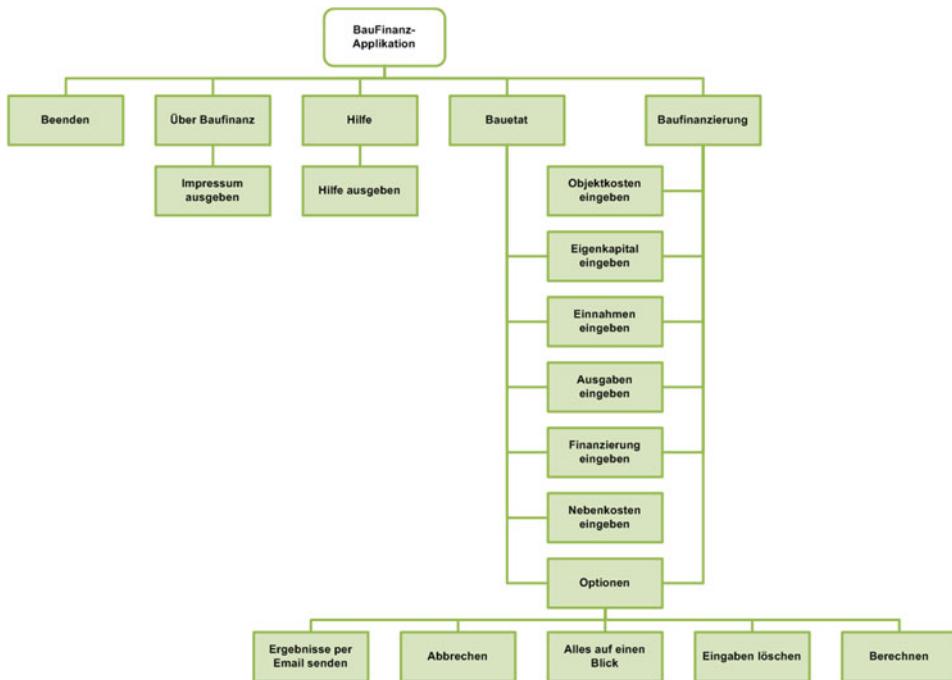
Zum besseren Verständnis der einzelnen Programmausschnitte werden zusätzlich folgende Dimensionen aufgeführt:

- Der Name der Funktion.
- Die an der Funktion beteiligten Akteure.
- Die Vorbedingungen der Funktion.
- Das mit der Funktion verbundene Szenario.

#### 7.3.2.3.1 Realisierung grundlegender Programmkonstrukte

Zunächst werden grundlegende Programmkonstrukte beschrieben, die zur Koordination des Ablaufs der Applikation sowie der Kommunikation einzelner Codefragmente untereinander benötigt werden. Vor diesem Hintergrund wird im Folgenden beispielhaft die Deklaration von Variablen, die Erzeugung von Prozeduren zur Kalibrierung der Steuerlemente sowie Methoden zum Aufrufen von Benutzeroberflächen erläutert. Abbildung 7.22 beschreibt die Definition einer Variablen.

Die Deklaration von Variablen dient der Zuteilung von Datentypen zu einer noch undefinierten Datenstruktur. Im obigen Beispiel wird der Variablen „Ladefortschritt“ die

**Abb. 7.21** Funktionsbaum zur mobilen Applikationsentwicklung

def Ladefortschritt as number 0	
Deklaration einer Variable	
Akteure	Applikation
Vorbedingungen	Keine; wird durch den Entwickler festgelegt
Szenario	Kein typisches Szenario aufzuweisen; dienen zum Programmablauf

**Abb. 7.22** Deklaration einer Variablen

Zahl „Null“ übergeben und dadurch impliziert, dass die Variable den Datentyp „Integer“ („number“) repräsentiert. Dies bedeutet, dass die Variable „Ladefortschritt“ innerhalb des Programmcodes nur numerische Werte annehmen darf. Analog verhält es sich mit Variablen des Datentyps „Charakter“ („text“). Variablen werden innerhalb der Baufinanzierungsapplikation für die Aufnahme und Weitergabe benutzerindividueller Daten verwendet, bspw. für die Ermittlung finanzmathematischer Rechnungen. Neben dem Einsatz von Variablen werden weiterhin Konstanten für gleichbleibende programm spezifische Aufgaben definiert, bspw. für die Skalierung der Steuerelemente (vgl. Abb. 7.23).

Der in Abb. 7.23 dargestellte Programmausschnitt ermöglicht die Skalierung des Platzhalters „PlatzhalterOben“, durch die Änderung der Höhe des Steuerelements. Die Funktion

	
<b>Skalierung der Steuerelemente</b>	
<b>Akteure</b>	Benutzer, Applikation
<b>Vorbedingungen</b>	a Benutzer hat die Applikation gestartet
<b>Szenario</b>	1 Benutzer startet die Applikation auf dem mobilen Endgerät

**Abb. 7.23** Skalierung der Steuerelemente

	
<b>Aufruf einer Benutzeroberfläche</b>	
<b>Akteure</b>	Benutzer, Applikation
<b>Vorbedingung</b>	a Zugehörige Schaltfläche ausgewählt
<b>Szenario</b>	1 Benutzer möchte zwischen den einzelnen Sichten der Applikation wechseln 2 Zugehörige Schaltfläche auswählen

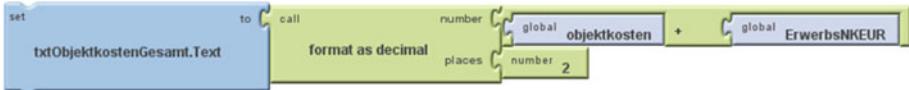
**Abb. 7.24** Aufruf einer Benutzeroberfläche

„ceiling“ ermittelt hierfür ein prozentuales Höchstmaß des Platzhalters, unter Berücksichtigung der durch das mobile Endgerät gegebenen Bildschirmhöhe. Im Beispiel wird die Höhe des Platzhalters auf 20 % der Bildschirmgröße gesetzt. Diese Skalierung ermöglicht die identische Darstellung der Steuerelemente auf unterschiedlichen Bildschirmen und wird bei der Initialisierung der Applikation auf alle Steuerelemente angewendet. Eine weitere wichtige Funktion ist der Aufruf einer Benutzeroberfläche, welcher in Abb. 7.24 aufgezeigt wird.

Die Verwendung mehrerer Benutzeroberflächen ermöglicht die Verteilung und bessere Strukturierung der Applikationsfunktionen auf mehrere Sichten. Der Wechsel zwischen den einzelnen Oberflächen wird wie oben dargestellt durchgeführt. Weiterhin besteht durch den Wechsel die Möglichkeit des Datentransfers („open another screen with start value“). Im nächsten Abschnitt wird auf die Programmierung finanzmathematischer Rechnungen eingegangen.

### 7.3.2.3.2 Programmierung finanzmathematischer Rechnungen

Zur Beschreibung der Umsetzung finanzmathematischer Rechnungen wird hierzu exemplarisch die Ermittlung der gesamten Objektkosten, die Berechnung des Eigenkapitals in Prozent zu den Objektkosten sowie die Ermittlung einer Finanzierungsrate genannt. Diese Berechnungen sind der Funktion „Baufinanzierung“ untergliedert und bilden den Ausgangspunkt der Ermittlung der Immobilienfinanzierung. Abbildung 7.25 beschreibt die Ermittlung der gesamten Objektkosten.

	
<b>Berechnung der gesamten Objektkosten</b>	
<b>Akteure</b>	Benutzer, Applikation
<b>Vorbedingungen</b>	<p>a Objektkosten eingetragen</p> <p>b Erwerbsnebenkosten (Euro oder Prozent) eingetragen</p> <p>c Schaltfläche "Berechnen" ausgewählt</p>
<b>Szenario</b>	<p>1 Schaltfläche "Objektkosten eingeben" auswählen</p> <p>2 Objektkosten eingeben</p> <p>3 Erwerbsnebenkosten (Euro oder Prozent) eingeben</p> <p>4 Schaltfläche "Berechnen" auswählen</p>

**Abb. 7.25** Berechnung der gesamten Objektkosten

	
<b>Berechnung des Eigenkapitals in Prozent der Objektkosten</b>	
<b>Akteure</b>	Benutzer, Applikation
<b>Vorbedingungen</b>	<p>a Gesamte Objektkosten ermittelt</p> <p>b Gesamtes Eigenkapital ermittelt</p>
<b>Szenario</b>	<p>1 Schaltfläche "Objektkosten eingeben" auswählen</p> <p>2 Objektkosten ermitteln (vereinfacht)</p> <p>3 Schaltfläche "Eigenkapital eingeben" auswählen</p> <p>4 Eigenkapital ermitteln (vereinfacht)</p>

**Abb. 7.26** Berechnung des Eigenkapitals in Prozent der Objektkosten

Zur Berechnung der gesamten Objektkosten werden zunächst die durch den Benutzer eingegebenen Objektkosten mit den Erwerbsnebenkosten addiert. Das Ergebnis der Addition wird in eine Dezimalzahl mit insgesamt zwei Nachkommastellen umgewandelt und anschließend an das Textfeld „txtObjektkostenGesamt.Text“ übergeben. Innerhalb der Baufinanzierungs-Applikation wurden Additionsverfahren weiterhin für die Ermittlung des gesamten Eigenkapitals oder gesamter Ein- und Ausgaben verwendet. Die Übertragung der Prozentrechnung wird nachfolgend in Abb. 7.26 dargestellt.

Die Ermittlung des Eigenkapitals in Prozent der Objektkosten errechnet sich aus der Division der gesamten Objektkosten mit dem Produkt aus Eigenkapital und dem Wert 100. Vor der Zuweisung des ermittelten Prozentsatzes an das Textfeld „txtProObjektkosten.Text“ wird das Ergebnis in eine Dezimalzahl mit zwei Nachkommastellen formatiert. Die Anwendung der Prozentrechnung wurde weiterhin für die Berechnung der

<pre> set txtRate1.Text to call   format as decimal     places 2   number [global Kreditbetrag1] / number 100 * [global Nominalzins1] + [global Tilgung1] / number 12 </pre>	
<b>Berechnung einer Finanzierungsrate</b>	
<b>Akteure</b>	Benutzer, Applikation
<b>Vorbedingungen</b>	<p>a Kreditbetrag eingetragen</p> <p>b Nominalzins eingetragen</p> <p>c Tilgungssatz eingetragen</p> <p>d Zinsbindung eingetragen</p> <p>e Schaltfläche "Berechnen" ausgewählt</p>
<b>Szenario</b>	<p>1 Schaltfläche "Finanzierung eingeben" auswählen</p> <p>2 Finanzierungskonditionen eintragen (vereinfacht)</p> <p>3 Schaltfläche "Berechnen" auswählen</p>

**Abb. 7.27** Berechnung einer Finanzierungsrate

Erwerbsnebenkosten in Prozent sowie für die in Abb. 7.27 dargestellte Ermittlung einer Finanzierungsrate verwendet.

Das Ergebnis der Gleichung aus Abb. 7.27 bildet die Ermittlung einer monatlichen Rate, die als Dezimalzahl mit zwei Nachkommastellen an das Textfeld „txtRate1.Text“ weitergegeben wird. Innerhalb der Applikationen können insgesamt zwei unterschiedliche Finanzierungsarten auf Grundlage der in Abb. 7.27 dargestellten Prozedur ermittelt werden.

Die bei der Realisierung der mobilen Applikation zur Baufinanzierung aufgetretenen Probleme und Herausforderungen stehen im Mittelpunkt der Betrachtungen der nachfolgenden Abschnitte. Insbesondere wird bei den Ausführungen auf die Identifikation und Eliminierung von Schwachstellen in Bezug auf die Gestaltung der Benutzeroberflächen eingegangen.

### 7.3.2.4 Beschreibung aufgetretener Probleme

Die Evaluierung von Software ist ein notwendiger Vorgang zur Kontrolle und Überprüfung der Produkteigenschaften. Hierzu müssen Softwaretests durchgeführt werden, die im bestmöglichen Fall den Nachweis einer mangelfreien Eignung der Software erbringen. Das Testen von Software zeigt weiterhin auf, in welcher Weise eine Software auf bestimmte Aufforderungen reagiert und somit für den Benutzer geeignet ist. Entwickler verwenden Softwaretests zudem für die Darstellung von Einschränkungen oder Schäden bei der typischen Produktanwendung.

Die Evaluierung des ersten funktionsfähigen Prototyps hat ergeben, dass hohe Ladezeiten beim Aufruf der Applikation sowie beim Wechsel zwischen den jeweiligen Benutzeroberflächen den reibungslosen Ablauf der Applikation verhindern. Zunächst erhob sich der Verdacht, dass die Größe der eingesetzten grafischen Elemente der wesentliche Grund für die immensen Ladezeiten ist. Die Abbildungen wurden daraufhin komprimiert

**Abb. 7.28** Komplexität vor der Verbesserung der Laufzeit

Operanden	Verwendung	Operatoren	Verwendung
Horizontale Platzhalter	29	Textfelder	4
Vertikale Platzhalter	0	Beschriftungen	8
		Schaltflächen	2
<b>n1 = 2</b>	<b>N1 = 29</b>	<b>n2 = 3</b>	<b>N2 = 14</b>

**Abb. 7.29** Ergebnisse der Halstead-Metriken vor der Verbesserung der Laufzeit

Halstead-Metrik	Bezeichnung	Formel	Ergebnis
Größe d. Alphabets	n	$n = n1 + n2$	5
Länge	N	$N = N1 + N2$	43
Volumen	V	$V = N * \log_2 n$	99,84
Schwierigkeit	D	$D = (n1/2) * (N2/2)$	7
Aufwand	E	$E = D * V$	465,25

und ein anderes Speicherformat gewählt, welches zusätzlich die Größe der Grafiken verringerte. Die daran anknüpfende Evaluierung erbrachte jedoch identische Ergebnisse, sodass die Grafiken in der ursprünglichen Qualität erneut verwendet wurden.

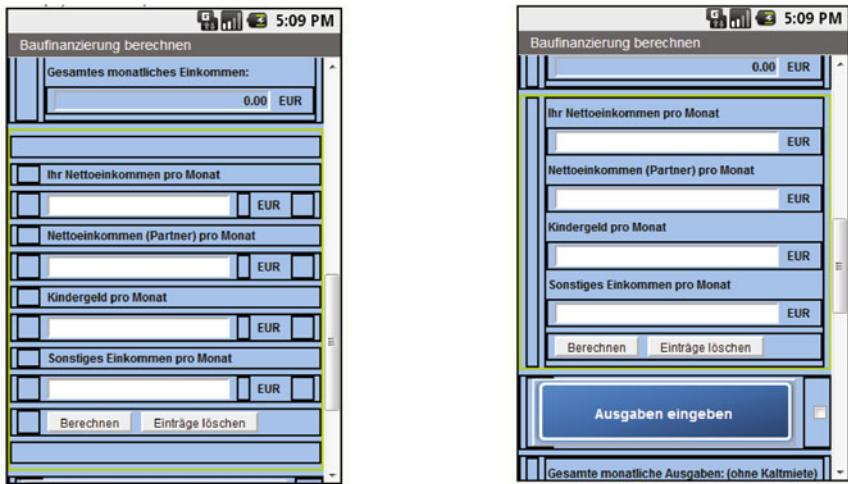
In einem nachfolgenden Schritt wurde zur Lösung des Problems die Gesamtgröße der Applikation betrachtet. Hierzu wurde das Projekt aus dem App Inventor auf die Festplatte des eingesetzten Rechners gespeichert. Durch den Vergleich des aktuellen Projektstands mit den Vorgängerversionen des Projektes wurde ersichtlich, dass die Größe der Applikation durch jede neu hinzukommende Benutzeroberfläche enorm anstieg. Aufgrund dieser Tatsache wurde eine Ist-Analyse zur Erhebung der Schwachstellen in Bezug auf die Struktur der Benutzeroberflächen durchgeführt. Die Analyse der Struktur der jeweiligen Benutzeroberflächen wurde mithilfe der Halstead-Metriken durchgeführt. Hierzu war eine Abwandlung der Metriken auf das vorliegende Problem notwendig:

- Als Operanden wurden alle horizontalen und vertikalen Platzhalter definiert.
- Textfelder, Beschriftungen und Schaltflächen beschreiben die notwendigen Operatoren.

Zur Erläuterung der Ergebnisse der Halstead-Metriken wird nachfolgend das Verfahren zur Ermittlung des gesamten monatlichen Einkommens betrachtet (vgl. Abb. 7.30). In Anlehnung an die zuvor beschriebene Abwandlung der Metriken können vor der Verbesserung der Laufzeit die in Abb. 7.28 dargestellten Operanden und Operatoren ermittelt werden.

Die Grundgesamtmenge an Operanden beträgt zwei, die an Operatoren drei. Insgesamt wurden 29 horizontale Platzhalter, vier Textfelder, acht Beschriftungen und zwei Schaltflächen zur Realisierung der Ermittlung des monatlichen Einkommens eingesetzt. Die erhobenen Daten führen zu den in Abb. 7.29 dargestellten Ergebnissen.

Die Ergebnisse der Halstead-Metriken zeigen, dass die erhöhte Laufzeit der Applikation auf die Vielzahl an eingesetzten horizontalen und vertikalen Platzhaltern zurückzuführen ist.



Eingabe der Einkommenssituation vorher

Eingabe der Einkommenssituation nachher

**Abb. 7.30** Eingabe der Einkommenssituation vor und nach der Verbesserung der Laufzeit**Abb. 7.31** Komplexität nach der Verbesserung der Laufzeit

Operanden	Verwendung	Operatoren	Verwendung
Horizontale Platzhalter	6	Textfelder	4
Vertikale Platzhalter	2	Beschriftungen	8
		Schaltflächen	2
$n1 = 2$	$N1 = 8$	$n2 = 3$	$N2 = 14$

ren ist. Die ermittelte Höhe des Volumens sowie die Länge weisen weiterhin auf einen hohen Speicherbedarf der eingesetzten Steuerelemente hin. Aufgrund der errechneten Halstead-Metriken erfolgte zur Verbesserung der Applikation die Umstrukturierung der Benutzeroberflächen. Hierzu wurden in einem ersten Schritt für die Ermittlung der Baufinanzierung notwendige Steuerelemente ermittelt und von austauschbaren Komponenten abgegrenzt. Die Abgrenzung erlaubt somit den Erhalt der Steuerelemente, denen bereits im Blocks-Editor fertige Funktionen und Aufgaben zugewiesen wurden. Im Zuge der Neugestaltung wurde daher die Struktur der einzelnen Benutzeroberflächen, hauptsächlich durch die Eliminierung horizontaler Platzhalter, verbessert. Das Ergebnis der Strukturüberarbeitung wird in Abb. 7.30 dargestellt.

Die Anzahl der eingesetzten Operanden und Operatoren hat sich nach der Optimierung der Struktur nicht verändert. In Bezug auf den Einsatz der Steuerelemente werden aufgrund der zuvor durchgeföhrten Reduzierung horizontaler Platzhalter die in Abb. 7.31 angeführten Änderungen ersichtlich.

Halstead-Metrik	Bezeichnung	Formel	Ergebnis vorher	Ergebnis nachher
Größe d. Alphabets	n	$n = n_1 + n_2$	5	5
Länge	N	$N = N_1 + N_2$	43	22
Volumen	V	$V = N * \log_2 n$	99,84	51,08
Schwierigkeit	D	$D = (n_1/2) * (N_2/2)$	7	7
Aufwand	E	$E = D * V$	465,25	238,03

**Abb. 7.32** Gegenüberstellung der Halstead-Metriken vor und nach der Verbesserung

Aus der Gegenüberstellung der zuvor ermittelten Metriken mit den Ergebnissen nach der Optimierung wird eine deutliche Verbesserung der Gesamtsituation erkennbar (vgl. Abb. 7.32).

Die Gegenüberstellung hat ergeben, dass die Größe und Schwierigkeit der Struktur nach der Optimierung konstant blieb. Im Zuge der Neugestaltungen konnte die Zeichenlänge von 43 Punkten auf 22 Punkte verringert sowie eine Verminderung des Volumens von 99,84 Punkten auf 51,08 Punkte ermöglicht werden. Ebenso konnte eine Verbesserung hinsichtlich des Strukturgrades der jeweiligen Benutzeroberfläche, welche durch die Dimension „Aufwand“ dargestellt wird, von 698,88 Punkten auf 357,56 Punkte dezimiert werden.

Die anhand des aufgezeigten Beispiels erreichte Laufzeitenverbesserung führte zur Umsetzung des neuen Layouts auf allen für die Applikation benötigten Benutzeroberflächen. Insgesamt konnten dadurch 188 horizontale und vertikale Platzhalter eingespart werden. Nachfolgend wird der gesamte Entwurfs- und Implementierungsprozess der mobilen Applikation nochmals zusammenfassend dargestellt.

### 7.3.3 Zusammenfassung

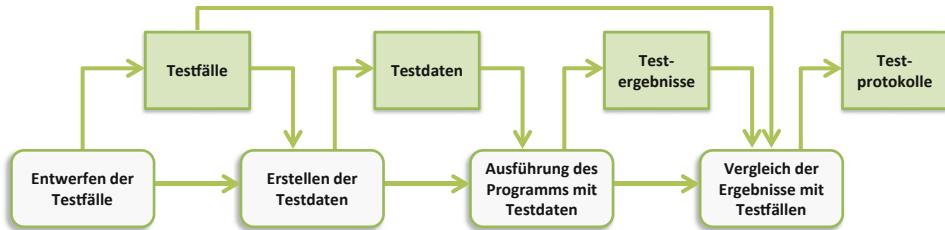
Zu Beginn der Entwurfsphase erfolgte auf Grundlage des zuvor aufgestellten Projektplans zunächst die Erhebung der Anforderungen an die mobile Applikation. Anhand allgemeiner Fragen zum Thema Baufinanzierung wurden die zu erreichenden Ziele abgeleitet. Gesamtziel des Softwareprojektes ist die Entwicklung einer Schnittstelle zwischen Kunden, Kreditinstituten und Immobilienmaklern. Anknüpfend an die Bestimmung der Ziele wurden Anforderungen an die Applikation genannt. Diese unterteilten sich in Muss-, Wunsch- und Ausschlusskriterien. Musskriterien an die Applikation sind bspw. die Ermittlung wahrheitsgetreuer Finanzierungswerte sowie die Gewährleistung der Datensicherheit. Zu den Wunschkkriterien gehörten unter anderem die Möglichkeit der Druckausgabe sowie die Integration einer Hilfefunktion. Bereits zu Beginn der Entwicklung wurden Funktionen zur Ermittlung von Sondertilgungen sowie zur Darstellung von Tilgungsplänen ausgeschlossen.

Im Anschluss an die Anforderungserhebung erfolgte die Umsetzung der Anforderungen in Programmcode. Hierzu wurde die Entwicklungsumgebung App Inventor verwendet, die zum besseren Verständnis nochmals detaillierter beschrieben wurde. Der Implementierungsprozess setzte sich aus den Phasen „Anlegen der Applikation“, „Gestaltung und Entwicklung der Applikation“ sowie in der „Realisierung von Funktionen und Datenstrukturen“ zusammen. Die erste Phase beschreibt die Erstellung eines neuen Projektes im App Inventor sowie die hier zu berücksichtigenden Rahmenbedingungen der Entwicklungsumgebung. Innerhalb der zweiten Phase erfolgte die Erstellung der jeweiligen Benutzeroberflächen der Applikation durch die Zusammenstellung notwendiger Steuerelemente im Design-Editor des App Inventors. Die letzte Phase beschreibt die Realisierung grundlegender Funktionen der Applikation sowie die Umsetzung finanzmathematischer Berechnungen mithilfe des Blocks-Editors. Zur Veranschaulichung des gesamten Entwicklungsprozesses wurden hierzu ausgewählte Beispiele aufgezeigt und detailliert erläutert.

Das vorliegende Kapitel endete mit der Beschreibung aufgetretener Probleme und deren Lösung bei der Realisierung der mobilen Applikation. Durch die Evaluierung des ersten funktionsfähigen Prototyps konnten lange Ladezeiten beim Aufruf einer Benutzeroberfläche sowie bei einigen Funktionen der Applikation festgestellt werden. Die Ermittlung der Komplexität der Applikationsstruktur unter Berücksichtigung der Halstead-Metriken ergab, dass die hohe Anzahl an eingesetzten horizontalen und vertikalen Platzhaltern die schlechten Laufzeiten der Applikation bedingen. Durch die daraus notwendige Umgestaltung der Struktur konnten bessere Laufzeiten und ein reduzierter Speicherbedarf ermöglicht werden. Die Anwendung des neuen Layouts auf die restlichen Benutzeroberflächen führte insgesamt zu einer Verbesserung der Applikation ohne Einschnitte oder Beschränkungen in der Benutzerfreundlichkeit.

An dieser Stelle sei abschließend die Anwendung des App Inventors in Kombination mit dem UCAN-Modell zu nennen. Ein besonderer Vorteil des App Inventors konnte erst durch die Anwendung der Entwicklungsumgebung erkannt werden, nämlich die exakte Umsetzung der Phasen des UCAN-Modells bzgl. der Entwicklung, Evaluierung und Weiterentwicklung von Prototypen. In diesem Zusammenhang ermöglicht der App Inventor die Entwicklung erster klick- und lauffähiger Prototypen durch die Zusammenstellung von Steuerelementen im Designer-Editor und die anschließende Zuweisung von Codefragmenten im Blocks-Editor für die Weiterentwicklung der Prototypen zu einer lauffähigen Applikation. Dieses Verfahren bewirkte eine schnelle Realisierung aller aufgestellten Mustskriterien sowie einiger Wunschkriterien, bspw. die Bereitstellung von Funktionen zur Ermittlung eines Bauetats, die Angabe eines Impressums sowie die Integration einer Hilfefunktion.

Die Beschreibung der Zielerreichung sowie die Evaluierung der mobilen Applikation zur Baufinanzierung werden im folgenden Kapitel behandelt. Hierfür wurde zum einen anhand einer aufgestellten Fallstudie die Funktionen der Applikation getestet und zum anderen die Anwendung an ausgewählte Testpersonen ausgehändigt und mithilfe eines Fragebogens Daten über die Leistungsfähigkeit der Applikation erhoben.



**Abb. 7.33** Ablauf des Fehlertests. (In Anlehnung an: Sommerville 2001, S. 449)

## 7.4 Evaluierungsphase

Aufgabe der Evaluierung von Softwaresystemen besteht in der Analyse und Bewertung entwickelter Softwarespezifikationen. Ziel ist die Überprüfung der Erfüllung vorher definierter Anforderungen an die Software sowie die Sicherstellung der Softwarequalität. Hierzu haben sich Komponenten- und Integrationstests als typische Maßnahmen der Qualitätssicherung etabliert. Die Phase der Komponententests beschäftigt sich mit dem Testen von Funktionen und Methoden einzelner Softwarekomponenten. Während des Integrationstests werden diese Komponenten zu einem Gesamtsystem integriert und das Zusammenwirken aller Komponenten sowie die Funktionalität und Geschwindigkeit des Gesamtsystems evaluiert. Im Gegensatz zu Komponententests, welche hauptsächlich durch die Softwareentwickler selbst realisiert werden, erfolgt die Durchführung von Integrationstests in Zusammenarbeit mit mehreren unabhängigen Testpersonen.<sup>32</sup>

Die Ermittlung der Qualität der Baufinanzierungs-Applikation erfolgte ebenfalls nach den Prinzipien der Komponenten- und Integrationstests. Zunächst wurde durch den Entwickler der Applikation in Form eines Fehlertests der einzelnen Komponenten der Anwendung überprüft. Das Ziel des Fehlertests bestand in der Ermittlung und Auffindung der im Anwendungssystem beinhalteten Fehler vor dessen Fertigstellung und Freigabe zur Nutzung. Unterstützt wurde der Fehler test anhand eines selbst erstellten Fallbeispiels basierend auf realen Testdaten. Der Ablauf des Fehlertests wird in Abb. 7.33 dargestellt.

Die Zuverlässigkeit, Leistungs- und Benutzerfreundlichkeit der mobilen Applikation wurde durch eine ausgewählte Testgruppe überprüft. Den Testpersonen wurde hierzu eine Vorabversion der Anwendung ausgeteilt. Ziel des Integrationstests ist die Überprüfung der Funktionen unter normalen Nutzungsbedingungen. Die Ergebnisse des Tests wurden über einen Fragebogen erhoben und anschließend ausgewertet. Die Bestandteile des Fragebogens sowie die Erläuterung der Ergebnisse werden in Abschn. 7.4.2 dargestellt. Nachfolgend wird das Fallbeispiel aufgezeigt und die Resultate des Komponententests beschrieben.

<sup>32</sup> Vgl. Sommerville 2001, S. 447 ff.

## 7.4.1 Beschreibung der Ergebnisse des Komponententests

Innerhalb der nächsten Abschnitte erfolgt die Durchführung und Beschreibung des Komponententests sowie die Erläuterung der erzielten Testergebnisse. Zur Auffindung möglicher Programmfehler wurde im Voraus ein Fallbeispiel generiert, das insbesondere Testroutinen zur Überprüfung der Baufinanzierungs- und Bauetatfunktion beinhaltet. Im Anschluss daran werden die durch das Fallbeispiel identifizierten Softwarefehler beschrieben und deren Lösung vorgestellt.

### 7.4.1.1 Fallbeispiel zur mobilen Baufinanzierungs-Applikation

Das Fallbeispiel handelt von Familie Schmidt. Stefan und Christina Schmidt sind seit zwölf Jahren verheiratet und haben einen zweieinhalbjährigen Sohn. Stefan Schmidt arbeitet als Webdesigner in einem gut geführten IT-Unternehmen. Christina Schmidt arbeitet halbtags als Erzieherin in einer Kindertagesstätte. Familie Schmidt wohnt zurzeit in einer Mietwohnung, welche zwar noch gut ausgestattet und modern eingerichtet ist, aber aufgrund eines weiteren zu erwartenden Nachwuchses auf Dauer etwas zu klein wird. Aus diesem Grund möchte Familie Schmidt ein Haus erwerben.

#### 7.4.1.1.1 Die Funktion Bauetat

**Schritt 1: Einkommen berechnen** Im Menüpunkt „Einkommen eingeben“ der Bauetatfunktion gibt Stefan Schmidt zunächst die Einkommenssituation ein: Er selbst verdient monatlich 2.600,00 €, seine Frau Christina Schmidt monatlich 600,00 €. Urlaubs- und Weihnachtsgeld bleiben jeweils unberücksichtigt. Für den Sohn von Familie Schmidt gibt es 154,00 € Kindergeld. Mit der Erstellung und Betreuung von Homepages verdient Stefan Schmidt nebenberuflich weitere 450,00 € im Monat. Die Funktion „Berechnen“ summiert alle Eingaben und ermittelt ein gesamtes monatliches Einkommen in Höhe von 3.804,00 € (Abb. 7.34).

In einem nächsten Schritt erfolgt die Ermittlung der gesamten monatlichen Ausgaben. Hierunter werden neben den Mietkosten auch Spar- und Versicherungsbeiträge sowie Lebenshaltungskosten berücksichtigt.

**Schritt 2: Ausgaben berechnen** Stefan und Christina Schmidt zahlen eine monatliche Miete von 650,00 €. Des Weiteren besitzen beide einen monatlichen Sparvertrag bei der örtlichen Bank in Höhe von 200,00 €. Für Versicherungen zahlt die Familie Schmidt folgende monatliche Beträge: Haustrat in Höhe von 82,76 €, Haftpflicht in Höhe von 7,81 € sowie Kapitallebensversicherung in Höhe von 34,00 €. Die gesamten monatlichen Versicherungskosten betragen somit 124,57 €. Das gemeinsame Auto von Familie Schmidt wird monatlich mit 70,00 € abbezahlt. Die Haftpflicht für das Auto kostet monatlich 68,37 €. Reparatur- und Benzinkosten werden mit jeweils 130,00 € und 526,98 € veranschlagt. Die KFZ-Steuer beträgt 9,37 €. Die gesamten monatlichen KFZ-Kosten betragen somit 801,73 €. Für die Lebenshaltungskosten werden pauschal 900,00 € veranschlagt. Für sonstige Ausgaben (Ausflüge, Restaurantbesuche ö. ä.) werden monatlich 250,00 € angegeben. Die Funktion „Ausgaben berechnen“ summiert alle Eingaben und berechnet die gesamten monatlichen Ausgaben in Höhe von 2.926,30 € (Abb. 7.35).

**Abb. 7.34** Funktion  
„Einkommen eingeben“ der  
BauFinanz-Applikation

Bauetat berechnen

Einkommen ändern

Ihr Nettoeinkommen pro Monat  
2600.00 EUR

Nettoeinkommen Partner pro Monat  
600.00 EUR

Kindergeld pro Monat  
154.00 EUR

Sonstiges Einkommen pro Monat  
450.00 EUR

Berechnen Abbruch Eingaben löschen

Ausgaben eingeben



**Abb. 7.35** Funktion  
„Ausgaben eingeben“ der  
BauFinanz-Applikation

Bauetat berechnen

Ausgaben ändern

Derzeitige Kaltmiete pro Monat  
650.00 EUR

Sparbeiträge pro Monat  
200.00 EUR

Versicherungsbeiträge pro Monat  
124.57 EUR

KFZ Kosten pro Monat  
801.73 EUR

Lebenshaltungskosten pro Monat  
900.00 EUR

Sonstige Ausgaben pro Monat  
250.00 EUR



**Abb. 7.36** Funktion „Eigenkapital eingeben“ der BauFinanz-Applikation

Eigentyp	Wert	Währung
Barvermögen	19000.00	EUR
Wertpapiere	0.00	EUR
Bausparguthaben	36429.80	EUR
Eigenleistung	0.00	EUR
Sonstiges Eigenkapital	10000.00	EUR

Berechnen    Abbruch    Eingaben löschen

Nachdem die gesamten monatlichen Ausgaben berechnet wurden, ermittelt der Baueatrechner zwei weitere Werte: Zum einen wird ein freier Betrag zur Finanzierung ermittelt, welcher sich aus der Gegenüberstellung der gesamten Einnahmen und Ausgaben ergibt und zum anderen erfolgt die Hochrechnung dieses monatlichen Freibetrags auf einen freien Jahresbetrag. Familie Schmidt stehen somit 877,70 € monatlich (10.532,40 € jährlich) zur Finanzierung zur Verfügung.

Innerhalb des nächsten Menüpunktes erfolgt die Ermittlung und Auswertung des Eigenkapitals. Hierbei werden auch kurzfristig verfügbare Anlagen wie Guthaben auf einem Giro- oder Tagesgeldkonto berücksichtigt.

**Schritt 3: Eigenkapital berechnen** Familie Schmidt verfügt über ein Tagesgeldkonto und hat hierauf bereits 19.000,00 € angespart. Wertpapiere besitzen sie nicht. Sowohl Stefan als auch Christina Schmidt besitzen jeweils ein Bausparguthaben. Insgesamt können sie dadurch über ein gesamtes Bausparguthaben in Höhe von 36.429,80 € verfügen. Da Stefan und Christina Schmidt berufstätig sind, ziehen beide in Betracht, die Bau- und Renovierungsarbeiten am Haus durch Fachkräfte durchführen zu lassen. Im Bereich Eigenleistung wird dadurch keine Angabe getätigt. Die Eltern von Stefan Schmidt wollen bei einem bevorstehenden Hauskauf Stefan und Christina mit 10.000 € unterstützen. Der Bauetarechner summiert alle Eingaben zusammen und berechnet das gesamte Eigenkapital in Höhe von 65.429,80 € (Abb. 7.36).

**Abb. 7.37** Funktion „Finanzierung eingeben“ der BauFinanz-Applikation



Anknüpfend an die Ermittlung des Eigenkapitals erfolgt die Berechnung einer Finanzierung. Die Baufinanzierungs-Applikation ermöglicht hierbei die Berücksichtigung mehrerer Finanzierungskonditionen zur Berechnung maximaler Objektkosten.

**Schritt 4: Finanzierung berechnen** Unter dem Punkt „Finanzierung eingeben“ werden spezielle Darlehen für den Immobilienwert in die Gesamtberechnung mit einbezogen. Maximal können zwei Darlehenssituationen abgefragt werden.

Stefan und Christina Schmidt entscheiden sich, jeweils zwei Kredite bei der örtlichen Bank in Höhe von 40.000,00 € (insgesamt 80.000,00 €) aufzunehmen. Bei den Krediten beträgt der Nominalzins jeweils 4,75 % und die Tilgung beträgt jeweils 1 %. Die Zinsbindung wurde jeweils auf zehn Jahre festgelegt. Aus den eingegebenen Daten wird die monatlich zu zahlende Rate in Höhe von 191,67 € (insgesamt 383,34 €) ermittelt (Abb. 7.37). Aus den errechneten Daten ergeben sich daher maximale Objektkosten in Höhe von 145.429,80 €.

**Schritt 5: Nebenkosten berechnen** Abschließend werden Nebenkosten betrachtet, die optional für weiter zu zahlende Beträge eingetragen werden können, um einen möglichst genauen maximalen Kaufpreis zu ermitteln. In diesem Sinne erwägt Familie Schmidt weitere Nebenkosten abzuschätzen, um den maximalen Kaufpreis der Immobilie genauer

**Abb. 7.38** Funktion „Nebenkosten eingeben“ der BauFinanz-Applikation



bestimmen zu können. Die Nebenkosten beziehen sich auf die bereits ermittelten maximalen Objektkosten. Hierfür geben sie für Bau- und Renovierungskosten 8.000,00 € an. Als Sicherheitszuschlag werden 5 % (7.271,49 €), als Maklercourtage werden 3,5 % (5.090,04 €), für Notar- und Grundbuchkosten werden 1,5 % (2.181,45 €) und für die Grunderwerbssteuer werden 3,5 % (5.090,04 €) geschätzt. Der Bauetabrechner ermittelt aus den eingegebenen Daten zu zahlende Nebenkosten in Höhe von 27.633,02 € (Abb. 7.38). Der angepasste maximale Kaufpreis der Immobilie beträgt nun 173.062,82 €.

#### 7.4.1.1.2 Die Funktion Baufinanzierung

Nachdem Familie Schmidt nun einen Überblick hat, welchen maximalen Kaufpreis sie durch ihre aktuelle finanzielle Situation abdecken können, entscheiden sie sich dafür, die Suche nach einem neuen Haus zu starten. Nach einiger Zeit haben sie bei einem Immobilienmakler ein Haus gefunden, welches ihnen von den Beschreibungen her zusagt. Auch der Preis liegt im Rahmen dessen, was die BauFinanz-Applikation ausgegeben hat (145.429,80 bis 173.062,82 €). Stefan und Christina möchten dennoch wissen, ob sie sich das Haus im Wert von 135.000,00 € wirklich leisten können und benutzen hierfür die Funktion „Baufinanzierung“.

**Schritt 1: Objektkosten berechnen** Im ersten Schritt werden die Erwerbsnebenkosten berechnet. Neben dem Kaufpreis (135.000,00 €) müssen somit noch die Maklercourtage (3,48 %), Notar- und Grundbuchkosten (1,5 %), die Grunderwerbssteuer (3,5 %) sowie ein

**Abb. 7.39** Funktion „Objektkosten eingeben“ der BauFinanz-Applikation



Sicherheitszuschlag (2,00 %) eingegeben werden. Von dem Immobilienmakler hat Familie Schmidt nun exakte Angaben zu den Nebenkosten bzgl. des Immobilienerwerbs erhalten. Die Erschließungskosten entfallen, da es sich bereits um eine fertige Immobilie handelt. Insgesamt belaufen sich die Nebenkosten somit auf 10,48 %. Der Baufinanzierungs-Rechner ermittelt aus den eingegebenen Daten die zu zahlenden Erwerbsnebenkosten in Höhe von 14.148,00 €. Die aktuell zu zahlenden Erwerbsnebenkosten sind somit um 4.757,86 € geringer, als die zuvor angenommenen Nebenkosten. Die Erwerbsnebenkosten werden mit den Objektkosten zu gesamten Objektkosten in Höhe von 149.148,00 € verrechnet (Abb. 7.39).

Innerhalb des nächsten Schrittes erfolgt die Ermittlung und Auswertung des Eigenkapitals. Hierbei ist anzumerken, dass bei der Anschaffung einer neuen Immobilie mindestens 20 bis 25 % des Kaufpreises aus Eigenkapital stammen müssen. Mögliche Darlehen können ansonsten von den Banken verwehrt werden.

**Schritt 2: Eigenkapital berechnen** Die Eigenkapitalsituation von Familie Schmidt wurde bereits bei den Vorplanungen mit dem Bauetatrechner ermittelt. Das hierbei ermittelte gesamte Eigenkapital in Höhe von 65.429,80 € wird in den Baufinanzierungsrechner übertragen. Das gesamte Eigenkapital in Prozent zu den Objektkosten beträgt 43,87 %. Insgesamt besteht ein Finanzierungsbedarf von 83.718,20 € (Abb. 7.40).

**Abb. 7.40** Funktion „Finanzierungsbedarf ausgeben“ der BauFinanz-Applikation



**Schritt 3 und 4: Einkommen und Ausgaben berechnen** Bei den Vorplanungen wurden bereits die monatlichen Einnahmen und Ausgaben der Familie Schmidt berechnet. Die bereits ermittelten Daten werden in den Baufinanzierungsrechner übernommen, jedoch ohne Berücksichtigung der derzeitigen Kaltmiete. Die gesamten monatlichen Ausgaben belaufen sich somit auf 2.276,30 € (2.926,30 € inklusive Kaltmiete). Die Gegenüberstellung der Einnahmen und Ausgaben ergibt einen monatlichen freien Betrag zur Finanzierung in Höhe von 1.527,70 € (Abb. 7.41).

**Schritt 5: Finanzierung berechnen** Da der ermittelte Finanzierungsbetrag über der zuvor angenommenen Finanzierungshöhe liegt, muss Familie Schmidt einen höheren Finanzierungsbetrag bei einem Kreditinstitut aufnehmen. Nach längerer Suche findet Familie Schmidt ein Kreditinstitut mit attraktiven Kreditkonditionen (Kreditbetrag: 90.000,00 €, Nominalzins 4,25 %, Tilgungssatz 1 %, Zinsbindung 20 Jahre), die Stefan Schmidt in den Baufinanzierungsrechner eingibt. Aus den eingegebenen Daten berechnet er die monatlich zu zahlenden Raten in Höhe von 393,75 € (Abb. 7.42).

Das Ergebnis sieht für Familie Schmidt sehr gut aus. Im Vergleich zur jetzigen Kaltmiete entsteht zunächst keine Mehrbelastung aufgrund der Finanzierung. Familie Schmidt kann eine monatliche Entlastung gegenüber der Kaltmiete in Höhe von 256,25 € aufweisen. Des Weiteren bleiben nach dem Kauf der Immobilie monatlich 1.133,95 € zur freien Verfügung übrig.

**Abb. 7.41** Funktion „Freier Betrag ausgeben“ der BauFinanz-Applikation



**Abb. 7.42** Funktion „Mehrbelastung ausgeben“ der BauFinanz-Applikation



Kriterium	Ausprägung
<b>Art:</b>	Fehler
<b>Klasse:</b>	Layout
<b>Titel:</b>	Allgemein
<b>Priorität:</b>	Unwesentlich
<b>Bereich:</b>	Hintergrundbilder
<b>Beschreibung:</b>	Da keine Hintergrundbilder eingebunden waren, konnten einige Angaben nicht oder nur schwer gelesen werden.

**Abb. 7.43** Fehleranalyse mobile Applikation – mangelnde Lesbarkeit der Benutzereingaben

#### 7.4.1.2 Darstellung der Testergebnisse

Die Durchführung des Fallbeispiels ermöglichte die Identifizierung verschiedener Probleme und Verbesserungspotenziale mit dem Umgang der mobilen Applikation. Zur übersichtlichen Darstellung der Ergebnisse der Fehleranalyse werden diese nachfolgend in einem Fehlerkatalog dargestellt. Der Fehlerkatalog beinhaltet die Kategorien Layout, Berechnungen, Logik und Systemtechnik. Jeder Katalogeintrag wird durch eine farbige Markierung in unwesentliche (gelb) und bedeutsame (orange) Fehler oder Verbesserungsvorschläge charakterisiert. Weiterhin erfolgen die Beschreibung des aufgetretenen Fehlers sowie die Erläuterung des Lösungsvorgangs.

##### 7.4.1.2.1 Layout der Applikation

Innerhalb der Analyse des Layouts der Applikation erfolgte die Betrachtung der grafischen Darstellung, der Grammatik und Rechtschreibung sowie sonstiger optischer Auffälligkeiten (Abb. 7.43).

Da innerhalb der Testversion keine Hintergrundbilder eingebunden wurden, konnten die Angaben in den Textfeldern stellenweise nicht identifiziert werden. Die finale Version beinhaltete ein einheitliches Erscheinungsbild mit einem hellblauen Hintergrund, der sich bestens zur Darstellung der Benutzereingaben eignete. Weiterhin konnten einige Rechtschreib- und Grammatikfehler erkannt und verbessert werden, die nachfolgend nicht im Fehlerkatalog aufgenommen wurden. Durch die Neugestaltung der Benutzerflächenstruktur konnten zudem keine sonstigen optischen Auffälligkeiten registriert werden (Abb. 7.44).

##### 7.4.1.2.2 Berechnungen innerhalb der Applikation

Die Eingabe von Centbeträgen innerhalb der Textfelder konnte nicht durch die Eingabe eines Kommas realisiert werden. Grund hierfür ist die in Java verankerte englische Währungsschreibweise mit einem Punkt. Da in App Inventor keine Möglichkeit besteht die Währungszeichen zu ändern, war es nicht möglich das vorliegende Problem zu beheben. In der integrierten Hilfe der mobilen Applikation wird auf die englische Schreibweise hingewiesen. Die Ergebnisse aus der Fallstudie wurden zudem mit manuell durchgeföhrten

Kriterium	Ausprägung
<b>Art:</b>	Fehler
<b>Klasse:</b>	Berechnung
<b>Titel:</b>	Eingabe von Fließkommazahlen
<b>Priorität:</b>	Bedeutsam
<b>Bereich:</b>	Baufinanzierung berechnen; Bauetat berechnen
<b>Beschreibung:</b>	Es ist nicht möglich, Fließkommazahlen mit einem Komma einzutragen. Die Eingabe kann nur in englischer Schreibweise mit Punkt erfolgen.

**Abb. 7.44** Fehleranalyse mobile Applikation – Fehler bei der Dateneingabe

Kriterium	Ausprägung
<b>Art:</b>	Verbesserungsvorschlag
<b>Klasse:</b>	Berechnung
<b>Titel:</b>	Eingabe von Fließkommazahlen
<b>Priorität:</b>	Unwesentlich
<b>Bereich:</b>	Baufinanzierung berechnen; Bauetat berechnen
<b>Beschreibung:</b>	Das Entfernen der eingegebenen Daten nach Abschluss der Ermittlung einer Baufinanzierung oder eines Bauetats ist sehr umständlich. Es muss jeweils die Schaltfläche "Einträge löschen" in den einzelnen Eingabemasken ausgewählt werden.

**Abb. 7.45** Fehleranalyse mobile Applikation – Fehler bei der Eingabe von Fließkommazahlen

Berechnungen überprüft. Der Testlauf zeigte, dass alle in der Applikation enthaltenen Berechnungen fehlerfrei umgesetzt werden konnten.

#### 7.4.1.2.3 Logik der Applikation

Innerhalb der Analyse in Bezug auf die Logik der Applikation wurde zunächst der allgemeine Programmablauf betrachtet. Im Anschluss daran wurde versucht, eine Vereinheitlichung von Begriffen vorzunehmen sowie fehlende sowie überflüssige Konstrukte zu benennen und kenntlich zu machen (Abb. 7.45).

Die Umsetzung des oben beschriebenen Verbesserungsvorschlags wurde mithilfe der in Smartphones eingebauten Bewegungssensoren realisiert, da das Hinzufügen einer zusätzlichen Schaltfläche den Ablauf zur Ermittlung einer Baufinanzierung oder eines Bauetats beeinträchtigt hätte. Eine benutzerfreundliche Alternative stellt hingegen das Schütteln des Smartphones dar, wodurch alle Benutzereingaben in den Textfeldern sowie die daraus ermittelten Zwischenergebnisse entfernt werden. Das versehentliche Entfernen von Benutzerangaben wird durch eine Abfrage aufgefangen und kann somit nicht erfolgen (Abb. 7.46).

Die Fehleranalyse ergab neben dem Auffinden von internen Programmfehlern, dass bisher keine Einblendung der Nutzungsbedingungen erfolgte. Des Weiteren wurde kein Haftungsausschluss für etwaige Fehlberechnungen oder Schäden durch die Ermittlung einer Baufinanzierung angegeben. Das Problem wurde durch den Aufruf einer

Kriterium	Ausprägung
<b>Art:</b>	Fehler
<b>Klasse:</b>	Fehlende Konstrukte
<b>Titel:</b>	Nutzungsbedingungen
<b>Priorität:</b>	Bedeutsam
<b>Bereich:</b>	Applikationsstart
<b>Beschreibung:</b>	Beim Starten der Applikation erfolgte keine Ausgabe der Nutzungsbestimmungen oder eines Haftungsausschlusses.

**Abb. 7.46** Fehleranalyse mobile Applikation – fehlende Nutzungsbedingungen

Informationsmeldung zu Beginn der Applikation gelöst. In dieser Meldung sind die Nutzungsbedingungen sowie der Haftungsausschluss enthalten. Der Benutzer muss diese vor der Nutzung der Applikation akzeptieren, bevor ein Zugriff auf die Optionen der Baufinanzierungsapplikation erfolgen kann. Die Ablehnung der Nutzungsbedingungen führt zur Beendigung der Anwendung. Eine interne Überprüfungsroutine sorgt dafür, dass die Informationsmeldung bei bereits akzeptierten Nutzungsbedingungen nicht mehr angezeigt wird.

Die Ergebnisse des Komponententests sind insgesamt als positiv zu betrachten. Schwerwiegende Fehler konnten nicht identifiziert werden. Im Folgenden werden die Ergebnisse des Integrationstests betrachtet sowie der Fragebogen zur mobilen Applikation vorgestellt und ausgewertet.

#### 7.4.2 Beschreibung der Evaluationsergebnisse

Übersetzt bedeutet das Wort „Evaluation“ erfassen und bewerten. Durch eine Evaluation wird immer eine vorher geleistete Arbeit nach ihrer Effektivität begutachtet und ausgewertet. Formen der Evaluation werden vor allem im Bereich der Forschung und Entwicklung sowie im Bereich der Lehre eingesetzt. Für die Durchführung einer Evaluation können quantitative und qualitative Werkzeuge eingesetzt werden. Zu den quantitativen Methoden der Evaluierung zählt z. B. die Erstellung von Fragebögen. Die Durchführung von mündlichen Rückmeldungen oder telefonischen Interviews zählt zu den qualitativen Instrumenten der Evaluierung. Letzteres benötigt jedoch immer eine schriftliche Protokollierung und Auswertung der Gespräche.<sup>33</sup>

Der Evaluation der mobilen Applikation zur Baufinanzierung kommt eine wichtige Bedeutung für den weiteren Verlauf des Entwicklungsprozesses zu. Für den Fall einer hohen Akzeptanz der Anwendung bei der Testgruppe erfolgt die Beendigung des Entwicklungsprozesses. Werden jedoch fehlerhafte Funktionen oder benutzerunfreundliche Strukturen ermittelt, müssen im Projektverlauf nochmals Rücksprünge in die Implementierungsphase erfolgen und die Applikation nach den Vorschlägen der Testpersonen verbessert

<sup>33</sup> Vgl. Zech 2008, S. 94.

werden. Für die Erhebung der Testdaten wurde ein Fragebogen erstellt, der sich aus Multiple-Choice-Fragen sowie Freitextfragen zusammensetzte. Die detaillierte Auswertung der Ergebnisse wird nach der folgenden Beschreibung des Aufbaus des Fragebogens aufgezeigt.

#### **7.4.2.1 Aufbau des Fragebogens zur mobilen Baufinanzierungs-Applikation**

Der Fragebogen zur mobilen Baufinanzierungs-Applikation ist eine anonyme Umfrage bestehend aus zehn Fragen zur Benutzerfreundlichkeit und Anwendbarkeit der mobilen Baufinanzierungs-Applikation. In der Zeit vom 4. bis 17. Juli haben insgesamt 25 Personen, davon 13 weibliche und zwölf männliche, im Alter zwischen 21 und 55 Jahren an der Umfrage teilgenommen. Neben dem Alter und Geschlecht wurden weitere demografische Daten über die Testpersonen erhoben, nämlich die Technikaffinität, der Grad der Nutzung mobiler Anwendungen sowie die Marke des aktuellen Mobiltelefons. Insgesamt gaben elf Testpersonen an, dass sie eine niedrige Technikaffinität besitzen. Jeweils sieben Personen gaben an, eine mittlere sowie eine hohe Technikaffinität aufweisen zu können. Die Frage, wie oft mobile Anwendungen durch die Testpersonen benutzt werden, beantworteten 14 Personen mit täglich, fünf Personen mit mehrmals die Woche und sechs Personen mit selten. Dies bildet eine gute Voraussetzung für den Integrationstest der Applikation, da sich die Testgruppe nicht nur aus professionellen Anwendern, sondern auch aus Laien zusammensetzt.

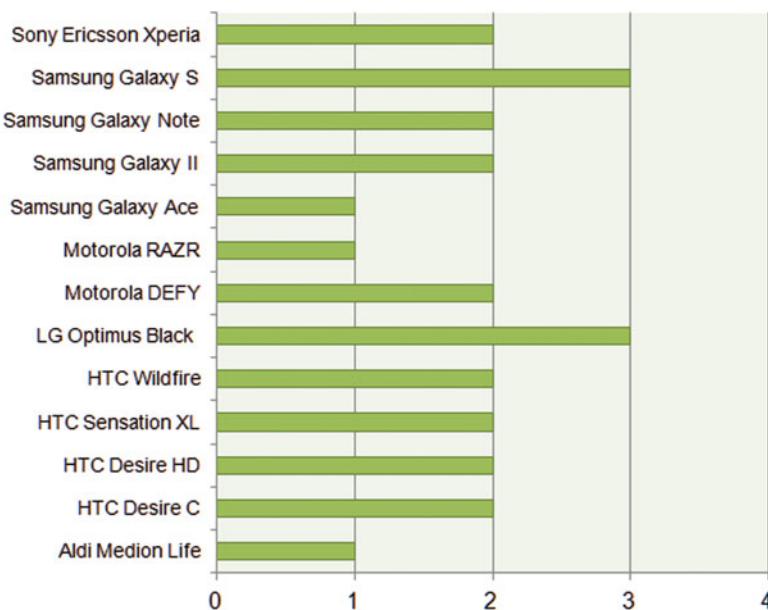
Für den Test der Applikation wurden Mobiltelefone verschiedener Hersteller mit abweichender Hardware eingesetzt. Dadurch konnten Informationen über das Layout sowie die Benutzerfreundlichkeit der Applikation auf unterschiedlichen Bildschirmgrößen erhoben werden. Die zur Durchführung des Softwaretests verwendeten Mobiltelefone lassen sich aus Abb. 7.47 entnehmen. Nachfolgend wird die Auswertung der Umfrage vorgenommen und die Ergebnisse erläutert.

#### **7.4.2.2 Auswertung des Fragebogens zur mobilen Baufinanzierungs-Applikation**

Für die Beurteilung des Fragebogens wurden alle 25 abgegebenen Fragebögen ausgewertet. Die von den Teilnehmern gegebenen Antworten wurden in einer Excel-Datei aufbereitet und ausgewertet. Nachfolgend werden die Ergebnisse der Umfrage vorgestellt und die Gründe für das jeweilige Ergebnis kurz kommentiert.

1. Frage: „*Hatten Sie Probleme die BauFinanz-Applikation auf Ihrem Smartphone zu installieren?*“

Bei der Beantwortung der Frage gaben 20 Teilnehmer an, keine Probleme bei der Installation der Applikation zu haben. Ein kleiner Teil der Testpersonen gab Probleme beim Download der Applikation (2) sowie beim Öffnen der Applikation vor der Installation (3) an. Die Fehler sind zurückzuführen auf die Verteilung einer fehlerhaften Applikationsversion. Nachdem die aktuellste Version für die fünf Teilnehmer zur Verfügung stand, konnte die Applikation auf allen Testgeräten problemlos installiert werden. Die



**Abb. 7.47** Eingesetzte Mobiltelefone beim Integrationstest

nachfolgenden Antworten des Fragebogens beziehen sich alle auf die letzte ausgegebene Testversion der Applikation.

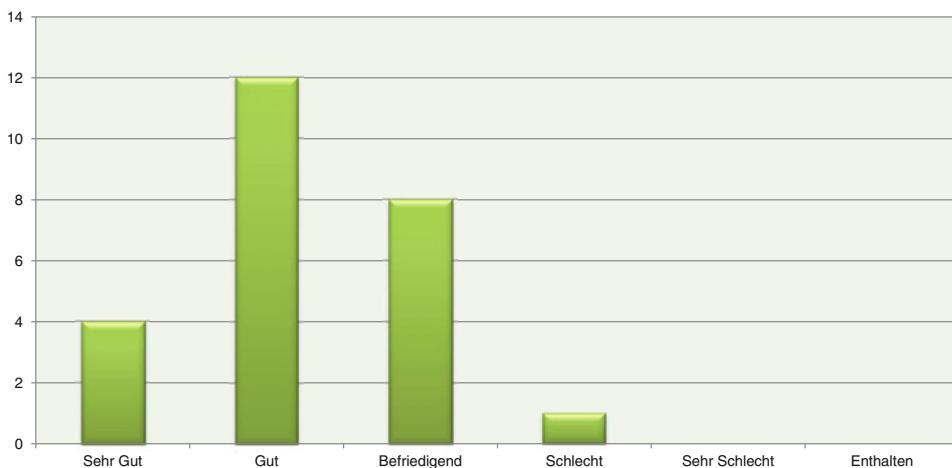
- Frage: „Hatten Sie Probleme beim Start der BauFinanz-Applikation?“

Diese Frage beantworteten alle Testpersonen mit Nein. Die Applikation lässt sich somit auf einer Vielzahl unterschiedlicher Android-Endgeräte installieren und anwenden. Die Aussagen der Testgruppen bestätigen weiterhin, dass eine hohe Anzahl Android-Nutzer über den Google-Play-Markt erreicht werden kann.

- Frage: „Wie ist der logische Aufbau/die Struktur der BauFinanz-App zu bewerten?“

Von den befragten Testteilnehmern empfanden elf Personen die Struktur und den logischen Aufbau der Applikation als gut. Fünf Personen gaben an, die Darstellungsweise als sehr gut zu empfinden, neun Personen fanden den Aufbau und die Struktur eher befriedigend. Die befriedigenden Bewertungen sind auf Testpersonen zurückzuführen, welche Smartphones mit kleiner bis mittelgroßer Bildschirmgröße zum Testzeitpunkt zur Verfügung hatten. Zu diesen Mobiltelefonen gehören bspw. das Medion Live von Aldi oder das Wildfire von HTC. Vor diesem Hintergrund eignet sich die mobile Baufinanzierungsapplikation eher für Smartphones mit großen Displays. Für die optimale Nutzung der Applikation auf kleineren Smartphones wurde die in Abb. 7.23 beschriebene Skalierung der Steuerelemente nochmals angepasst. Die Anpassung konnte jedoch nur über den Android-Emulator und nicht mehr an einem der Testgeräte überprüft werden.

- Frage: „Wie empfanden Sie die Darstellung und Erklärung der Inhalte der BauFinanz-App?“ Die überwiegende Mehrheit der Testpersonen (12) gab an, dass die Darstellung



**Abb. 7.48** Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der vierten Frage

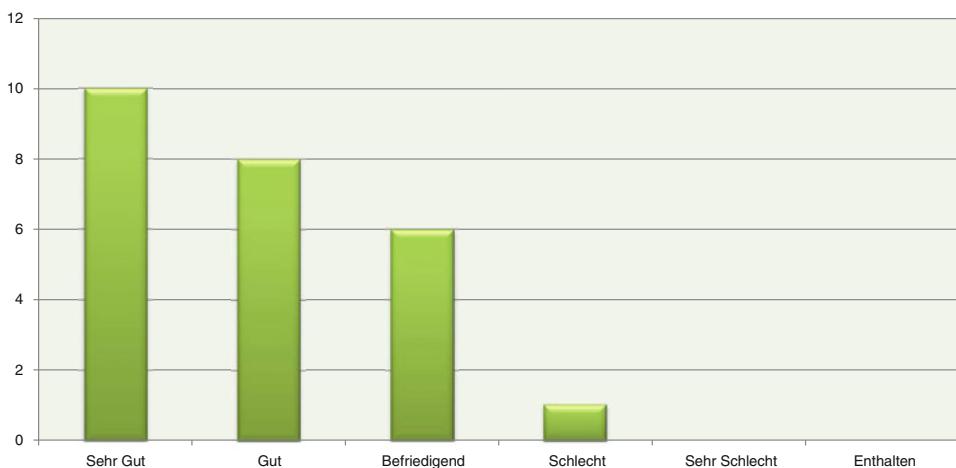
und Erklärung der Inhalte gut umgesetzt wurde. Insgesamt vier Personen finden die Darstellung als sehr gut, acht Personen als befriedigend. Eine Testperson gab an, dass die Erläuterung und Darstellung der Inhalte schlecht umgesetzt wurde. Die befriedigend bis schlecht ausfallende Bewertung in Bezug auf die Inhalte der Applikation war zum Testzeitpunkt auf eine mangelnde Erläuterung der benötigten Benutzereingaben zurückzuführen. Aufgrund dessen erfolgte nach dem Integrationstest eine Überarbeitung der Hilfetexte zur Baufinanzierung und zum Bauetat (Abb. 7.48).

5. Frage: „*Waren Sie mit der Nutzung der vorhandenen Funktionen zufrieden?*“

Die Auswertung der fünften Frage ergab, dass 19 Personen mit der Nutzung zufrieden waren. Sechs Personen bemängelten, dass die ermittelten Daten nicht gespeichert werden können oder zur weiteren Verarbeitung in einer anderen digitalen Form zur Verfügung stehen würden. Dieser Verbesserungsvorschlag wurde nach Abschluss der Testphase umgesetzt. In der finalen Version der Applikation ist es nunmehr möglich, die benutzerspezifischen Daten per Email an eine gewünschte Adresse zu versenden. Die Daten können bspw. an einen stationären PC gesendet und anschließend ausgedruckt werden.

6. Frage: „*Wie ist die Bedienbarkeit der BauFinanz-App über den Touchscreen zu bewerten?*“

Für die Beantwortung der Frage standen wiederum mehrere Antwortmöglichkeiten zur Verfügung. Zehn Teilnehmer empfanden die Bedienung über den Touchscreen als sehr gut, acht Personen als gut. Weitere sechs Personen fanden die Bedienung befriedigend und eine Person gab eine schlechte Bewertung ab. Bedingt durch die automatische Skalierung der Steuerelemente erfolgte bei der Nutzung kleinerer Smartphones eine Reduzierung der Größe von Schaltflächen, Textfeldern und Beschriftungen. Die bereits angesprochene Optimierung der Skalierungsfunktion sollte das vorliegende Problem beheben. Die Überprüfung der Funktion konnte jedoch ebenfalls nur auf dem Android-Emulator durchgeführt werden (Abb. 7.49).



**Abb. 7.49** Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der sechsten Frage

7. Frage: „*Entstanden bei der Nutzung der BauFinanz-App hohe Ladezeiten?*“

Bedingt durch die Nutzung von Smartphone-Modellen mit einer veralteten Hardwa-reausstattung gaben drei Personen bei der Nutzung der mobilen Applikation längere Ladezeiten an. Die restlichen 22 Testpersonen hatten während der Nutzung der Appli-kation keine Probleme mit hohen Ladezeiten. Eine Verbesserung der Laufzeit durch die Neugestaltung der Struktur ist somit auch auf den meisten Testgeräten erkennbar.

8. Frage: „*Bietet die BauFinanz-Applikation Ihrer Meinung nach einen Mehrwert?*“

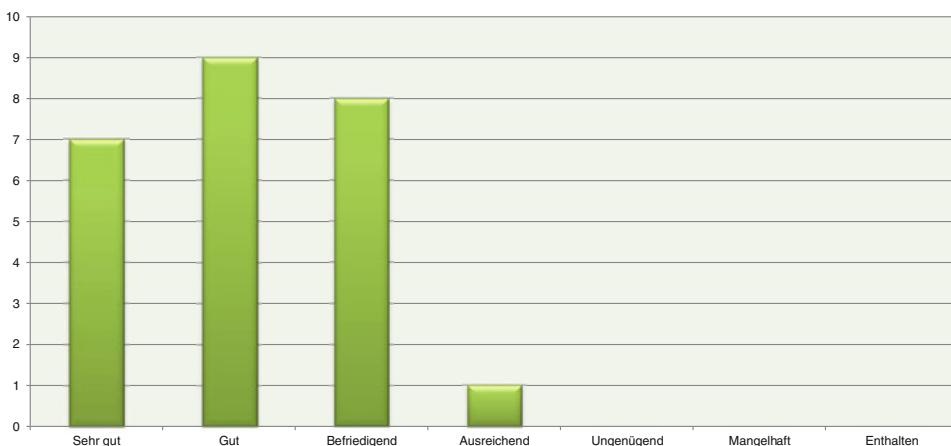
Von den befragten Teilnehmern empfanden 17 Personen einen Mehrwert durch die Nutzung der mobilen Baufinanzierung im Vergleich zu einer klassischen Baufinanzierungsberatung. Laut Angaben der Testgruppe besteht der Vorteil der Applikation hauptsächlich in der schnellen Ermittlung einer Baufinanzierung von unterwegs aus. Acht Personen würden eine klassische Beratungsdienstleistung bevorzugen. Als Grund wird hier die fehlende Ansprechperson für anfallende Fragen zur Baufinanzierung genannt.

9. Frage: „*Sollte die BauFinanz-Applikation zur Unterstützung klassischer Immobilienfinanzierungen eingesetzt werden?*“

Die Auswertung der neunten Frage ergab, dass insgesamt 15 Personen für die Aufnahme einer Immobilienfinanzierung die Ergebnisse der mobilen Applikati-on als Grundlage hinzunehmen würden. Zehn Personen würden die Ergebnisse der Baufinanzierungs-Applikation nicht einsetzen. Hauptsächlich wurde als Grund die Ermittlung einer Baufinanzierung durch einen professionellen Finanzberater angegeben.

10. Frage: „*Bitte geben Sie der BauFinanz-Applikation eine Schulnote.*“

Zu Beantwortung der Frage mussten die Testteilnehmer eine Note aus einer vorgegebe-nen Notenskala angeben. Die Frage wurde insgesamt siebenmal mit sehr gut, neunmal



**Abb. 7.50** Fragebogen zur Baufinanzierungs-Applikation – Ergebnis der zehnten Frage

mit gut, achtmal mit befriedigend und einmal mit ausreichend beantwortet. Der Notendurchschnitt der mobilen Applikation beträgt demnach 2,12. Die Applikation wurde somit von der Testgruppe als gut bewertet (Abb. 7.50).

Der Integrationstest ist insgesamt positiv ausgefallen. Ein Großteil ist mit der Struktur sowie dem Leistungs- und Funktionsumfang zufrieden. Die während des Tests identifizierten Fehler und Verbesserungsvorschläge wurden unmittelbar nach dem Abschluss der Testphase verbessert oder nach den Vorstellungen der Testpersonen umgesetzt.

#### 7.4.2.3 Zusammenfassung

Zu Beginn der Evaluierungsphase erfolgte zunächst die Beschreibung der allgemeinen Vorgehensweise. In diesem Zusammenhang wurden zunächst unterschiedliche Softwaretests zur Analyse und Bewertung von Softwareeigenschaften voneinander abgegrenzt. Ergebnis dieser Abgrenzung bildete die Erläuterung des anstehenden internen Komponententests und des durch eine externe Testgruppe durchgeföhrten Integrationstests.

Im Anschluss an die Beschreibung der Vorgehensweise wurde abschließend der Ablauf des Komponententests vorgestellt und erläutert. Anhand eines ausgearbeiteten Fallbeispiels erfolgte die Überprüfung der einzelnen Funktionen der Baufinanzierungs-Applikation. Die hierbei identifizierten Fehler oder Verbesserungsvorschläge wurden in einen Fehlerkatalog aufgenommen und nach unterschiedlicher Priorität sortiert. Die Darstellung und Umsetzung des Lösungsvorgangs der einzelnen Fehler und Verbesserungsvorschläge bildete den Abschluss des Komponententests. Die Durchführung des Integrationstests wurde durch einen auf die Baufinanzierungs-Applikation zugeschnittenen Fragebogen unterstützt. Der Fragebogen wurde zusammen mit der Testversion der mobilen Anwendung an eine Testgruppe mit einer Größe von 25 Personen ausgehändigt. Die durch die Testpersonen zusätzlich registrierten Probleme oder Verbesserungsvorschläge wurden im Anschluss an den Integrationstest umgesetzt.

## 7.5 Fazit und Ausblick

Innerhalb der Planungs- und Analysephase wurden zunächst aktuelle Herausforderungen im Markt für private Baufinanzierungen betrachtet. Ergebnis dieser Betrachtung bildete die Visualisierung zu überbrückender Hürden in Bezug auf Kunden, Prozesse, Vertriebsweg und den Wettbewerb. Darauf aufbauend erfolgte die Analyse gegenwärtig am Markt vorhandener Konkurrenzprodukte. Insgesamt wurden drei Applikationen zur Baufinanzierung identifiziert und bewertet. Im Mittelpunkt der Analysephase stand die Auswahl geeigneter Instrumente zur Applikationsentwicklung. Die Auswahl eines Vorgehensmodells, eines mobilen Betriebssystems sowie einer Entwicklungsumgebung erfolgte unter Berücksichtigung mehrerer Bewertungsmatrizen sowie unter Zuhilfenahme eines vorgegebenen Punktesystems. Als Vorgehensmodell wurde das UCAN-Modell ausgewählt. Das Betriebssystem Android stellte im Vergleich zu den anderen mobilen Betriebssystemen die beste Alternative dar. Die Applikation sollte weiterhin mit der App-Inventor-Entwicklungsumgebung verwirklicht werden. Die Ausarbeitung eines Projektplans zur Anwendungsentwicklung stellte das Gesamtergebnis der Planungs- und Analysephase dar. Neben der Beschreibung der einzelnen Projektphasen wurden weiterhin Termine und Meilensteine für die Fertigstellung von Zwischenergebnissen festgelegt.

Ziel der Entwurfsphase ist die Erhebung realistischer und umsetzbarer Anforderungen an eine Software. Vor diesem Hintergrund wurden im Rahmen der Anforderungserhebung zunächst eindeutige Ziele, welche durch die mobile Applikation verwirklicht werden sollten, bestimmt. Darauf folgend wurden die jeweiligen Anforderungen spezifiziert und in Muss-, Kann- und Ausschlusskriterien unterteilt. Abschluss der Anforderungserhebung bildete die Darstellung eines ersten Entwurfs des Programmablaufplans. Die Umsetzung der Anforderung erfolgte nach den Richtlinien des UCAN-Modells in mehreren Schritten. Zunächst wurde die eigentliche Anwendungsstruktur im Design-Editor des App Inventors angelegt und getestet. In einem weiteren Schritt wurden die im Blocks-Editor entwickelten Funktionen und Methoden den zur Realisierung der Anwendungsstruktur verwendeten Steuerelementen zugewiesen. Die Implementierungsphase endete mit der Beschreibung aufgetretener Probleme bei der Applikationsentwicklung. Durch die vom UCAN-Modell vorgeschriebene permanente Evaluierung und Validierung der mobilen Applikation konnten Komplikationen in Bezug auf den flüssigen Ablauf der Applikation registriert werden. Die Analyse der vorliegenden Problemsituation ergab, dass hohe Ladezeiten auf die schlechte Struktur der Benutzeroberflächen zurückzuführen sind. Die Neugestaltung der Struktur führte zu einer Verbesserung des Programmablaufs. Die Verbesserung wurde durch die Berechnung der Applikationskomplexität mittels der Metriken nach Halstead bestätigt.

Der Entwicklungsprozess endete mit der Evaluierung der mobilen Applikation. Hierzu wurden einerseits ein interner Komponenten- sowie ein externer Integrationstest durchgeführt. Im Rahmen des Komponententests wurden mithilfe eines speziell auf die Problemsituation ausgerichteten Fallbeispiels die Funktionen zur Ermittlung einer Baufinanzierung und eines Bauetats überprüft. Aufgetretene Fehler und Verbesserungen

rungsvorschläge wurden dokumentiert und im Anschluss an den Test umgesetzt. Der Integrationstest wurde durch eine nicht am Entwicklungsprozess beteiligte Testgruppe durchgeführt. Der Gruppe wurde zur Ausführung des Tests die finale Applikation zur Baufinanzierung zur Verfügung gestellt. Die Erfahrungen mit dem Umgang der Applikation wurden durch einen ausgeteilten Fragebogen erhoben und die ausgewerteten Ergebnisse erläutert. Die Evaluierungsphase endet mit einem Leistungsvergleich gegenüber den zuvor ermittelten Konkurrenten sowie der Aufstellung einer Portfoliomatrix zur Positionierung der Baufinanzierungs-Applikation im Online-Markt.

Auch zukünftig wird die private Baufinanzierung ein Produkt mit hohem Kundenbindungspotenzial darstellen. In Abhängigkeit vom Produktdesign und der Bearbeitung von Kreditaufträgen werden sich jedoch bezüglich der im Geschäftsfeld der Baufinanzierung generierten Erträge größere Unterschiede herauskristallisieren. Große Herausforderungen für alle Finanzdienstleister liegen darin, den Ansprüchen des Vertriebs gerecht zu werden. Insbesondere ist die konsequente Strukturierung und Einhaltung innerbetrieblicher Prozesse maßgeblich für einen erfolgreichen Vertrieb in der privaten Baufinanzierung. Um angesichts der gerade beschriebenen Anforderungen des Marktes entsprechend reagieren zu können, müssen Kreditinstitute flexible und anpassungsfähige Produkte und Prozesse zur Verfügung stellen. Intelligente IT-Lösungen können hierbei die fehlende Lücke zwischen Finanzdienstleister und Kunden schließen und so die aktive Kundenbindung fördern. Mobile Applikationen weisen alle Bedingungen für eine solche IT-Lösung auf und rücken dadurch immer mehr in das Interesse vieler Kreditinstitute. Durch den Einsatz mobiler Anwendungen können Finanzdienstleister Prozesszeiten verkürzen, Kosten senken, die Qualität verbessern sowie die Produktivität steigern.

Die Entwicklung einer Android-Applikation ist, wie sich gezeigt hat, mit vielen Herausforderungen verbunden und verlangt nach der Berücksichtigung vieler verschiedener Methoden zur Planung und Umsetzung des Entwicklungsprozesses. Die App-Inventor-Entwicklungsumgebung bietet zwar eine sehr solide Grundlage für die Entwicklung mobiler Applikationen, dennoch ist die Durchführung des gesamten Entwicklungsprozesses keine einfache Angelegenheit und setzt hohe Ansprüche an die Softwareentwickler. Dennoch ist ein scheinbar unaufhaltsamer Aufwärtstrend in Bezug auf den Vertrieb und die Vermarktung mobiler Applikationen und Endgeräte sowie die damit verbundene steigende Nutzung des mobilen Internets erkennbar. Nach Meinung des Autors werden jedoch in naher Zukunft die Begriffe „mobiles Internet“ und „mobile Endgeräte“ nicht mehr voneinander unterschieden. Vielmehr erfolgt die Betrachtung verschiedener Endgeräteeigenschaften, bspw. die Bildschirmgröße oder die Prozessorleistung, und unterschiedlicher Nutzungsszenarien, bspw. von zu Hause oder unterwegs. Immer häufiger werden mobile Endgeräte für die schnelle Informationsrecherche stationären Rechnern vorgezogen. Die fortschreitende Entwicklung moderner Smartphones und Tablet-PCs wird diesen Effekt noch mehr verstärken.

## Literatur

- Aichele, C.: Intelligentes Projektmanagement. W. Kohlhammer, Stuttgart (2006)
- Apple Inc.: Apple developer programs. <https://developer.apple.com/programs>. (2012b) Zugriffen 9. May 2013
- Apple Inc.: Über 500.000 Apps. Fürs Arbeiten, Spielen und alles andere. <http://www.apple.com/de/iphone/from-the-app-store>. (2012a) Zugriffen 9. May 2013
- Bunse, C., Knethen von, A.: Vorgehensmodelle kompakt, 2. Aufl. Spektrum Akademischer, Heidelberg, (2008)
- Capgemini: Studie IT-Trends. Business-IT-Alignment sichert die Zukunft. Capgemini, Berlin (2012)
- Engstler, M., Praeg, C.-P., Vocke, C.: Zusammenfassung zur Trendstudie Bank & Zukunft 2007. In: Spath, D. (Hrsg.) Bank & Zukunft 2007. Mit Prozessexzellenz und Vertriebsinnovationen die Bank der Zukunft gestaltet. Fraunhofer Institut für Arbeitswirtschaft und Organisation, Stuttgart (2007)
- Gartner Inc.: Gartner says worldwide smartphone sales soared in fourth quarter of 2011 with 47 % growth, Egham. 15 Februar. <http://www.gartner.com/it/page.jsp?id=1924314> (2012). Zugriffen 9. May 2013
- Gernert, C.: Agiles Projektmanagement. Risikogesteuerte Softwareentwicklung. Carl Hanser, München (2003)
- Google Android Developer Portal: Entwickler-Registrierung. <http://support.google.com/googleplay/android-developer/bin/answer.py?hl=de&an-swer=113468> (2012). Zugriffen 9. May 2013
- Kloss, J. H.: Android-Apps. Mobile Anwendungen entwickeln mit App Inventor. Markt + Technik, München (2011)
- Krcmar, H.: Informationsmanagement, 5. Aufl. Springer, Berlin (2010)
- Microsoft Corp.: 50.000 Apps im Windows Phone Marketplace, 9.01.2012a. [http://www.microsoft.com/germany/msdn/aktuell/news/show.mspx?id=msdn\\_de\\_45321](http://www.microsoft.com/germany/msdn/aktuell/news/show.mspx?id=msdn_de_45321) (2012a). Zugriffen 9. May 2013
- Microsoft Corp.: Windows Phone, dein Weg zur eigenen App, <http://www.microsoft.com/germany/msdn/academic/windows-phone/registriere-dich-als-entwickler.aspx> (2012b). Zugriffen 9. May 2013
- Moormann, J., Schaefer, A.: Smartphone-Apps im Banking. Internationale Untersuchung des State of the Art von Smartphone-Apps im Bankgeschäft. IM – Fachzeitschrift für Information Management und Consulting 27(1), S. 46–53 (2012)
- Müller, M. B.: Backoffice-Prozesse als Teil eines erfolgreichen Vertriebs gestalten. Bank und Markt 40(8), 33–35 (2011)
- My Private Banking GmbH: Mobile Apps for Banking. Benchmarking – Best Practices – Strategies, April 2013, Kreuzlingen (2013)
- Parbel, M.: Android und iOS beherrschen die Smartphone-Welt, <http://www.heise.de/resale/meldung/Android-und-iOS-beherrschen-die-Smartphone-Welt-1584381.html> (2012). Zugriffen 9. May 2013
- Pohl, K.: Requirements Engineering, Grundlagen, Prinzipien, Techniken, 2. Aufl. dpunkt, Heidelberg (2008)
- Schilling, K., Reuter, C.: Vertriebsinnovationen: Mobile Banking kommt in Schwung. die Bank, Zeitschrift für Bankpolitik und Praxis, 4, Berlin, S. 34–37 (2010)
- Schmidt, G.: Informationsmanagement. Modelle, Methoden, Techniken, 2. Aufl., Springer, Berlin (1999)
- Sommerville, I.: Software Engineering, 6. Aufl. Pearson Studium, München (2001)
- Zech, R.: Handbuch Qualität in der Weiterbildung. Weiterbildung und Qualifikation. Beltz Taschenbuch, Weinheim (2008)

Christian Radny

*Konzeption und Entwicklung einer Anwendung im  
Mobile-Devices-Bereich am Beispiel einer PowerPoint-Presenter-App  
für Android.*

---

## Zusammenfassung

Immer wieder kommt es vor, dass Präsentationen über die Tastatur des Präsentationscomputers bedient werden müssen, weil kein Presenter verfügbar ist. Da Smartphones mittlerweile Alltagsgegenstände geworden sind, liegt die Überlegung nahe, damit die Präsentation zu steuern. In diesem Kapitel geht es um die Entwicklung und Konzeption einer Presenter-Anwendung für PowerPoint, welche für das mobile Betriebssystem Android umgesetzt werden soll. Der Leser soll die Softwareentwicklung unter Android verstehen und den Aufbau der entwickelten Applikation nachvollziehen können.

---

## 8.1 Einleitung

„Die IDC-Marktanalyse für 2012 zeigt ein weltweites Wachstum der Tablet-Neuverkäufe von fast 80 % gegenüber dem Vorjahr. Smartphones legten um mehr als 45 % zu, während PCs und Notebooks um 4,1 % bzw. 3,4 % nachgaben. Insgesamt ist der Markt für ‚Smart Connected Devices‘ um 29 % gewachsen.“<sup>1</sup> Durch dieses Wachstum sind Smartphones zum

---

<sup>1</sup> Doctronic (2013).

Alltagsgegenstand geworden und infolgedessen ist ein großer Markt für Anwendungen im mobilen Bereich entstanden.

In diesem Kapitel geht es um die Entwicklung und Konzeption einer Presenter-Anwendung für PowerPoint, welche für das mobile Betriebssystem Android umgesetzt werden soll. Der Leser soll die Softwareentwicklung unter Android verstehen und den Aufbau der entwickelten Applikation nachvollziehen können.

Zunächst werden einige besondere Eigenschaften, welche bei der Entwicklung für mobile Geräte beachtet werden sollten, aufgezeigt.

Im Folgenden wird näher auf Android im Speziellen eingegangen. Hierzu werden die wichtigsten Komponenten in Android kurz erklärt und im Anschluss Entwicklungsumgebungen vorgestellt, mit deren Hilfe die Softwareentwicklung unter Android erleichtert werden kann. Die nächsten Punkte beschäftigen sich damit, wie ein Konzept für eine mobile Anwendung erstellt wird und wie die darin erarbeiteten Funktionen umgesetzt werden können.

Anschließend wird die Presenter-Anwendung realisiert. Zunächst wird mithilfe eines passenden Vorgehensmodells dafür ein Konzept entwickelt. Hiernach wird die Realisierung mit Codebeispielen erläutert. Dabei gibt es folgende Schwerpunkte: die Verbindung zwischen mobilem Gerät und PC, das PowerPoint-Dateiformat und die grafische Benutzeroberfläche. Hier werden verschiedene Möglichkeiten vorgestellt, wie bestimmte Funktionen umgesetzt werden könnten und anschließend, unter Beachtung von Gesichtspunkten wie Schwierigkeitsgrad der Realisierung, Ressourcenverbrauch, Kompatibilität etc., die passende Lösung für dieses Projekt ermittelt.

---

## 8.2 Besonderheiten der Softwareentwicklung für mobile Geräte

Die Entwicklung von Applikationen im mobilen Bereich unterscheidet sich zum Teil von der Softwareentwicklung für PC und Web. Zum einen gibt es gewisse Einschränkungen durch die Größe der Anzeige, die Leistungsfähigkeit der Geräte und die inkonsistente Verfügbarkeit der Internetverbindung. Da die Anwendungen unterwegs genutzt werden können, kann es vorkommen, dass ein Nutzer seine Aufmerksamkeit nur teilweise der Anwendung zur Verfügung stellt. Generell gibt es durch die Mobilität einige Nutzungsszenarien, die bedacht werden sollten. Beispielsweise ist die Funktion, die Musikwiedergabe durch Schütteln des Geräts zu unterbrechen, beim Joggen eventuell hinderlich. Der Entwickler kann jedoch durch den sinnvollen Einsatz der vielen Sensoren, die in modernen mobilen Geräten zur Verfügung stehen, verschiedene Anwendungsfälle erkennen und somit erheblich zur Usability beitragen.<sup>2</sup>

---

<sup>2</sup> Vgl. Strang und Lichtenstein (2012, S. 420).

Besonders im Hinblick auf die Usability gibt es im mobilen Bereich erhöhte Ansprüche an die Anwendungen. Als Maß für die Benutzerfreundlichkeit können folgende Kriterien dienen:

- „Erlernbarkeit – Wie leicht ist es für einen Nutzer bei der erstmaligen Nutzung der App, mit der App einfache Aufgaben zu erfüllen?
- Effizienz – Wenn der Umgang mit der App grundsätzlich klar ist, wie schnell kann der Anwender bestimmte Aufgaben damit erfüllen?
- Einprägsamkeit – Wenn der Nutzer nach einer gewissen Zeit ohne Nutzung die App erstmals wieder nutzt, wie schnell ist er wieder mit der App und ihren Funktionen vertraut?
- Fehler – Wie viele Fehler macht ein Nutzer, wie schwer sind diese Fehler, und wie leicht kann er eine Lösung zur Behebung des Problems finden?
- Zufriedenheit – Wie zufriedenstellend ist die Nutzung der App?“<sup>3</sup>

Diese fünf Punkte sollten vor der Veröffentlichung einer mobilen Anwendung genauestens analysiert werden. Denn wenn die ersten Nutzer wegen schlechter Usability unzufrieden sind und der Anwendung schlechte Bewertungen geben, wird es schwer, Erfolg zu haben, auch wenn die Anwendung vom Konzept her gut ist.

Nach Rohles spielen bei der Bewertung einer mobilen Anwendung folgende Faktoren eine Rolle:

- „Gebrauchstauglichkeit bzw. Usability
- wahrgenommene Nützlichkeit,
- Unterhaltungspotenzial,
- sozialer Einfluss anderer und der Medien“<sup>4</sup>

Weiterhin muss im Vorfeld festgelegt werden, ob eine native Applikation oder eine Web-App entwickelt werden soll. Eine weitere Alternative ist die Entwicklung einer hybriden App. „Bei ihnen handelt es sich im Kern um Web-Apps, die auf dem Endgerät allerdings nicht in einem Browser bedient werden, sondern in den Rahmen einer nativen App eingebettet sind. In der Tat ist es auf den ersten Blick für einen Laien nicht ersichtlich, ob eine Anwendung nativ oder als hybride App entwickelt wurde, zumal hybride wie native Apps über die jeweiligen App-Stores der Betriebssystemhersteller verteilt werden.“<sup>5</sup> Mehr Informationen zu diesen Alternativen gibt es in Abschn. 8.3.4 zu lesen.

Egal welcher Ansatz ausgewählt wird, muss die Applikation auf einer Vielzahl unterschiedlicher Geräte getestet werden, vor allem wenn man für den Android-Markt

<sup>3</sup> Vgl. Strang und Lichtenstein (2012, S. 427).

<sup>4</sup> Rohles (2012).

<sup>5</sup> Rühl und Schenkel (2012).

entwickelt. „Während Apple als alleiniger Hersteller von iOS-Geräten ein vergleichsweise kleines, konsistentes Sortiment auf den Markt gebracht hat, tummeln sich im Android-Segment zahlreiche Gerätehersteller, die mit unterschiedlicher Hardware und Geräteausstattung den Markt für Smartphones und Tablets beliefern. Die Unterschiede erstrecken sich nicht nur auf die Hardware der Geräte, sondern umfassen auch die Android-Versionen und speziellen Anpassungen der Benutzeroberfläche durch die Gerätehersteller.“<sup>6</sup> Um auf möglichst vielen Endgeräten eine vernünftig einsetzbare Applikation zu entwickeln, muss diese so flexibel wie möglich gestaltet werden.<sup>7</sup>

Auch die Zielgruppe für Anwendungen im mobilen Bereich sollte genauer betrachtet werden. Diese besteht zum Großteil aus relativ jungen, gebildeten, technikaffinen Menschen.<sup>8</sup>

---

### 8.3 Mobile Betriebssysteme

► „Ein **Betriebssystem** ist die Softwaregrundlage eines jeden Rechners, welche selbstständig alle fundamentalen Funktionen der Rechenprozesse sicherstellt und deren fehlerfreie Ausführung gewährleistet und überwacht. [...] Für mobile Endgeräte gibt es eigens entwickelte Betriebssysteme.“<sup>9</sup>

Sie unterscheiden sich vor allem darin zu den Betriebssystemen herkömmlicher Rechner, dass sie auf die spezielle Hardware und Eingabemethoden mobiler Geräte ausgelegt sind und ein aufwendiges Energiemanagement unterstützen, um die Akkulaufzeiten zu maximieren. Außerdem ist es durch die begrenzten Speicherkapazitäten notwendig, die Größe der Betriebssysteme auf ein Minimum zu reduzieren (Abb. 8.1).<sup>10</sup>

In diesem Kapitel geht es vor allem um das Betriebssystem Android. Jedoch sollen an dieser Stelle auch kurz andere erfolgreiche mobile Betriebssysteme vorgestellt werden.

#### 8.3.1 iOS

Zunächst gibt es einen kurzen Überblick zur Entstehungsgeschichte dieses Betriebssystems. Den Namen iOS trägt das System seit 2010. Zuvor wurde der Name iPhone OS benutzt. Das System wurde zusammen mit dem iPhone im Juni 2007 eingeführt und entstand

---

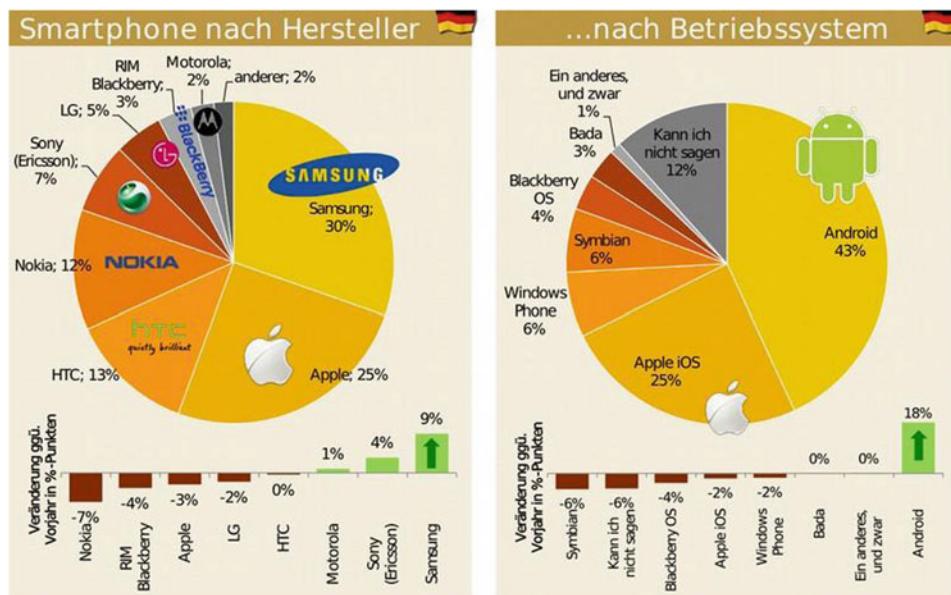
<sup>6</sup> Rühl und Schenkel (2012a).

<sup>7</sup> Vgl. Rühl und Schenkel (2012).

<sup>8</sup> Vgl. Rohles (2012).

<sup>9</sup> Bundschuh (2011).

<sup>10</sup> Bundschuh (2011).



**Abb. 8.1** Smartphone Marktanteile. (Goldmedia Custom Research GmbH 2012)

als mobile, an die besondere Touchscreenbedienung angepasste Version von OS X. Die Möglichkeit, Applikationen von Drittanbietern zu installieren, bekam das Betriebssystem mit Version 2.0 im Jahr 2008.<sup>11</sup> Das iPhone gilt oft als Vorreiter moderner Smartphones. Bis heute läuft iOS nur auf Geräten des Herstellers Apple.

Es folgt eine Beschreibung des Betriebssystems iOS von dessen Anbieter Apple. „Das fortschrittlichste Betriebssystem der Welt“<sup>12</sup> – mit diesem Satz eröffnet Apple die Informationsseite zu seinem Betriebssystem. „Mit seiner benutzerfreundlichen Oberfläche, fantastischen Features und höchster Stabilität ist iOS das Fundament des iPhone, iPad und iPod touch. Und auch wenn andere versuchen aufzuschließen, sorgen die in iOS integrierten Technologien und Funktionen dafür, dass deine Apple Geräte anderen weiterhin um Jahre voraus sind.“<sup>13</sup> Hier wird versucht, das eigene Betriebssystem als sehr viel besser darzustellen, als die Betriebssysteme der Konkurrenz. Um das Betriebssystem aber kritisch zu analysieren, folgen nun einige Kritikpunkte des Systems aus Entwicklersicht.

Der große Erfolg des iPhones hat dazu geführt, dass viele Entwickler Applikationen für iOS entwickeln. Jedoch gibt es einige Punkte, welche die Softwareentwicklung für iOS erschweren. Zunächst ist zum Testen auf Endgeräten, statt im Simulator, eine kostenpflichtige Mitgliedschaft im iOS Developer Program erforderlich.<sup>14</sup> Weiterhin läuft das

<sup>11</sup> Vgl. Parker (2012).

<sup>12</sup> Apple (2013).

<sup>13</sup> Apple (2013).

<sup>14</sup> Vgl. Rodewig und Wagner (2012).

iOS-SDK, welches zum Entwickeln von iOS-Applikationen dient, „nur unter Mac OS X und setzt Apple-Rechner mit Intel-Prozessoren voraus. Alte Macs mit PowerPC-CPPUs sind daher ebenso außen vor wie Rechner mit anderen Betriebssystemen.“<sup>15</sup> Da Anwendungen für iOS nur über den App-Store von Apple erhältlich sind, geht der Entwickler ein gewisses Risiko ein. Wird eine Anwendung von Apple nicht genehmigt oder ändern sich Richtlinien, kann die Anwendung nicht über alternative Wege vertrieben werden und die Arbeit war umsonst.

### 8.3.2 Windows Phone

Microsofts Windows Phone ist die Weiterentwicklung des mobilen Betriebssystems Windows Mobile 6.5 und wurde im Oktober 2010 als Windows Phone 7 eingeführt. Windows Phone 7 setzt dabei auf ein neues Bedienkonzept, welches mit Windows Phone 8 erneut überarbeitet wurde.<sup>16</sup> „Mit Windows Phone 8 macht Microsoft jetzt einen erneuten radikalen Schnitt und räumt mit praktisch allen Kritikpunkten der Vorversionen auf. Zusätzlich wird das ursprüngliche Konzept noch einmal umfangreich optimiert, sodass sich erst jetzt das echte Potenzial von Windows Phone zeigt.“<sup>17</sup>

Wie bei modernen Smartphone-Betriebssystemen üblich gibt es auch in Microsofts Windows Phone die Möglichkeit, Anwendungen von Drittanbietern zu installieren. Hierfür bietet Microsoft den Microsoft Store an.<sup>18</sup>

Für Microsofts Windows Phone gilt das gleiche, wie für Apples iOS: Zur Entwicklung wird das firmeneigene Desktopbetriebssystem zwingend benötigt. Die Entwicklung mit dem Windows Phone SDK 8.0 ist nur mit einem Microsoft-Windows-8-Betriebssystem möglich.<sup>19</sup>

### 8.3.3 Android

Die Idee hinter Android ist, ein quelloffenes, freies Betriebssystem für mobile Geräte bereitzustellen. Vorangetrieben von Google wird hierfür im November 2007 die Open Handset Alliance gegründet. „Die internationale Vereinigung besteht zunächst aus 34 Unternehmen, darunter Handyhersteller, Mobilfunknetzbetreiber, Halbleiterproduzenten, Softwarefirmen und Marketingspezialisten.“<sup>20</sup> Ein Ziel dieser Vereinigung ist, „das geballte Know-how zahlreicher Branchengrößen“<sup>21</sup> einzusetzen, um „den Massenmarkt

---

<sup>15</sup> Vgl. Rodewig und Wagner (2012).

<sup>16</sup> Vgl. teltarif.de (2013).

<sup>17</sup> Theiss (2013).

<sup>18</sup> Vgl. Theiss (2013).

<sup>19</sup> Vgl. Microsoft (2013).

<sup>20</sup> Erdle (2013).

<sup>21</sup> Erdle (2013).

mit Touch-Handys [zu] erobern.“<sup>22</sup> Im Jahr 2008 kommt das erste mit Android betriebene Mobiltelefon auf den Markt. Die Möglichkeit, Anwendungen von Drittanbietern zu installieren, ist auch hier ein zentraler Bestandteil des Systems.

Es folgt eine Kurzbeschreibung des Betriebssystems von der offiziellen Webseite.

#### Kurzbeschreibung des Betriebssystems von der offiziellen Webseite

„Android is the world’s most popular mobile platform. [...] Android devices are already smart, and will only get smarter, with new features you won’t find on any other platform, letting you focus on what’s important and putting you in control of your mobile experience.“<sup>23</sup>

Auch hier wird dem Nutzer wieder vermittelt, dass dieses Betriebssystem anderen Betriebssystemen überlegen ist.

Als Entwickler bietet Android im Gegensatz zu iOS einige Vorteile. Zum einen kann unter Windows, Linux und Mac OS entwickelt werden, sodass jeder Entwickler in seiner bevorzugten Arbeitsumgebung entwickeln kann. Die benötigte Software ist kostenlos und es fallen keine Gebühren an, um die entwickelten Applikationen auf den eigenen Geräten zu testen. Beim Vertrieb der Applikationen gibt es einige Alternativen, sodass niemand auf nur einen einzigen Anbieter angewiesen ist.<sup>24</sup>

„Von Googles Android-Betriebssystem existieren am Markt eine Reihe von Versionen parallel. Diese werden anhand ihrer Versionsnummer oder ihres Codenamens unterschieden.“<sup>25</sup>

Dies ist für Entwickler deshalb interessant, da die verschiedenen Versionen dem Entwickler unterschiedliche Funktionen bieten. Im Android-SDK sind die Versionen in sogenannte API-Level unterteilt. Beim Erstellen eines Android-Projektes muss der Entwickler festlegen, welches API-Level er in seiner Anwendung unterstützen will. Neuere Versionen sind abwärtskompatibel.

In Abb. 8.2 zeigt sich, dass der Marktanteil der Versionen mit einem API-Level unter 10 relativ gering ist. Android-Versionen mit dem API-Level 10 haben jedoch einen Anteil von 39,7 %. Um möglichst viele Nutzer zu erreichen, sollte also höchstens das API-Level 10 ausgewählt werden.

Im folgenden Abschnitt soll erläutert werden, weshalb es so viele Smartphones mit älteren Android-Versionen gibt.

Ein Update muss in der Regel von den Herstellern kommen, da nur diese die nötigen Treiber für die Hardware haben. „Bei den 2009 und 2010 erschienenen Android-Smartphones dauerte es im Schnitt rund neun Monate, bis Gerätshersteller wie Samsung, Motorola oder Sony ein von Google veröffentlichtes Android-Update an ihre Nutzer wei-

---

<sup>22</sup> Erdle (2013).

<sup>23</sup> Google (2013).

<sup>24</sup> Vgl. Post (2012, S. 66–67).

<sup>25</sup> Kluczniok (2012).

Plattform	API-Level	Anteil
Android 1.6 Donut	4	0,1 %
Android 2.1 Eclair	7	1,7 %
Android 2.2 Froyo	8	4,0 %
Android 2.3.0 - 2.3.2 Gingerbread	9	0,1 %
Android 2.3.3 - 2.3.7 Gingerbread	10	39,7 %
Android 3.2 Honeycomb	13	0,2 %
Android 4.0.3 - 4.0.4 Ice Cream Sandwich	15	29,3 %
Android 4.1.x Jelly Bean	16	23,0 %
Android 4.2.x Jelly Bean	17	2,0 %

**Abb. 8.2** Marktanteil der Android-Versionen. (In Anlehnung an Pakalski (2013))

terreichten. Google veröffentlicht im Halbjahresrhythmus Updates. Viele Geräte hinken dem aktuellen Stand deshalb die meiste Zeit hinterher.<sup>26</sup>

Diese Updateproblematik kann als einer der großen Kritikpunkte des Android-Betriebssystems gesehen werden. Denn aus Sicht der Nutzer hat hier die Konkurrenz weit weniger Probleme.

### 8.3.4 Betriebssystemunabhängige Alternativen

Neben den von Betriebssystemen abhängigen nativen Applikationen gibt es noch Web-Applikationen. Web-Apps werden mittels HTML 5 und Javascript realisiert und benötigen zur Ausführung nur einen Browser und eine Internetverbindung. Leider können Web-Apps jedoch nicht auf plattformabhängige Features zurückgreifen. Auch entstehen Einschränkungen durch die Notwendigkeit einer Internetverbindung.

Abhilfe schaffen hier hybride Apps, welche versuchen die Vorteile von nativen Apps und Web-Apps zu nutzen und sich auch besser ins System einfügen als Web-Applikationen. Diese hybriden Apps bestehen aus einem Container, welcher aus einer nativen App besteht, die ein Webview beinhaltet. Der Kern der Anwendung besteht, wie bei Web-Apps, aus HTML 5 und Javascript und wird in diesen Container eingebettet. Die HTML-Dateien können in der App gespeichert werden, weshalb keine Internetverbindung notwendig ist, um die App auszuführen.

Gartner rechnet damit, dass bis 2016 mehr als die Hälfte der eingesetzten mobilen Applikationen hybrid sein wird.<sup>27</sup>

<sup>26</sup> Heise et al. (2012).

<sup>27</sup> Schaffry (2013).

Doch hybride Apps haben auch Nachteile gegenüber nativen Apps. Zum einen sind sie wesentlich langsamer. Dies liegt beispielsweise daran, dass sie kein Threading nutzen können, weshalb es nicht möglich ist, Aktionen parallel abzuarbeiten. Auch die fehlende Hardwarebeschleunigung für Grafikoperationen kann zu Problemen führen. Weiterhin ist es nicht immer einfach, die gerätespezifischen Features zu nutzen.<sup>28</sup>

---

## 8.4 Entwicklung einer Android-Applikation

### 8.4.1 Grundlagen zur Softwareentwicklung unter Android

Anwendungen unter Android werden in der Regel mit der Programmiersprache Java geschrieben. Der Android-SDK kompiliert diesen Java-Code und packt das Programm und alle anderen zugehörigen Ressourcen, wie Layoutdateien, Grafiken und Texte, in eine .apk-Datei. Diese Datei kann nun auf ein Gerät, welches mit dem Android-Betriebssystem läuft, installiert werden.<sup>29</sup>

#### 8.4.1.1 Dalvik Virtual Machine

Normalerweise wird Java-Bytecode von der Java Virtual Maschine (JVM) ausgeführt. Die JVM ist allerdings als Allzwecklösung für verschiedene Betriebssysteme konzipiert worden. Das Entwicklerteam von Dalvik wollte jedoch eine virtuelle Maschine, welche speziell an die Einschränkungen mobiler Geräte angepasst ist. Das Ergebnis ist eine ressourcenschonende, energieeffiziente virtuelle Maschine.<sup>30</sup>

Die Dalvik-VM führt nun allerdings nicht direkt den Java-Bytecode aus. Dieser muss zunächst nochmals mit dem Dalvik-Compiler zu Dalvik-Bytecode kompiliert werden.<sup>31</sup> Der Grund, warum nicht direkt von Java-Quellcode zu Dalvik-Bytecode kompiliert wird, ist, dass selbst wenn an der Sprache Java Änderungen vorgenommen werden, der Java-Bytecode gleich bleibt.<sup>32</sup> Abbildung 8.3 veranschaulicht die unterschiedlichen Kompilationsschritte zwischen Java-VM und Dalvik-VM. Für den Entwickler fallen hierbei keine zusätzlichen Arbeitsschritte an. Alle Kompilationsschritte werden von den Entwicklungswerkzeugen automatisiert.<sup>33</sup>

Jede Anwendung unter Android wird in einer eigenen virtuellen Maschine ausgeführt. Dies wird auch als Sandbox bezeichnet. Eine Sandbox soll dafür sorgen, dass jede Anwendung nur auf ihre Daten zugreifen kann.<sup>34</sup>

---

<sup>28</sup> Vgl. Henkelmann (2013).

<sup>29</sup> Vgl. Google (2013a).

<sup>30</sup> Vgl. Gargenta (2011, S. 9–10).

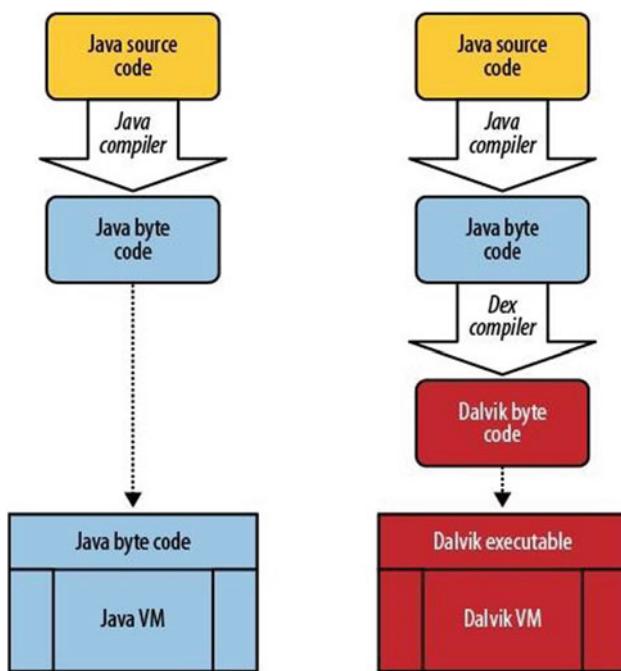
<sup>31</sup> Vgl. Gargenta (2011, S. 10).

<sup>32</sup> Vgl. Gargenta (2011, S. 11).

<sup>33</sup> Vgl. Gargenta (2011).

<sup>34</sup> Vgl. Google (2013a).

**Abb. 8.3** Vergleich JVM und Dalvik-VM. (Gargenta 2011, S.10)



Eine Android-Applikation besteht aus mehreren Komponenten. Wichtig sind für das Verständnis der nachfolgenden Ausführungen vor allem folgende Komponenten:

- Activities
- Services
- Intents
- Android Manifest
- Komponenten der Benutzeroberfläche

#### 8.4.1.2 Activities

Zunächst sollte die Definition einer Activity aus der offiziellen Internetseite für Entwickler von Android-Applikationen betrachtet werden:

„An Activity is an application component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface. The window typically fills the screen, but may be smaller than the screen and float on top of other windows.“

An application usually consists of multiple activities that are loosely bound to each other. Typically, one activity in an application is specified as the ‚main‘ activity, which is presented to the user when launching the application for the first time. Each activity can then start another activity in order to perform different actions. Each time a new activity

starts, the previous activity is stopped, but the system preserves the activity in a stack (the ‚back stack‘). When a new activity starts, it is pushed onto the back stack and takes user focus. The back stack abides to the basic ‚last in, first out‘ stack mechanism, so, when the user is done with the current activity and presses the Back button, it is popped from the stack (and destroyed) and the previous activity resumes.<sup>35</sup>

Activities sind also Komponenten, die zur Interaktion mit dem Benutzer dienen. Sie haben jeweils eine eigene Benutzeroberfläche und können wiederum andere Activities starten. Ein Programm besteht normalerweise aus mehreren Activities, welche für verschiedene Aufgaben vorgesehen sind.

Activities haben einige Methoden, die je nach Status aufgerufen werden. Abbildung 8.4 zeigt den Lebenszyklus dieser Methoden. Im Folgenden wird kurz auf die Methoden aus der Abbildung eingegangen.

**onCreate(){}:** Hier wird die Activity erstellt. In dieser Methode wird auch die Benutzeroberfläche gezeichnet.

**onStart(){}:** Die Activity wird sichtbar. Diese Methode wird sowohl beim ersten Start ausgeführt als auch nachdem eine andere Activity ausgeführt wurde und der Benutzer wieder zurückkehrt.

**onResume(){}:** Die Activity bekommt den Fokus. Ähnlich wie onStart(), jedoch wird diese Methode auch aufgerufen, wenn ein anderes Fenster kurz den Fokus hatte.

**onPause(){}:** Wird aufgerufen, wenn eine andere Activity in den Vordergrund gerät.

**onStop(){}:** Sobald die Activity nicht mehr sichtbar ist, wird onStop() aufgerufen.

**onDestroy(){}:** Wenn die Activity beendet wird, wird vorher onDestroy() aufgerufen.

#### 8.4.1.3 Services

Auch hier zunächst die Definition von der offiziellen Entwicklerwebsite:

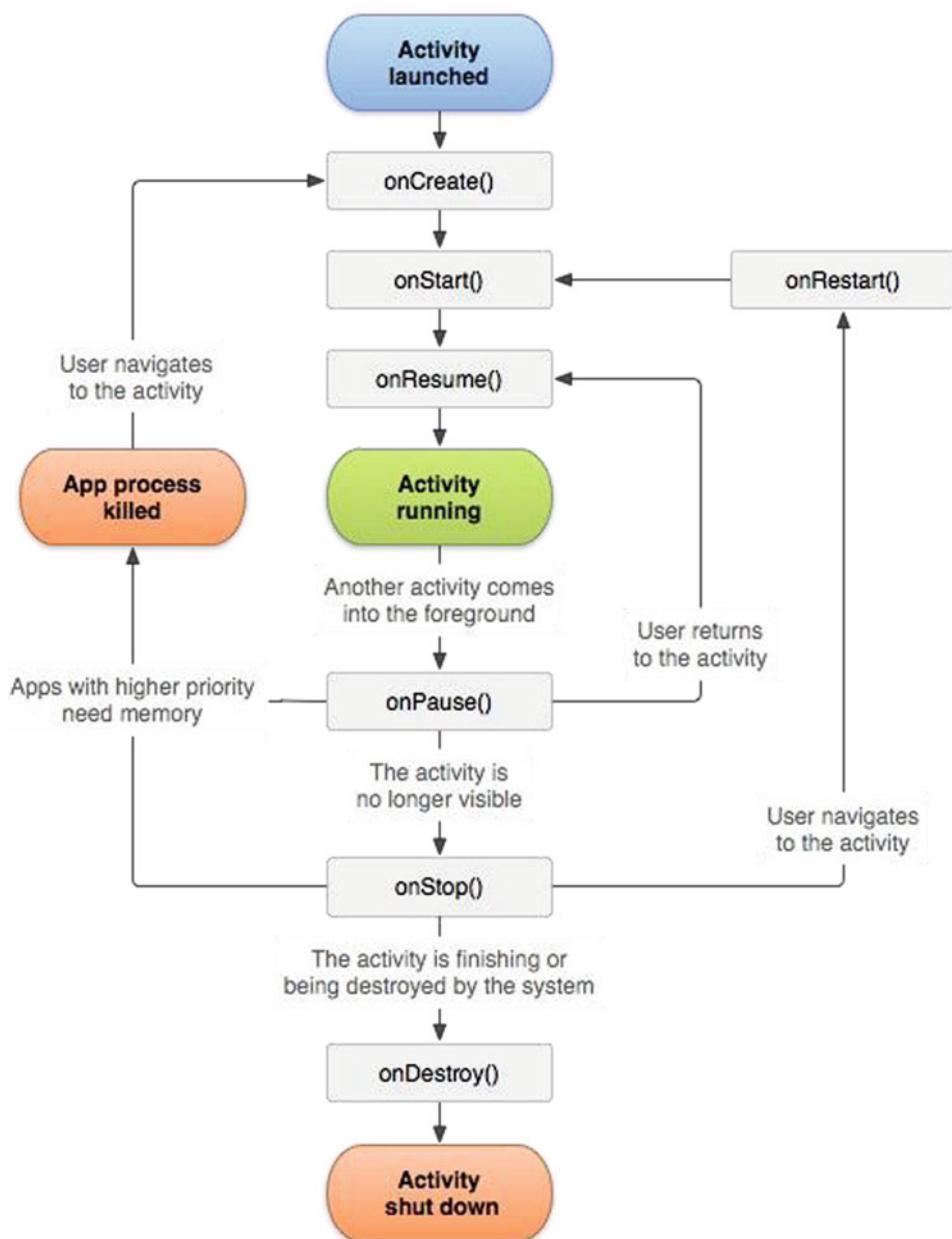
A Service is an application component that can perform long-running operations in the background and does not provide a user interface. Another application component can start a service and it will continue to run in the background even if the user switches to another application.<sup>36</sup>

Das heißt, ein Service ist der Teil einer Anwendung, welcher im Hintergrund und ohne UI ausgeführt wird. Ein Service wird von einer anderen Komponente, wie einer Activity, gestartet und läuft auch weiter, wenn die Activity nicht mehr im Vordergrund ist. Ein

---

<sup>35</sup> Google (2013b).

<sup>36</sup> Google (2013c).



**Abb. 8.4** Lebenszyklus einer Android Activity. (Google Inc. ohne Jahr b)

Beispiel ist eine App, die Musik abspielt. Auch wenn nach dem Start der Musik eine andere Anwendung aufgerufen wird, soll die Musik weiterlaufen.

Generell gibt es zwei Arten von Services: „Started“ und „Bound“. Wie der Name schon vermuten lässt, werden Started-Services einmal gestartet und laufen dann, bis sie einen Stoppbefehl bekommen. Bound-Services werden gestartet, sobald eine andere Komponente sich mit ihnen verbindet, und laufen so lange, bis alle verbundenen Komponenten die Verbindung lösen. Es ist auch möglich beide Arten miteinander zu kombinieren.<sup>37</sup>

#### 8.4.1.4 Intents

Activities und Services werden dadurch aufgerufen, dass eine andere Komponente eine Nachricht an sie sendet. Diese Nachrichten werden Intent genannt. In dem Intentobjekt werden verschiedene Daten gespeichert.<sup>38</sup>

► **Intentobjekt** Intent intent = new Intent(ErsteActivity.this, ZweiteActivity.class);

Hier wurde ein Intent erstellt, der als Kontext die erste Activity enthält und als Ziel die zweite Activity. Es können nun je nach Bedarf noch Daten, welche der neuen Activity übermittelt werden, in den Intent gespeichert werden, bevor die neue Activity gestartet wird.

#### 8.4.1.5 Android Manifest

Die offizielle Beschreibung der Android-Manifest-Datei lautet:

Every application must have an `AndroidManifest.xml` file (with precisely that name) in its root directory. The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's code.<sup>39</sup>

Ohne die Informationen aus der Android-Manifest-Datei kann das System also keine Anwendung ausführen.

„Among other things, the manifest does the following:

It names the Java package for the application. The package name serves as a unique identifier for the application.

It describes the components of the application – the activities, services, broadcast receivers, and content providers that the application is composed of. It names the classes that implement each of the components and publishes their capabilities (for example, which Intent messages they can handle). These declarations let the Android system know what the components are and under what conditions they can be launched.

It determines which processes will host application components.

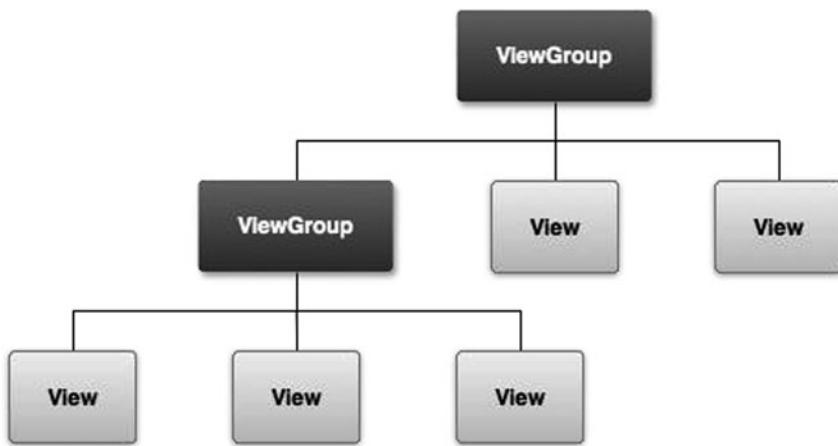
It declares which permissions the application must have in order to access protected parts of the API and interact with other applications.

---

<sup>37</sup> Vgl. Google (2013c).

<sup>38</sup> Vgl. Google (2013d).

<sup>39</sup> Google (2013e).



**Abb. 8.5** Beispielhafter Aufbau von View Groups und Views. (Google Inc., ohne Jahr, Creative Commons Attribution 2.5 Lizenz)

It also declares the permissions that others are required to have in order to interact with the application's components.<sup>40</sup>

Die Android-Manifest-Datei ist damit der zentrale Bestandteil jeder Android-App. Darin stehen wichtige Informationen über die Anwendung und deren Bestandteile. Alle Komponenten müssen im Android Manifest angemeldet werden, damit die Anwendung darauf zurückgreifen darf. Auch welche Activity als Einstiegspunkt festgelegt wird, welche Berechtigungen die Anwendung benötigt und was die minimal benötigte Android-Version ist, steht im Android Manifest. Außerdem steht darin die Versionsnummer der Anwendung, die bei Aktualisierungen über den Google Play Store eine Rolle spielt.

#### 8.4.1.6 Komponenten der Benutzeroberfläche

Die Benutzeroberfläche, auch User Interface (UI) genannt, wird in Android-Anwendungen durch sogenannte Views und View Groups abgebildet. Eine View ist ein Baustein, wie beispielsweise ein Textfeld (Textview) oder ein Bild (Imageview). View Groups sind Objekte, die Views und andere View Groups enthalten und bestimmen, wie diese angeordnet werden.<sup>41</sup>

In Abb. 8.5 ist ein beispielhafter Aufbau von View Groups und Views zu sehen. Die erste View Group enthält hier zwei Views und eine weitere View Group. Diese wiederum enthält drei weitere Views.

In der Regel wird das UI in XML-Dateien definiert. Dies macht es leichter für den Entwickler Design und Java-Code voneinander zu trennen. Auch der verwendete Text der App sollte nach Möglichkeit in einer XML-Datei statt im Java-Code untergebracht werden.

<sup>40</sup> Google ([2013e](#)).

<sup>41</sup> Vgl. Google ([2013f](#)).

Dies erleichtert das spätere Ändern von Texten und ist für eine mehrsprachige App eine Voraussetzung.<sup>42</sup>

Um die Übersicht zu erleichtern, empfiehlt es sich, für jede Activity eine XML-Datei mit dem Namen der Activity zu erstellen, welche dann das Layout dieser Activity enthält.

## 8.4.2 Die Entwicklungsumgebung

### 8.4.2.1 Eclipse und Android Developer Tools

Da Android-Programme in Java programmiert werden, muss als erstes der Java Development Kit (JDK) installiert werden. Der JDK dient zum Kompilieren der Java-Textdateien in .class-Dateien.

Eine weitere Voraussetzung ist der Android Software Development Kit (SDK): „The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.“<sup>43</sup>

Der Programmcode kann nun theoretisch in jedem Editor geschrieben und in der Kommandozeile kompiliert werden. Es empfiehlt sich aber die integrierte Entwicklungsumgebung Eclipse zu verwenden. Hierfür gibt es das Android-Developer-Tools Plug-in (ADT): „The Android Developer Tools (ADT) plugin for Eclipse provides a professional-grade development environment for building Android apps. It's a full Java IDE with advanced features to help you build, test, debug, and package your Android apps.“<sup>44</sup>

Am einfachsten gestaltet sich die Installation von Android-SDK, Android-ADT und Eclipse durch die Nutzung des ADT-Pakets, welches auf developer.android.com unter „Get the SDK“ zu erhalten ist.

Auch wenn dies in diesem Projekt nicht genutzt wird, soll trotzdem erwähnt werden, dass es die Möglichkeit gibt, C und C++ Code in Android-Apps zu verwenden. Hierzu gibt es den Android NDK (Native Development Kit). „The NDK is a toolset that allows you to implement parts of your app using native-code languages such as C and C++. For certain types of apps, this can be helpful so you can reuse existing code libraries written in these languages, but most apps do not need the Android NDK.“<sup>45</sup> Hierbei sollte jedoch der Großteil der Applikation in Java programmiert werden. Nur die Teile der Anwendung, welche beispielsweise auf eine C++ Bibliothek zugreifen müssen, sollten mit dem Android-NDK entwickelt werden.

#### 8.4.2.1 Installation und Konfiguration des ADT-Pakets

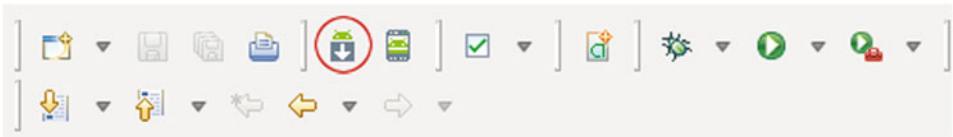
Nach dem Download des ADT-Pakets muss das Paket entpackt werden. In dem entpackten Ordner sollten sich nun die Ordner „eclipse“ und „sdk“ befinden. Im Ordner

<sup>42</sup> Vgl. Post (2012, S. 105–107).

<sup>43</sup> Google (2013g).

<sup>44</sup> Google (2013h).

<sup>45</sup> Google (2013i).



**Abb. 8.6** SDK-Manager starten

„eclipse“ befindet sich eine ausführbare Datei zum Starten der Entwicklungsumgebung. Um zukünftiges Starten zu erleichtern empfiehlt es sich, eine Verknüpfung darauf zu erstellen.

Beim ersten Start von Eclipse muss nun noch ein Workspace angelegt werden. Das ist der Ordner, in dem später die Projekte und deren Quelltexte abgespeichert werden.

Nach dem Start, sollte zunächst der SDK-Manager gestartet werden. Dies geschieht, indem das Android-Symbol aus Abb. 8.6 angeklickt wird. Hier können nun die verschiedenen Android-Versionen ausgewählt werden. Wie in Abschn. 8.3.3 beschrieben, sind die API-Level abwärtskompatibel. Deshalb muss der Entwickler hier die niedrigste unterstützte Version auswählen. Abbildung 8.7 zeigt beispielhaft wie der SDK-Manager eingestellt werden sollte, wenn API-Level 8 und höher unterstützt werden sollen.

#### 8.4.2.1.2 Virtuelle Geräte verwalten

Das ADT-Paket bringt ein sehr nützliches Feature mit: den Virtual-Device-Manager. Hiermit lassen sich virtuelle Android-Geräte erstellen. Diese lassen sich mit verschiedenen Android-Versionen, Displayauflösungen, internem Speicherplatz, Arbeitsspeicher und vielem mehr erstellen. Dies dient vor allem dem Testen der Anwendung auf verschiedenen Geräten.

Abbildung 8.8 zeigt das Symbol zum Starten des Virtual-Device-Managers.

In Abb. 8.9 sind die Einstellungen für ein Einsteiger-Smartphone mit Android 2.2 dargestellt.

Abbildung 8.10 zeigt wie ein virtuelles Android-Gerät aussieht, nachdem es gestartet wurde.

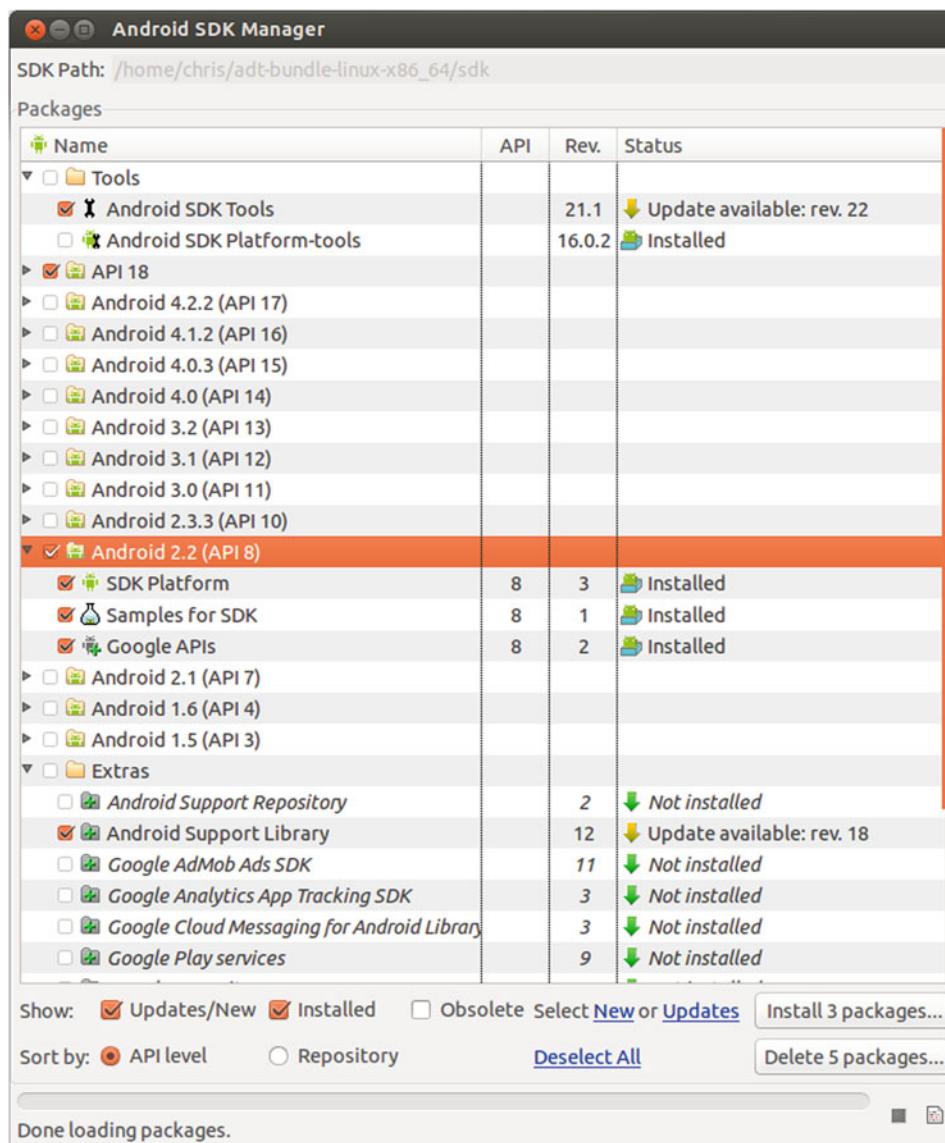
Da die Geräte jedoch nur simuliert werden, können bestimmte Eigenschaften, wie zum Beispiel die Bewegungssensoren, nicht wirklich getestet werden. Auch ist die Leistung eines mobilen Geräts nicht unbedingt gleich der Leistung des simulierten Geräts. Deshalb empfiehlt es sich, die Anwendung auf möglichst vielen echten Geräten zu testen.

#### 8.4.2.1.3 Ein echtes Gerät mit Eclipse verbinden

Um nun ein Smartphone oder Tablet mit Eclipse zu verbinden und die erstellte App darauf zu testen, müssen zunächst die Entwickleroptionen des Geräts aktiviert werden.

In Android Version 2.3 ist dies einfach zu finden. Unter Einstellungen/Anwendungen/Entwicklung muss der Haken bei USB-Verbindung gesetzt werden.

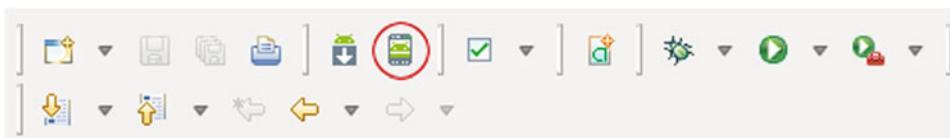
In Android Version 4.2 sind die Entwickleroptionen zunächst versteckt. Um sie sichtbar zu machen, muss unter Einstellungen/Über das Tablet bzw. Telefon 7-mal auf den Punkt



**Abb. 8.7** Android-SDK-Manger

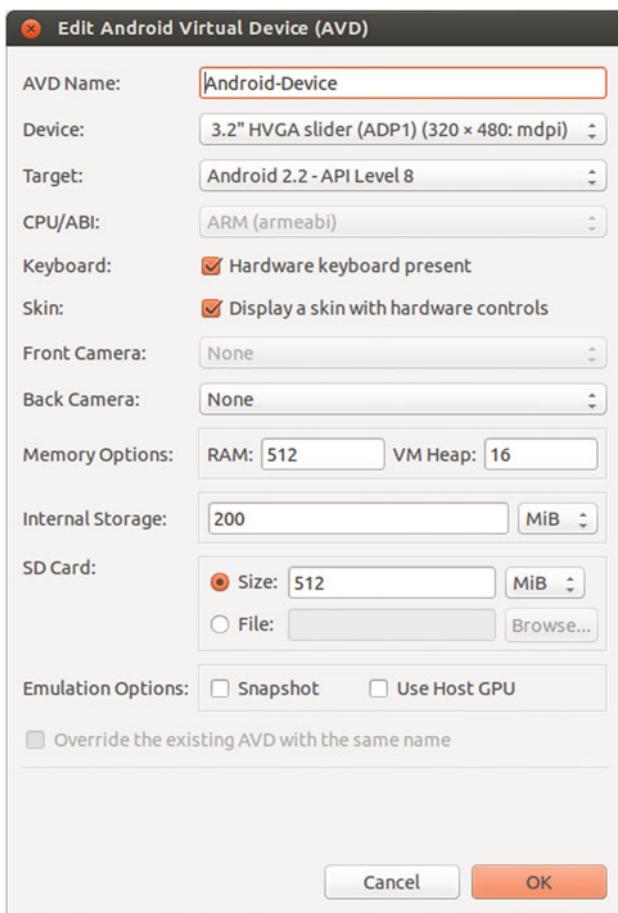
Build-Nummer getippt werden. Danach erscheint in den Einstellungen bei der Kategorie „System“ der Eintrag „Entwickleroptionen“. Hier muss nun der Haken bei USB-Debugging gesetzt werden.

Hiernach kann das Android-Gerät per USB an den PC angeschlossen werden. Nachdem in Eclipse die Anwendung gestartet wird, erscheint in der Liste der Geräte das angeschlossene Gerät. Wird das Gerät ausgewählt und installiert, startet Eclipse die Anwendung automatisch.



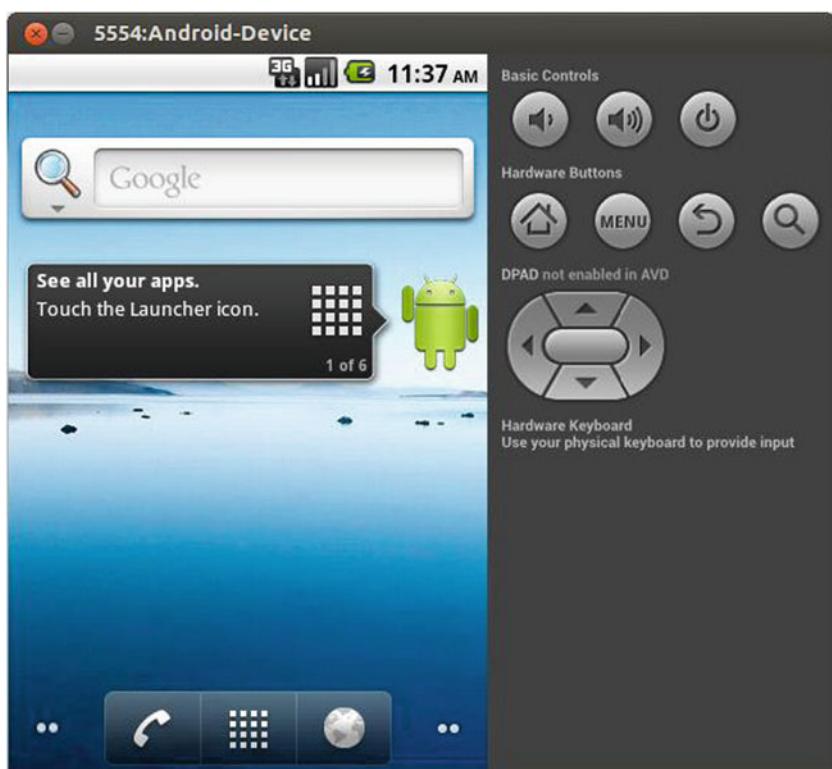
**Abb. 8.8** Starten des Virtual-Device-Managers

**Abb. 8.9** Android Virtual-Device-Manager



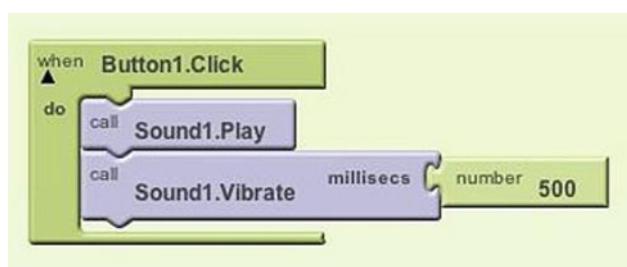
#### 8.4.2.2 MIT App Inventor

Der App Inventor von Google und dem Massachusetts Institute of Technology (MIT) ist ein webbasiertes Programm zur Erstellung von Android-Applikationen. Ziel ist es, Anwendungen für das Android-Betriebssystem ohne Programmierkenntnisse zu entwickeln. Hierfür wird per Drag-and-drop zunächst die Benutzeroberfläche erstellt. Das Verhalten



**Abb. 8.10** Virtuelles Android-Gerät

**Abb. 8.11** Entwicklungsoberfläche des App Inventors.  
(Massachusetts Institute of Technology ohne Jahr)



der Applikation wird dann mittels Bausteinen zusammengesetzt.<sup>46</sup> Abbildung 8.11 zeigt, wie eine Schaltfläche auf einen Klick reagieren soll: Es wird eine Audiodatei abgespielt und das Gerät vibriert für 500 ms.

Mithilfe des App Inventors lassen sich sehr schnell funktionsfähige Anwendungen entwickeln. Dadurch eignet das Werkzeug sich auch dafür, für komplexere Anwendungen

<sup>46</sup> Abelson et al. (2011, S. XV).

erste Prototypen zu entwickeln oder zum Testen der Usability verschiedener grafischer Benutzeroberflächen.

Leider lassen sich die Anwendungen nicht als Java-Quelltext exportieren und weiter bearbeiten. Da jedoch einige Limitierungen vorhanden sind, können momentan nicht alle Funktionen ohne Weiteres realisiert werden. Beispielsweise kann eine XML-Datei nur dadurch eingelesen werden, indem die Datei an einen Webserver übergeben wird, welcher die Lesevorgänge vornimmt und die gewünschten Daten an die Applikation zurückgibt.<sup>47</sup>

### 8.4.3 Konzeption und Realisierung

In der Phase der Konzeption wird festgehalten, was die Anwendung leisten soll. Ausgehend von der Idee werden die benötigten Funktionen abgeleitet. In Abschn. 8.5.1 wird dies am Beispiel der Presenter-Anwendung dargestellt.

Wenn es um die Konzeption eines IT-Systems geht, fallen oft auch die Begriffe Lastenheft und Pflichtenheft. Ein Lastenheft beschreibt die Wünsche und Anforderungen an ein IT-System aus Sicht des Auftraggebers. Ein Lastenheft kann als ein Ausschreibungsdokument verwendet werden. Das Gegenstück zu einem Lastenheft ist das Pflichtenheft. Dieses ist die Antwort eines IT-Dienstleisters auf ein Lastenheft. Das Pflichtenheft beschreibt detailliert die Leistungen, zu der sich der Dienstleister verpflichtet [...].<sup>48</sup>

Nach der Konzeption sollte feststehen, welche Funktionen realisiert werden müssen bzw. können. Nun kann beispielsweise, wie in Extreme Programming, iterativ vorgegangen werden. Hierbei wird zunächst eine Funktion realisiert und getestet. Sobald diese Funktion problemlos läuft, geht es zur Realisierung der nächsten Funktion, welche wieder getestet wird. Dies wiederholt sich, bis alle Funktionen implementiert sind. Es empfiehlt sich, die Funktionen mit Prioritäten zu versehen und zunächst die Grundfunktionalität der Anwendung zu realisieren, bevor zusätzliche Features wie alternative Eingabemethoden entwickelt werden.

Bevor jedoch programmiert werden kann, muss zunächst noch weiter analysiert werden, wie die verschiedenen Funktionen umgesetzt werden sollen. Dabei sind vor allem die Komponenten, welche Android zur Verfügung stellt, zu berücksichtigen. Ist es sinnvoll, eine Funktion im Hintergrund als Service zu starten? Ab wann wird eine neue Activity gestartet, statt die vorhandene mit neuen Daten zu füllen?

---

<sup>47</sup> Massachusetts Institute of Technology (2013 und 2013a).

<sup>48</sup> Brandt-Pook und Kollmeier (2008, S. 14).

## 8.5 Die Android Presenter App

### 8.5.1 Konzept

Die entwickelte Applikation soll aus einem Android-Smartphone oder einem Tablet-PC ein Präsentationsgerät für PowerPoint-Präsentationen machen. Übliche Funktionen, welche ein Presenter hat, sind unter anderem die Navigation durch die Präsentation, ein Laserpointer zum Hervorheben bestimmter Inhalte, eine kabellose Verbindung zu dem Präsentationscomputer und bei einigen Geräten auch ein Zeitmesser.

Hieraus lassen sich folgende Funktionen für die Applikation ableiten. Zunächst Funktionen, welche die Basisfunktionalität sicherstellen:

- Kabellose Verbindung aufbauen
- Steuerung der Präsentation

Darüber hinaus könnte die Applikation über folgende Funktionen verfügen:

- Anzeige von Informationen zur aktuellen Präsentation
- Timer
- Simulation eines Laserpointers

Einstiegspunkt der Applikation wird also eine Activity sein, welche es ermöglicht, eine kabellose Verbindung zum Präsentationscomputer aufzubauen. Sobald die Verbindung steht, ist es sinnvoll, diese als Service im Hintergrund laufen zu lassen, sodass sie durchgehend zur Verfügung steht.

Die Steuerung der Präsentation sollte eine eigene Activity sein. Diese könnte aufgerufen werden, nachdem die Verbindung zum Präsentationscomputer aufgebaut ist. Werden später weitere Activities implementiert, sollten diese dann vor der Steuerungsactivity eingebunden werden.

### 8.5.2 Umsetzung

#### 8.5.2.1 Kabellose Verbindung

Zur Steuerung der Präsentation über ein Smartphone oder einen Tablet-PC müssen Präsentationscomputer und das Android-Gerät Daten austauschen können. Es ist sinnvoll, dies über eine kabellose Verbindung zu realisieren. Als Möglichkeiten stehen hierzu Bluetooth und Wireless-LAN zur Verfügung.

### 8.5.2.1.1 Bluetooth

► „Bluetooth ist ein Funkverfahren, welches es ermöglicht, Geräte über eine weltweit standardisierte Schnittstelle miteinander zu verbinden. [...]“<sup>49</sup>

Bei der Bluetoothtechnik wird nur mit 10–100 mW gesendet. Dadurch beträgt die Reichweite 10–100 m. Die Datenrate ist im Vergleich zu Wireless-LAN relativ gering einzustufen.<sup>50</sup>

Da Bluetooth vor allem für mobile Geräte entwickelt wurde und diese eine begrenzte Stromversorgung haben, wurde Bluetooth von Anfang an mit dem Ziel, möglichst wenig Energie zu verbrauchen, entwickelt.<sup>51</sup>

Bluetooth arbeitet im weltweit lizenzfrei verfügbaren 2,4 GHz ISM-Band. Da hier die Kanalauslastung potenziell sehr hoch ist, muss ein Verfahren genutzt werden, um Störungen zu vermeiden.<sup>52</sup> Dabei wird ein spezielles Frequenzsprungverfahren verwendet. Hierbei wechseln die Geräte bis zu 1600-mal pro Sekunde die Frequenz. Durch dieses Verfahren kommt es selten zu Störungen.<sup>53</sup>

### 8.5.2.1.2 WLAN nach IEEE-802.11-Standard

**IEEE 802.11** ist eine Gruppe von Standards für ein Funknetzwerk auf Basis von Ethernet. Damit ist IEEE 802.11 das am weitesten verbreitete drahtlose Netzwerk bzw. Wireless Local Area Network (WLAN). [...] Der Standard baut auf den anderen Standards von IEEE 802 auf. IEEE 802.11 ist, vereinfacht ausgedrückt, eine Art schnurloses Ethernet. IEEE 802.11 definiert die Bitübertragungsschicht des OSI-Schichtenmodells für ein Wireless LAN. Dieses Wireless LAN ist, wie jedes andere IEEE-802-Netzwerk auch, vollkommen Protokoll-transparent. Drahtlose Netzwerkkarten lassen sich deshalb ohne Probleme in jedes vorhandene Ethernet einbinden. So ist es ohne Einschränkungen möglich, eine schnurgebundene Ethernet-Verbindung nach IEEE 802.3 gegen eine Wireless-LAN-Verbindung nach IEEE 802.11 zu ersetzen.“<sup>54</sup>

Wireless-LAN nach dem IEEE-802.11-Standard arbeitet im Frequenzbereich des ISM-Bands (2,4 GHz) von 2,4000 bis 2,4835 GHz.<sup>55</sup> Ursprünglich wurde hiermit eine Über-

---

<sup>49</sup> Walter (2003, S. 4).

<sup>50</sup> Vgl. Walter (2003, S. 4).

<sup>51</sup> Vgl. Walter (2003, S. 4).

<sup>52</sup> Vgl. Sikora (2008).

<sup>53</sup> Vgl. Telzerow (2008).

<sup>54</sup> Schnabel (2013a).

<sup>55</sup> Vgl. Schnabel (2013b).

tragungsrate von bis zu 2 MBit erreicht. Der Standard wurde jedoch mehrfach erweitert. Folgende Übersicht zeigt einige dieser Erweiterungen und deren Übertragungsrate:<sup>56</sup>

- „IEEE 802.11b/WLAN mit 11 MBit
- IEEE 802.11 g/WLAN mit 54 MBit
- IEEE 802.11a/h/j/WLAN mit 54 MBit
- IEEE 802.11n/WLAN mit 150 MBit<sup>57</sup>

### 8.5.2.1.3 Eignung der Verfahren

Generell sind beide Verfahren gut geeignet und es wäre möglich beide zu implementieren. Jedoch wird aus Zeitgründen zunächst nur das am besten geeignete Verfahren ausgewählt.

„Während der Entwicklung des WLAN-Standards IEEE 802.11 und Bluetooth haben sich schnell Gemeinsamkeiten herausgestellt. Beide Funkstandards arbeiten im Frequenzband 2,4 GHz und sollen unterschiedliche Geräte über Funk miteinander verbinden. Beide Standards zeichnen sich durch individuelle Stärken aus und kommen dadurch in verschiedenen Geräten auf den Markt.

Wireless LAN übertrifft Bluetooth in seiner Reichweite und Übertragungsgeschwindigkeit und kommt deshalb in lokalen Netzwerken zum Einsatz.

Bluetooth ist mit geringen Hardwarekosten, niedrigem Stromverbrauch und Echtzeitfähigkeit in den Bereichen Sprachübertragung, Audio-Video-Lösungen und Adhoc-Verbindungen zwischen Kleinstgeräten besser geeignet. Bluetooth löst hier Irda (Infrarot) erfolgreich ab. Und Bluetooth 3.0 macht sich WLAN-Techniken zunutze, um große Datenmengen zu übertragen.“<sup>58</sup>

Hier wurden bereits die größten Unterschiede zwischen den Verfahren dargestellt. Die Eigenschaften, die bei der Auswahl für dieses Projekt eine Rolle spielen, sind:

- Stromverbrauch
- Verfügbarkeit
- Konfiguration
- Störanfälligkeit
- Schwierigkeitsgrad der Implementierung

Der Stromverbrauch von Bluetooth ist mit 10–100 mW deutlich geringer als der Verbrauch, welcher mit einer WLAN-Verbindung entsteht. Bei modernen Smartphones ist normalerweise die Verfügbarkeit beider Verfahren gewährleistet. Jedoch muss hier auch der Präsentationscomputer betrachtet werden. Je nachdem, ob es sich hierbei um einen Desktop-PC, ein Notebook oder ein Netbook handelt, ist die Verfügbarkeit der beiden Verfahren unterschiedlich. Jedoch lässt sich beides für wenig Geld über einen USB-Adapter nachrüsten. Die Konfiguration ist bei beiden Verfahren, je nach Geschick des Entwicklers,

<sup>56</sup> Vgl. Schnabel (2013a).

<sup>57</sup> Vgl. Schnabel (2013a).

<sup>58</sup> Vgl. Schnabel (2013a).

	<i>Bluetooth</i>	<i>Wireless-LAN</i>
Stromverbrauch	1	0
Verfügbarkeit	0	1
Konfiguration	1	0
Störanfälligkeit	1	0
Schwierigkeitsgrad	0	0
Summe	3	1

**Abb. 8.12** Eignung von Bluetooth und Wireless-LAN

einfach bis umständlich. Jedoch kann es vorkommen, dass durch wechselnde IP-Adressen die Konfiguration per WLAN jedesmal erneut durchgeführt werden muss, wohingegen die Konfiguration zwischen Bluetoothgeräten normalerweise nicht wiederholt werden muss. „Die Bluetooth-Technologie hat den besonderen Vorteil, dass sie unempfindlich ist gegenüber Überlagerungen parallel betriebener Funknetze in einer Wohnung oder einem Büro. Zudem kennen Bluetooth-Verbindungen keine Störanfälligkeit durch Geräte, die im gleichen Frequenzband arbeiten (Mikrowellen, Garagentüröffner, schnurlose Telefone). Bei WLAN-Netzwerken besteht grundsätzlich die Gefahr von Beeinträchtigungen der Datenrate beim Betrieb eines WLANs in der Nähe solcher Geräte. Zu Störungen im WLAN-Empfang kann es auch kommen, wenn in einem kleinen Umkreis mehrere WLANs betrieben werden“<sup>59</sup> Die Störanfälligkeit ist also bei Bluetooth geringer als bei WLAN. Die Implementierung gestaltet sich bei beiden Verfahren ähnlich, jedoch muss bei Bluetooth nicht darauf geachtet werden, dass die Verbindung von einer Firewall blockiert werden könnte.

Wie aus Abb. 8.12 ersichtlich wird, eignet sich Bluetooth besser als Wireless-LAN. Deshalb wird im Rahmen dieser Arbeit das Bluetoothverfahren zum Einsatz kommen. Es ist jedoch festzuhalten, dass es sinnvoll ist, in kommenden Versionen der Anwendung auch andere Möglichkeiten der Datenübertragung zu implementieren.

#### 8.5.2.1.4 Implementierung

Um die Verbindung zwischen PC und Android-Gerät herzustellen, muss auf beiden Geräten eine Anwendung laufen. In diesem Projekt wird die Verbindung dadurch realisiert, dass die Anwendung auf dem PC, ab jetzt auch Bluetooth-Server genannt, auf eine eingehende Verbindung eines gekoppelten Gerätes wartet. Um den Verbindungsaufbau zu initialisieren wird in der Android-Applikation eine neue Activity erstellt, in welcher eine Liste der gekoppelten Bluetoothgeräte erscheint. Hier soll auch ein Hinweis erscheinen, wie PC und Android-Gerät gekoppelt werden können. Sind die Geräte gekoppelt,

<sup>59</sup> AVM (2013).

kann die Verbindung aufgebaut werden. Die Vorgehensweise orientiert sich stark an der Bluetooth-Fernsteuerung von Thuy.<sup>60</sup>

Im Folgenden wird zunächst ein Ausschnitt des Quelltextes der Bluetooth-Server-Anwendung für den PC analysiert. Dieser Ausschnitt wurde auf die relevanten Teile gekürzt.

```
StreamConnectionNotifier notifier;
StreamConnection connection = null;
try {
    UUID uuid = new UUID("0000110100001000800000805F9BCACA", false);
    String url = "btsp://localhost:" + uuid.toString() +
";name=RemoteBluetooth";
    notifier = (StreamConnectionNotifier)Connector.open(url);
} catch (BluetoothStateException e) {e.printStackTrace();return;} catch
(IOException e) {e.printStackTrace();return;}
// waiting for connection
while(true) {try {
    connection = notifier.acceptAndOpen();
    Thread processThread = new Thread(new
ProcessConnectionThread(connection));
    processThread.start();
} catch (Exception e) {e.printStackTrace();return;}}
}
```

### Quelltextfragment 1: Bluetooth-Server-Anwendung<sup>61</sup>

Zunächst wird ein Universally Unique Identifier (UUID) vergeben. Die UUID dient der Identifikation der Anwendung und gibt an, welcher Bluetoothdienst verwendet wird.<sup>62</sup> Hiernach wird mittels des Objekts „notifier“ der Klasse „StreamConnectionNotifier“ auf eine eingehende Verbindung gewartet. Diese Verbindung kann nur von einer Anwendung mit derselben UUID aufgebaut werden. Sobald eine passende Verbindungsanfrage eingeht, wird versucht die Verbindung aufzubauen. Da sich das Warten auf eine Verbindung und der Verbindungsaufbau in einer Schleife befinden, wird, nachdem die Verbindung beendet wurde, erneut auf eine eingehende Verbindung gewartet. Der Status der Verbindung wird anhand eines Symbols in der Taskleiste angezeigt. Hier befindet sich auch die Option, das Programm zu beenden.<sup>63</sup>

<sup>60</sup> Vgl. Thuy (2011).

<sup>61</sup> Vgl. Thuy (2011).

<sup>62</sup> Vgl. Bluetooth SIG (2013).

<sup>63</sup> Vgl. Oracle (2013).

Damit die Android-Anwendung nun also mit dem Bluetooth-Server eine Verbindung aufbauen kann, muss dieselbe UUID genutzt werden. Weiterhin wird zum Aufbau der Verbindung die MAC-Adresse (Media Access Control) des Bluetooth-Servers benötigt. Die MAC-Adresse ist die eindeutige Gerätenummer eines Netzwerkadapters.<sup>64</sup> Diese und weitere Informationen können mithilfe der Methode „getBondedDevices()“ der „BluetoothAdapter“-Klasse für gekoppelte Geräte ermittelt und in ein Objekt der Klasse „BluetoothDevice“ gespeichert werden.<sup>65</sup> Nachdem das zu verbindende Gerät ermittelt ist, kann mittels dieses Objekts ein Bluetoothsocket erstellt werden, welcher nun die Verbindung zu dem wartenden Bluetooth-Server aufbauen kann.

Alle Vorgänge, welche für die Verbindung zum Server und für die Datenübertragung verantwortlich sind, sind in der Klasse „BluetoothService“ implementiert. Diese Klasse ist eine Android-Serviceklasse und kann deshalb im Hintergrund laufen. Dadurch können alle Activities der Anwendung darauf zugreifen und die Verbindung bleibt bestehen bis die Anwendung beendet wird.<sup>66</sup> Um den Nutzer darauf aufmerksam zu machen, dass der Service läuft, erscheint ein Symbol in der Benachrichtigungsleiste von Android. Hierüber ist es auch möglich, die Anwendung zu beenden.<sup>67</sup>

### 8.5.2.2 Steuerung der Präsentation

Nachdem die Verbindung zwischen den Endgeräten aufgebaut ist, muss nun die Steuerung der Präsentation über das mobile Gerät ermöglicht werden. Zuerst muss hierfür festgelegt werden, über welche Eingaben am Android-Gerät die Steuerung der Präsentation ermöglicht werden soll. Ein mögliches Konzept wäre die Steuerung über Schaltflächen auf der Benutzeroberfläche. Es wäre auch möglich die Steuerung mittels sogenannter „Wischgesten“ durchzuführen. Hierfür wird mit dem Finger von einer Seite des Displays zur anderen gewischt. Auch ist es möglich, die Hardwaretasten mit neuen Funktionen zu belegen und hierüber die Steuerung vorzunehmen.

Es können natürlich mehrere Konzepte realisiert werden. Denkbar wäre dann, den Nutzer über Optionen auswählen zu lassen, welche Art der Steuerung er bevorzugt. Da die Anwendung zunächst nur ein Steuerungskonzept benötigt, wird die Steuerung über die Hardwaretasten realisiert, da diese am ehesten der eines handelsüblichen Presenters gleicht. Bei diesem Steuerungskonzept muss auch seltener auf das Display geschaut werden.

---

<sup>64</sup> Vgl. University of Arizona (2013).

<sup>65</sup> Vgl. Google (2013j).

<sup>66</sup> Vgl. Google (2013k).

<sup>67</sup> Vgl. Google (2013c).

```
@Override  
public boolean dispatchKeyEvent(KeyEvent event) {  
    int action = event.getAction();  
    int keyCode = event.getKeyCode();  
    switch (keyCode) {  
        case KeyEvent.KEYCODE_VOLUME_UP:  
            if (action == KeyEvent.ACTION_UP) {  
                mService.write(BluetoothCommandService.VOL_UP);  
            }  
    }  
}
```

### Quelltextfragment 2: Funktionstasten zur Steuerung neu belegt

Hierzu werden, sobald die Activity zur Steuerung erstellt wird, die Eingaben über die Lautstärketasten überwacht. Außerdem wird die Standardfunktion dieser Tasten deaktiviert und ihnen wird eine neue Funktion zugeteilt. Sobald eine dieser Tasten gedrückt wird, wird ein ihr zugeteilter Wert über die Bluetoothverbindung übertragen. Auf dem Präsentationscomputer empfängt die Anwendung, welche für die Bluetoothverbindung verantwortlich ist, diesen Wert und, je nachdem welcher Wert übertragen wurde, wird nun auf dem Präsentationscomputer ein Tastendruck der Pfeil-nach-Links-Taste oder der Pfeil-nach-rechts-Taste simuliert.

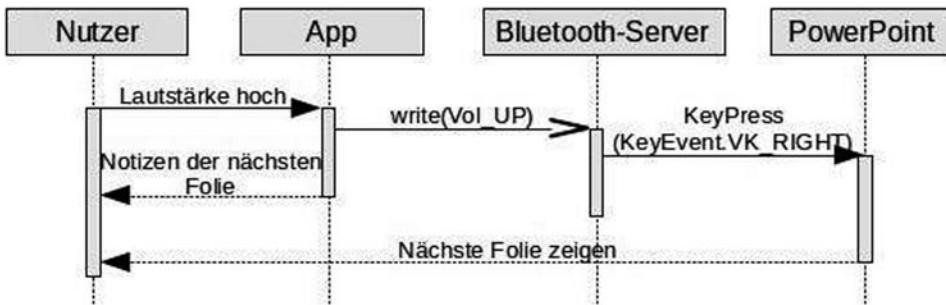
In Quelltextfragment 2 wird, sobald eine Taste gedrückt wird, geschaut, welche Taste gedrückt wurde. Ist die Taste die Lautstärke-Hoch-Taste, wird die Methode „write(int out)“ der Klasse „BluetoothService“ aufgerufen, welche den ihr übergebenen Wert per Bluetooth an den Präsentationscomputer übergibt. In dem Beispiel wird die Konstante VOL\_UP übergeben, welche den Wert 1 enthält (Abb. 8.13).

#### 8.5.2.3 Verbesserte Navigation

Eine solche Anwendung kann natürlich nicht nur dazu genutzt werden, eine Folie vor oder zurück zu navigieren. Es ist auch möglich, zu einer bestimmten Folie zu springen, egal bei welcher Folie die Präsentation gerade ist.

Hierfür muss lediglich die Foliennummer ermittelt werden und die Tastenkombination FOLIENNUMMER + ENTER an den Präsentationscomputer geschickt werden. Die Umsetzung könnte so aussehen, dass der Anwender in der Präsentationsactivity die Möglichkeit bekommt, mit einer Wischgeste vom linken Rand zur Bildschirmmitte eine Liste mit allen Folien aufzurufen. Wählt er eine Folie aus, dann springt die PowerPoint-Präsentation zu dieser Folie.

Um dies zu implementieren, muss in der Android-Anwendung die Methode „write(int out)“ der Klasse „BluetoothService“ angepasst werden. Diese könnte nun in einem zweiten Parameter die Foliennummer übergeben. Außerdem muss natürlich die Bluetooth-



**Abb. 8.13** Sequenzdiagramm zur Steuerung der Präsentation

Server-Anwendung angepasst werden, damit diese dann die Tastenkombination ausführt. Weiterhin muss die Activity so angepasst werden, dass eine Wischgeste die Liste mit den Folien aktiviert.

Um diese Liste aus der Präsentationsdatei auszulesen, muss die Anwendung jedoch zunächst das PowerPoint-Dateiformat lesen können.

### 8.5.2.4 Anzeige von Informationen zur aktuellen Präsentation

#### 8.5.2.4.1 Analyse der Alternativen

Um Informationen über die aktuelle Präsentation anzeigen zu können, muss die Applikation diese Informationen zunächst erhalten. Um die gewünschten Informationen aus der Präsentationsdatei auszulesen, gibt es verschiedene Möglichkeiten.

Eine Möglichkeit wäre, die Datei von der Applikation selbst einlesen zu lassen. Hierfür muss die Datei auf dem Gerät sein und entweder manuell geöffnet oder von der Applikation gefunden werden. Hierfür könnte die Speicherkarte des Gerätes nach Präsentationen durchsucht werden. Die Applikation könnte nun eine Liste der gefundenen Präsentationen anzeigen. Bei vielen Präsentationen und geringen Speicherkapazitäten könnte dies jedoch bald dazu führen, dass die Speicherkarte voll ist.

Eine weitere Möglichkeit wäre, die Präsentation auf einen Webserver zu laden, welcher die Präsentationsdatei auswertet und die gewünschten Informationen an die Applikation zurücksendet. Der Vorteil wäre, dass wesentlich weniger Speicherkapazitäten auf dem mobilen Gerät benötigt werden. Jedoch könnte das Speichern der Dateien im Internet bei Nutzern Bedenken bezüglich des Datenschutzes auslösen. Außerdem müsste zunächst eine Infrastruktur für den Webservice angelegt werden. Die Kapazitäten müssten bei schnell steigenden Nutzerzahlen auch einfach erweiterbar sein.

Es wäre auch möglich, die Dateien nur auf dem Präsentationscomputer zu haben. Die Anwendung, welche auch für die Fernsteuerung der Präsentation auf dem Präsentationscomputer zuständig ist, könnte in diesem Fall erweitert werden, um die Präsentation auszuwerten und die gewünschten Informationen an das Smartphone zu senden.

Da nach der Erfahrung des Autors oft an fremden Computern präsentiert wird, muss die Präsentation in diesem Fall auf einem mobilen Datenträger mitgebracht werden. In der ersten vorgestellten Variante kann das Smartphone die Datei direkt an den Präsen-

tationscomputer übertragen. Ein Vorteil gegenüber der letzten Variante ist, dass die Informationen zur Präsentation schon auf dem Smartphone vorliegen und somit auch vor der Präsentation nochmals abrufbar sind. In diesem Projekt wird aus diesen Gründen zunächst die erste Variante implementiert. Es ist jedoch möglich in zukünftigen Versionen alle Möglichkeiten anzubieten und den Anwender entscheiden zu lassen.

#### 8.5.2.4.2 Gewünschte Informationen definieren

Bevor nun erläutert wird, wie Informationen aus einer Präsentationsdatei ausgelesen werden können, wird an dieser Stelle definiert, welche Informationen dem Nutzer einen Mehrwert bringen.

Da das Display eines Smartphones relativ klein ist, ergibt es wenig Sinn, die komplette Folie auf dem Bildschirm darzustellen. Interessante Informationen für den Präsentierenden, welche nicht zu viel Platz auf dem Display einnehmen, wären beispielsweise die Foliennummer, die Anzahl der Animationen der aktuellen Folie oder die Überschrift der Folie.

Außerdem bietet PowerPoint eine Notizfunktion an. Diese Funktion ermöglicht es, für jede Folie Notizen zu hinterlegen. Da hier Informationen stehen können, die sich nicht auf der Folie befinden, bietet es sich an, diese Informationen auf dem Smartphone bereitzustellen.

Da die Notizen jede Menge wertvolle Informationen für den Präsentierenden bereitstellen können, wird hierauf nun der Fokus gelegt.

#### 8.5.2.4.3 PowerPoint-Dateiformat

Da die Anwendung Informationen aus PowerPoint-Dateien auslesen soll, wird an dieser Stelle das PowerPoint-Dateiformat analysiert. PowerPoint-Dateien können in unterschiedlichen Formaten vorliegen. Bis zu der Veröffentlichung von Microsoft Office 2007 wurde das binäre Dateiformat PPT genutzt, welches von dem Open XML Format PPTX abgelöst wurde.<sup>68</sup>

Zunächst wird nun das ältere Format PPT betrachtet. Wie bereits erwähnt handelt es sich hierbei um ein binäres Dateiformat. Was dies bedeutet, wird im folgenden Absatz kurz erläutert.

Es handelt sich um Dateien mit numerischen Inhalten, wobei ohne Kenntnis der Spezifikation des jeweiligen Dateiformats nicht erkennbar ist, ob die gespeicherten Zahlen an bestimmten Stellen für Befehle oder für Nutzdaten stehen. Die meisten Formate nehmen nichtsdestotrotz eine klare Trennung vor: Sie schreiben die Verwaltungsinformationen zu Beginn der Datei in einen Header und hängen die Nutzdaten hinten an. Auf diese Weise wird einem Programm, das eine solche Datei öffnet, zuerst mitgeteilt, wie genau es mit den Daten verfahren soll, bevor diese Daten selbst gelesen werden.<sup>69</sup>

---

<sup>68</sup> Vgl. Microsoft (2006).

<sup>69</sup> Kersken (2004).

Daraus folgt, dass, um die Nutzdaten auszulesen, zunächst die Verwaltungsinformationen gefunden und ausgewertet werden müssen. Das binäre PowerPoint-Format besteht aus einer Baumstruktur von Einträgen. Diese Einträge können wiederum andere Einträge enthalten. Sie sind dann also Ordner oder sie enthalten Daten. Die Information, um welchen Typ Eintrag es sich handelt, lässt sich in den ersten 8 Bytes des Eintrags finden.<sup>70</sup> Da es eine Menge unterschiedlicher Einträge gibt, ist der Aufwand, dieses Format einzulesen, relativ hoch. Hierfür gibt es allerdings das Apache-POI-Projekt, eine Programmierschnittstelle, die sich als Bibliothek in Java einbinden lässt und das Ein- und Auslesen von PPT-Dateien übernimmt. Leider ist Apache POI noch nicht vollständig mit der Java-Version von Android kompatibel. Dies liegt vor allem an der Nutzung des Java-AWT Pakets, welches von Android nicht unterstützt wird.<sup>71</sup>

Als nächstes wird untersucht, ob sich das neue Dateiformat von PowerPoint besser eignet. Microsoft schreibt Folgendes zu dem Open-XML-Dateiformat: „Jede Anwendung, in der XML unterstützt wird, kann auf Daten im neuen Dateiformat zugreifen und diese verarbeiten. Die Anwendung muss nicht Teil des Microsoft Office-Systems und nicht einmal ein Microsoft-Produkt sein. Benutzer können mit Standardtransformationen Daten extrahieren oder für andere Zwecke verwenden.“<sup>72</sup>

Hieraus folgt, dass das neue Open-XML-Format sich wesentlich besser für dieses Projekt eignet als das binäre und nicht offene alte Dateiformat.

„Der neue Dateicontainer baut auf der einfachen, komponentenbasierten und komprimierten ZIP-Dateiformatspezifikation auf. Kern der neuen Office XML-Formate ist die Verwendung von XML-Referenzschemas und eines ZIP-Containers. Jede Datei setzt sich aus einer Auflistung einer beliebigen Anzahl von Komponenten zusammen. Diese Auflistung definiert das Dokument.

Dokumentkomponenten werden mithilfe des ZIP-Formats in der Containerdatei bzw. dem Paket gespeichert. Bei den meisten Komponenten handelt es sich um XML-Dateien, die in der Containerdatei gespeicherte Anwendungsdaten, Metadaten und sogar Kundendaten beschreiben. Im Containerpaket können andere, Nicht-XML-Komponenten vorhanden sein, einschließlich Komponenten wie Binärdateien, die im Dokument eingebettete Bilder oder OLE-Objekte darstellen. Zusätzlich werden durch Beziehungs-komponenten die Beziehungen zwischen Komponenten festgelegt. Dieser Entwurf stellt die Struktur für Office-Dateien dar. Während sich der Inhalt der Datei aus Komponenten zusammensetzt, beschreiben die Beziehungen, wie die einzelnen Komponenten zusammenarbeiten.“<sup>73</sup>

Um Dateien im Open-XML-Format einzulesen, muss die Anwendung also ZIP-Dateien einlesen können. Ob und wie dies in Java funktioniert, wird an späterer Stelle erläutert.

---

<sup>70</sup> Vgl. The Apache Software Foundation (2013).

<sup>71</sup> Vgl. Delap (2007).

<sup>72</sup> Microsoft (2006).

<sup>73</sup> Microsoft (2006).

Wenn eine PowerPoint 2007-Präsentation im PowerPoint XML-Format gespeichert wird, kann auf die Inhalte dieser Präsentation problemlos zugegriffen werden. Da es sich um die erste PowerPoint-Version handelt, die XML-Formate verarbeiten kann, bietet sie viele Anwendungsmöglichkeiten, die in früheren PowerPoint-Versionen nicht verfügbar sind. Sie können jetzt auf Folien und Foliennotizen als Text zugreifen.<sup>74</sup>

Weiterhin ist es für dieses Projekt notwendig, an die Foliennotizen zu gelangen. Da diese als Text in den XML Dateien vorliegen, wird im nächsten Abschnitt darauf eingegangen, wie diese Dateien in Java eingelesen werden können.

#### 8.5.2.4.4 Implementierung

Da sich das Open-XML-Format sehr viel besser eignet als das binäre Dateiformat von PowerPoint, wird nun erläutert, wie sich die gewünschten Informationen in einer Android-Anwendung einlesen lassen. Hierzu wird beim Start der Anwendung zunächst die SD-Karte nach PPTX-Dateien durchsucht.<sup>75</sup>

Um nun an die Informationen zu gelangen, muss zunächst die PowerPoint-Datei als Zip-Archiv eingelesen werden. Java stellt hierfür die Klasse „ZipFile“ bereit, welche zum Erstellen eines Objekts den Dateinamen inklusive Pfad als String benötigt. Mit der Methode „entries()“ werden alle Einträge der Zip-Datei in ein Feld vom Typ „ZipEntry“ eingelesen. Da die Notizen eingelesen werden sollen, ist zunächst von Interesse wie viele Notizen es gibt. Hierfür wird jeder Eintrag daraufhin überprüft, ob er eine Notiz ist.<sup>76</sup>

```
ZipEntry entry = (ZipEntry) entriesIter.nextElement();
if (entry.getName().contains("ppt/notesSlides/notesSlide"))
```

#### Quelltextfragment 3: Datei mittels Java in Zip-Archiv finden.<sup>77</sup>

Die Einträge, die übereinstimmen, sind die gesuchten XML-Dateien und werden in eine Liste aufgenommen. In diesen Dateien sind alle Informationen zu den Foliennotizen gespeichert.

In Abb. 8.14 kann der Aufbau einer XML-Datei für Foliennotizen betrachtet werden. Der nächste Schritt ist die XML-Dateien einzulesen. Dies kann über die Klasse „XML-Reader“ relativ einfach bewerkstelligt werden. Damit diese Klasse jedoch arbeiten kann, benötigt sie zuerst einen Content Handler. Darin wird festgelegt, wie die XML-Datei aufgebaut ist, welche Elemente relevant sind und was mit den Daten relevanter Elemente getan wird. In dem Beispiel aus Abb. 8.12 entspricht jedes Element <a:p> einem Absatz und enthält den Text aus Element <a:t>. Dieser Text kann von dem Content Handler eingelesen und beispielsweise in einem Feld gespeichert werden.

<sup>74</sup> Microsoft ([2006](#)).

<sup>75</sup> Vgl. Althof ([2007](#)).

<sup>76</sup> Vgl. Neward ([2007](#)).

<sup>77</sup> Vgl. Neward ([2007](#)).

**Abb. 8.14** Auszug aus einer PowerPoint-XML-Datei

```

<a:p>
  <a:r>
    <a:rPr lang="de-DE"/>
    <a:t>Text Absatz 1 der Notiz</a:t>
  </a:r>
  <a:endParaRPr/>
</a:p>
<a:p>
  <a:r>
    <a:rPr lang="de-DE" sz="2600"/>
    <a:t>Text Absatz 2 der Notiz</a:t>
  </a:r>
  <a:endParaRPr/>
</a:p>

```

```

ZipEntry notesSlideXML = pptxFile.getEntry(xmlDateiname);
XMLReader xmlReader = XMLReaderFactory.createXMLReader();
InputSource inputSource =
  new InputSource(pptxFile.getInputStream(notesSlideXML));
NotesContentHandler note = new NotesContentHandler();
xmlReader.setContentHandler(note);
xmlReader.parse(inputSource);
allNotesText.add(note.giveArrayList());

```

#### Quelltextfragment 4: XML-Datei aus Zip-Archiv mittels Java einlesen<sup>78</sup>

In diesem Ausschnitt aus dem Quelltext wird eine XML-Datei aus dem Archiv der PowerPoint-Datei extrahiert und als Input Stream dem Konstruktor der XML-Klasse „InputSource“ übergeben. Als nächstes wird ein Content-Handler-Objekt erstellt, welches dem XML-Reader-Objekt übergeben wird. Mittels der Methode „parse(InputSource)“ kann nun das Einlesen gestartet werden.<sup>79</sup>

##### 8.5.2.5 Timer

Um dem Präsentierenden einen besseren Überblick darüber zu geben, wie lange er bereits präsentiert, soll als nächstes ein Timer konzipiert und implementiert werden.

Zunächst wird festgelegt, welche Ereignisse den Timer starten, beenden oder neu starten. Es ist sinnvoll, den Timer zu aktivieren, sobald die Activity zur Präsentationssteuerung gestartet wird, denn ab diesem Zeitpunkt sollte präsentiert werden. Der Timer sollte auf jeden Fall weiterlaufen, auch wenn die Activity pausiert, indem beispielsweise das Display abgeschaltet oder eine andere Anwendung in den Vordergrund kommt. Beendet wird der

<sup>78</sup> Vgl. Roth (2010).

<sup>79</sup> Vgl. Roth (2010).

Timer, indem die Präsentationsactivity beendet oder der Service zur Steuerung geschlossen wird.

Realisiert wird dieser Timer mittels der Methode „scheduleAtFixedRate()“ der Klasse „Timer“. Diese Methode bekommt als Attribute einen Timer Task, die Verzögerung in Millisekunden, bevor der Timer gestartet wird, und das Intervall, mit welchem der Timer wieder aufgerufen wird. Die Methode „run()“ des Objekts Timer Task wird nun im angegebenen Intervall immer wieder aufgerufen.<sup>80</sup>

Da hierdurch ein neuer Thread gestartet wird, Android aber nicht erlaubt, Elemente der Benutzeroberfläche außerhalb des UI-Threads zu verwenden, muss an dieser Stelle noch die Methode „runOnUiThread()“ aufgerufen werden.<sup>81</sup>

```
t.scheduleAtFixedRate(new TimerTask() {  
    @Override  
    public void run() {  
        runOnUiThread(new Runnable() {  
            @Override  
            public void run() {  
                // simple Timer Funktion:  
                // Sekunden zählen und in Minuten / Stunden umwandeln  
            } ); } }, 0,1000);
```

Quelltextfragment 5: Timer-Funktion<sup>82</sup>

### 8.5.2.6 Simulation eines Laserpointers

Präsentationsgeräte verfügen oftmals über einen Laserpointer, um dem Präsentierenden zu helfen, bestimmte Details während der Präsentation hervorzuheben. Da moderne Smartphones oder Tablet-PCs üblicherweise über keinen Laserpointer verfügen, muss, falls eine solche Funktion realisiert werden soll, eine Softwarelösung konzipiert werden. Hierfür gilt es, vor allem zwei Punkte genauer zu analysieren: Die Steuerung des Laserpointers und die Anzeige auf dem Präsentationscomputer.

Da die Funktion des Laserpointers in diesem Projekt nicht realisiert wurde, werden im Folgenden nur Möglichkeiten für eine Umsetzung vorgestellt, ohne dass diese anhand von Quelltexten näher erläutert werden.

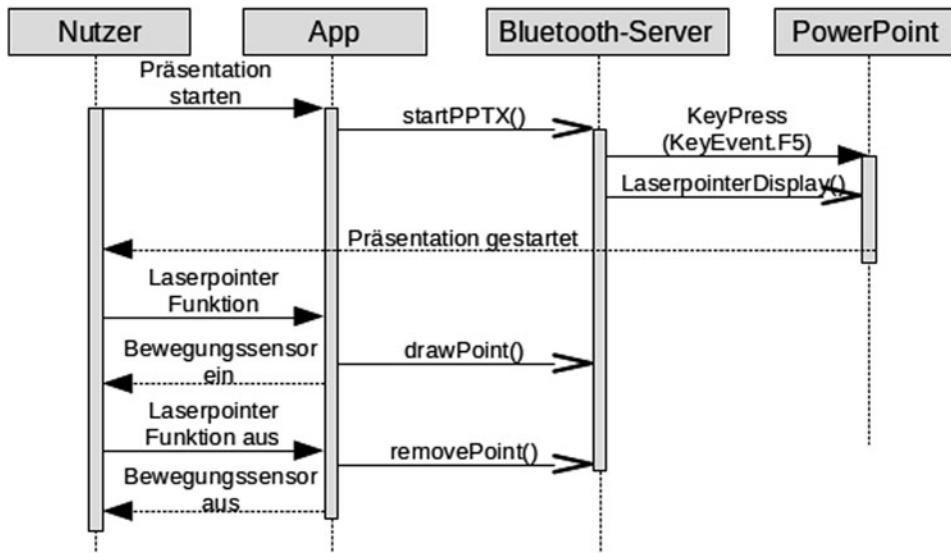
#### 8.5.2.6.1 Steuerung des Laserpointers

Da modernen Smartphones Sensoren zur Verfügung stehen, welche Beschleunigung und Drehung messen, ist es möglich ein Steuerungskonzept zu entwickeln, welches dem eines normalen Laserpointers ähnlich ist.

<sup>80</sup> Vgl. Yiu (2012).

<sup>81</sup> Vgl. Google (2013l).

<sup>82</sup> Vgl. Yiu (2012).



**Abb. 8.15** Sequenzdiagramm – Laserpointer

Alternativ wäre es möglich, den Laserpointer über den Touchscreen zu steuern. Dies würde eine Steuerung ermöglichen, wie sie viele Notebooks zur Bedienung der Maus einsetzen.

Die Steuerung über die Bewegungssensoren ist nach Meinung des Autors jedoch die erste Wahl, da sie intuitiver zu handhaben ist. Voraussetzung hierfür ist jedoch, dass die Sensoren die Bewegungen des Geräts möglichst genau aufnehmen.

Neben der Genauigkeit der Sensoren ist ein weiterer wichtiger Punkt die verfügbare Bandbreite und Latenz der Verbindung zwischen Präsentationscomputer und mobilem Gerät. Ist die Latenz zu hoch oder die Bandbreite nicht ausreichend, um genügend Aktualisierungen pro Sekunde zu ermöglichen, wird die Steuerung möglicherweise als nicht flüssig empfunden.

#### 8.5.2.6.2 Anzeige des Laserpointers auf dem Präsentationscomputer

Um die Anzeige eines Punktes während einer Präsentation auf dem Bildschirm umzusetzen, wäre eine mögliche Lösung die Bluetooth-Server-Software so zu erweitern, dass während der Präsentation eine transparente Anzeige im Vordergrund ist. Diese Anzeige muss sich über den gesamten Bildschirm ausbreiten und zeichnet bei Aktivierung der Laserpointerfunktion einen Punkt, welcher nun über die Android-Anwendung gesteuert werden kann.

Zu beachten ist hierbei, dass die Anzeige zwar im Vordergrund ist, die PowerPoint-Präsentation jedoch den Fokus hat oder zumindest die Eingaben übermittelt bekommt.

Abbildung 8.15 zeigt sehr vereinfacht die Interaktionen zwischen Präsentierendem, Android-Anwendung und Präsentationscomputer. In der Abbildung wird dargestellt,

**Abb. 8.16** Bildschirmaufnahme Activity 1: Verbindungsauftbau



wie zunächst die Präsentation gestartet wird, wodurch sich die Anzeige für den Laserpointer über die PowerPoint-Präsentation legt. Danach wird die Laserpointerfunktion aktiviert und, nachdem ein Punkt erscheint, wieder deaktiviert, wodurch der Punkt wieder verschwindet.

#### 8.5.2.7 User Interface

Die Benutzeroberfläche der Android-Presenter-App ist in vier verschiedene Activities unterteilt. Die Activity aus Abb. 8.16 dient dem Verbindungsauftbau und unterscheidet sich im Aufbau leicht von den anderen Activities. Die Titelleiste ist kleiner und enthält kaum Informationen. Im Gegensatz hierzu befindet sich in der Titelleiste der anderen Activities ein Icon, welches den Status der Verbindung widerspiegelt. Zusätzlich wird diese Information auch als Text dargestellt. Diese große Titelleiste ist erst ab Android 4.0 vorgesehen. Um dies dennoch unter älteren Versionen zu nutzen, gibt es die Bibliothek „ActionBarSherlock“.<sup>83</sup>

Quelltextfragment 6 zeigt, wie die Farbinformationen in einer XML-Datei gespeichert werden. Somit können die Farben aller Activities an einer zentralen Stelle geändert werden.

<sup>83</sup> Vgl. Vogel (2013a).

**Abb. 8.17** Bildschirmaufnahme Activity 2: Präsentationsauswahl



```
<color name="color_dark">#265665</color>
<color name="color_light">#e9efef</color>
<color name="color_light_2">#80A8B1</color>
```

#### Quelltextfragment 6: Farbinformationen aus einer XML-Datei<sup>84</sup>

Nach dem Verbindungsaufbau wählt der Nutzer, wie in Abb. 8.17 zu sehen, die Präsentationsdatei aus.

Die Activity aus Abb. 8.18 ist die Präsentationssteuerung. Sobald diese aktiv ist, läuft der Timer und die Lautstärketasten können zur Steuerung verwendet werden. Diese Activity ist im Vollbildmodus, sodass keine Benachrichtigungen oder Ähnliches angezeigt werden, damit der Präsentierende nicht abgelenkt wird.

Außerdem kann in Abb. 8.16 das Icon der Serviceklasse in der Benachrichtigungsleiste gesehen werden. Wird die Leiste aufgeklappt, kann hierüber die Verbindung beendet werden (siehe Abb. 8.19).

Die Activity aus Abb. 8.20 dient zur Auswahl der Einstiegsfolie, da, wie beispielsweise in Vorlesungen, nicht immer bei Folie Nr. 1 begonnen wird. Hier werden der Folientitel, die erste Zeile der Notizen und die Foliennummer angezeigt.

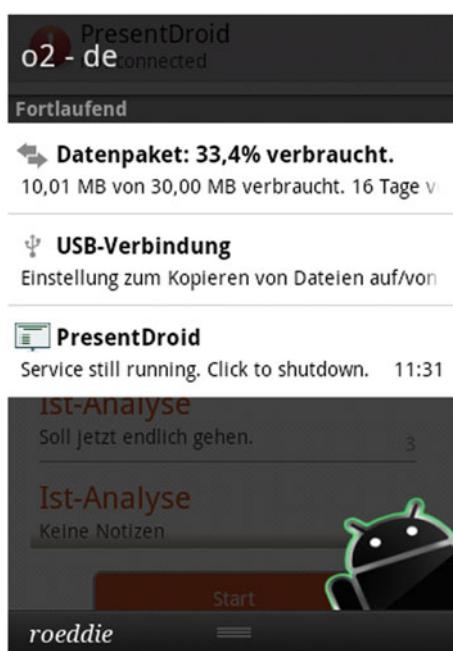
---

<sup>84</sup> Vgl. Vogel (2013b).

**Abb. 8.18** Bildschirmaufnahme Activity 3: Steuerung



**Abb. 8.19** Bildschirmaufnahme der Benachrichtigungsleiste



**Abb. 8.20** Bildschirmaufnahme Activity 4: Folienauswahl



Alle Activities der Anwendung beinhalten sogenannte List Views. Diese können dynamisch erweitert werden. Sie umfassen daher beliebig viele Elemente, durch welche per Wischgeste nach oben oder unten hindurch navigiert werden kann. Die Daten bekommen die List Views über ein Adapterobjekt.<sup>85</sup> Für List Views gibt es fertige Vorlagen im Android-SDK, es können aber auch eigene Vorlagen mit einem anderen Aufbau entworfen werden.<sup>86</sup>

Um die Auswahl einer Activity an die folgende Activity zu übernehmen, kann dem Intentobjekt, welches die neue Activity startet, mit der Methode „putExtra()“ eine Variable und deren Bezeichnung übergeben werden. Durch unterschiedliche Bezeichnungen können somit mehrere Variablen übergeben werden.

Um dem Nutzer einen Verbindungsabbruch anzuzeigen, wird regelmäßig überprüft, ob die Verbindung noch vorhanden ist und falls nicht wird die Titelleiste aktualisiert.

### 8.5.3 Resultat

An dieser Stelle wird nun die fertiggestellte Applikation vorgestellt.

Nach dem Start der Anwendung wird überprüft, ob Bluetooth verfügbar ist. Falls nicht wird eine entsprechende Meldung angezeigt und die Anwendung wird beendet. Ist Blue-

<sup>85</sup> Vgl. Vogel (2013c).

<sup>86</sup> Vgl. Geetha (2010).

tooth verfügbar, wird geprüft, ob Bluetooth eingeschaltet ist. Falls dies nicht der Fall ist, wird der Benutzer aufgefordert Bluetooth einzuschalten. Nun wird eine Liste mit gekoppelten Geräten angezeigt. Hier steht auch der Hinweis, die Anwendung zur Steuerung der Bluetoothverbindung auf dem Präsentationscomputer zu starten. Falls der Präsentationscomputer nicht in der Liste ist, kann hier das Android-Gerät sichtbar gemacht werden. Eine Anleitung zum Koppeln der Geräte erscheint. Nachdem die Geräte gekoppelt sind, erscheint der Rechner nun in der Liste. Nach der Auswahl des Computers wird eine Bluetoothverbindung hergestellt. War die Verbindung erfolgreich, wird ein Symbol in der Benachrichtigungsleiste des Android-Geräts angezeigt.

Als Nächstes wird die SD-Karte nach PowerPoint-Dateien durchsucht. Diese werden in einer Liste angezeigt. Nachdem der Benutzer eine Datei ausgewählt hat, wird diese geöffnet. Dabei wird eine Liste aller Folien angelegt und die zugehörigen Notizen werden eingelesen. Außerdem werden die Animationen, welche per Klick ausgeführt werden, gezählt, damit die Folien auf dem mobilen Gerät und Präsentationscomputer synchron bleiben. In zukünftigen Versionen könnte an dieser Stelle die Datei an den Präsentationsrechner geschickt werden. Außerdem wäre es denkbar, PowerPoint durch die Android-Anwendung starten zu lassen. Hier könnte dann auch direkt die Präsentation geladen und an der richtigen Stelle geöffnet werden. Nach Auswahl der Folie erscheint auf dem Display des Android-Geräts der Titel der aktuellen Folie, die Foliennummer und die Notizen zu dieser Folie. Außerdem gibt es einen Zähler der Animationen bis zum Folienwechsel. Dieser wird im unteren Bereich zusammen mit einem Zeitmesser angezeigt. Im Zeitmesser kann der Benutzer ablesen, wie lange er schon präsentiert.

Die Steuerung der Präsentation erfolgt über die Lautstärketasten. Sie wird aktiviert, sobald die Startfolie ausgewählt ist. An dieser Stelle wird auch die Stand-by-Funktion des Displays deaktiviert, damit während der Präsentation das Android-Gerät aktiv bleibt.

### 8.5.4 Anwendungsgebiete für die Presenter App

**Für wen ist die App gedacht?** Die Idee zur Android-Presenter-App entstand im Hochschulalltag. Immer wieder kommt es vor, dass Präsentationen über die Tastatur des Präsentationscomputers bedient werden müssen, weil kein Presenter verfügbar ist. Da Smartphones mittlerweile Alltagsgegenstände geworden sind, liegt die Überlegung nahe, damit die Präsentation zu steuern.

Der Nutzen der Anwendung beschränkt sich aber nicht nur auf Studenten. Jedoch sollten noch einige Funktionen hinzugefügt werden, bis die App auch im Businessbereich einen Mehrwert gegenüber vorhandenen Presentern bringt.

Im Folgenden wird analysiert, welche Vor- und Nachteile die App gegenüber einem Presenter bringt. Aus den Ergebnissen wird abgeleitet, welche Funktionen noch umzusetzen sind, damit die App auch im Businessbereich Erfolg haben könnte.

## Vorteile gegenüber Presenter

- Wesentlich geringere Kosten
- Zusätzliche Informationen
- Bessere Navigation

## Nachteile

- Momentan ist ein Nachteil der Presenter App, die aufwendigere Installation gegenüber dem Plug-and-play von herkömmlichen Geräten
- Kein Laserpointer

Diese Nachteile lassen sich mit einigen Stunden Arbeitsaufwand beheben. Da die geringeren Kosten keinen wirklichen Mehrwert bieten, wenn die Presenter schon vorhanden sind, sind die zusätzlichen Informationen neben der verbesserten Navigation also der eigentliche Nutzen.

---

## 8.6 Fazit

Die Softwareentwicklung im mobilen Bereich unterscheidet sich zwar in einigen Punkten von der Entwicklung im Desktopbereich, aber wesentliche Punkte bleiben identisch. Hierzu gehören unter anderem die Auswahl eines passenden Vorgehensmodells oder die Entscheidung, für welches Betriebssystem entwickelt werden soll. Interessant für Entwickler ist die Prognose, dass Tablet-PC und Smartphone den Desktop-PC in den Bereichen Kommunikation und Unterhaltung ablösen.<sup>87</sup>

Wird, wie in diesem Projekt, als Zielplattform Android ausgewählt, stehen dem Entwickler umfangreiche Anleitungen und Beispielquelltexte zur Verfügung. Je nach Präferenz kann in Java und XML unter Eclipse oder ohne Programmierkenntnisse mit dem MIT-App-Inventor entwickelt werden. Da der in fast allen Android-Geräten integrierte Play Store zum Vertrieb der Anwendungen zur Verfügung steht, wird dem Entwickler einiges an Arbeit und Kosten bei der Vermarktung abgenommen. Jedoch geht hierfür ein Teil des Umsatzes ab.

Viele Anwendungen werden in einer kostenlosen Version, welche im Funktionsumfang eingeschränkt ist, und einer kostenpflichtigen Version in den Play Store eingestellt. Dies wäre auch für die Präsentationsanwendung aus diesem Projekt denkbar. Die grundlegenden Funktionen, die bereits implementiert wurden, könnten die kostenfreie Version darstellen. Die kostenpflichtige Version könnte dann um Funktionen wie Laserpointer, Dateiversand an Desktop-PC, Speichern der Präsentationen im Internet und ähnlichen

---

<sup>87</sup> Interrogare (2012).

bereits vorgestellten Funktionen erweitert werden. Dies würde gewährleisten, dass auch die kostenlose Version für Studenten im Hochschulalltag einen Nutzen bringt, da die grundlegenden Funktionen eines Presenters abgedeckt werden.

---

## Literatur

- Abelson, H., Spertus, E., Looney, L., Wolber, D.: App Inventor. O'Reilly, Sebastopol (2011)
- Althof, S.: Verzeichnis auslesen in Java. <http://mrfoo.de/archiv/315-Verzeichnis-auslesen-in-Java.html> (2007). Zugegriffen 20. Mai 2013
- Apple Inc. (Hrsg.): iOS – Das fortschrittlichste Betriebssystem der Welt. <http://www.apple.com/de/iphone/ios/> (2013). Zugegriffen 18. Mai 2013
- AVM Computersysteme Vertriebs GmbH (Hrsg.): WLAN und Bluetooth – der Vergleich. [http://www.avm.de/de/News/artikel/newsletter/bluetooth\\_wlan.html](http://www.avm.de/de/News/artikel/newsletter/bluetooth_wlan.html) (2013). Zugegriffen 20. Mai 2013
- Bluetooth SIG, Inc. (Hrsg.): Service discovery. <https://www.bluetooth.org/en-us/specification/assigned-numbers-overview/service-discovery> (2013). Zugegriffen 20. Mai 2013
- Brandt-Pook, H., Kollmeier, R.: Softwareentwicklung kompakt und verständlich. Vieweg + Teubner, Wiesbaden (2008).
- Bundschuh, C.: Betriebssysteme Für Mobile Devices. GRIN, Norderstedt (2011)
- Delap, S.: Google's Android SDK Bypasses Java ME in Favor of Java Lite and Apache Harmony. <http://www.infoq.com/news/2007/11/android-java> (2007). Zugegriffen 20. Mai 2013
- doctronic GmbH & Co. KG (Hrsg.): Mobile Devices vs. PCs 2012, 2013 und darüber hinaus. [http://www.doctronic.de/index.php?id=doctronicnews&tx\\_ttnews%5Btt\\_news%5D=29&cHash=9ad25fc27fe90d4491a4650a2408dc7c](http://www.doctronic.de/index.php?id=doctronicnews&tx_ttnews%5Btt_news%5D=29&cHash=9ad25fc27fe90d4491a4650a2408dc7c). (2013). Zugegriffen 26. April 2013
- Erdle, F.: Android: Die Geschichte des Erfolgs. <http://www.connect.de/ratgeber/android-geschichte-des-erfolgs-1491130.html> (2013). Zugegriffen 18. Mai 2013
- Gargenta, M.: Einführung in die Android-Entwicklung. O'Reilly, Köln (2011)
- Geetha, S.: Custom ListView – Android developer tutorial (Part 17). <http://saigeethamn.blogspot.in/2010/04/custom-listview-android-developer.html>. (2010). Zugegriffen 20. Mai 2013
- Goldmedia Custom Research GmbH (Hrsg.): Mobile Monitor 2012– Geräte- Dienst- Kundenzufriedenheit (Teaser für die Langfassung). [http://www.goldmedia.com/uploads/media/Mobile\\_Monitor\\_2012\\_Teaser\\_Goldmedia.pdf](http://www.goldmedia.com/uploads/media/Mobile_Monitor_2012_Teaser_Goldmedia.pdf). (2013). S. 4. Zugegriffen 18. Mai 2013
- Google Inc. (Hrsg.): Application fundamentals. <http://developer.android.com/guide/components/fundamentals.html> (2013a). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Activities. <http://developer.android.com/guide/components/activities.html> (2013b). Zugegriffen: 20 Mai 2013
- Google Inc. (Hrsg.): Services. <http://developer.android.com/guide/components/services.html> (2013c) Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Intents and intent filters. <http://developer.android.com/guide/components/intents-filters.html> (2013d). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): The AndroidManifest.xml File. <http://developer.android.com/guide/topics/manifest/manifest-intro.html> (2013e). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): UI overview. <http://developer.android.com/guide/topics/ui/overview.html> (2013f). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Get the Android SDK. <http://developer.android.com/sdk/index.html> (2013g). Zugegriffen 20. Mai 2013

- Google Inc. (Hrsg.): Developer tools. <http://developer.android.com/tools/index.html> (2013h). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Android NDK. <http://developer.android.com/tools/sdk/ndk/index.html> (2013i). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): BluetoothAdapter. <http://developer.android.com/reference/android/bluetooth/BluetoothAdapter.html> (2013j). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Bound Services. <http://developer.android.com/guide/components/bound-services.html> (2013k). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Processes and threads. <http://developer.android.com/guide/components/processes-and-threads.html> (2013l). Zugegriffen 20. Mai 2013
- Google Inc. (Hrsg.): Android. <http://www.android.com/about/> (2013). Zugegriffen 18. Mai 2013
- Heise, C., Heise, A., Persson, C. (Hrsg.): Google will das Android-Update-Problem entschärfen. <http://www.heise.de/mobil/meldung/Google-will-das-Android-Update-Problem-entschaerfen-1628129.html> (2012). Zugegriffen 18. Mai 2013
- Henkelmann, C.: Mobile Apps – Nativ vs. Hybrid oder: Wer billig kauft, kauft zwei mal. <http://www.seosweet.de/blog/2013/08/01/mobile-apps-nativ-vs-hybrid-oder-wer-billig-kauft-kauft-zwei-mal/> (2013). Zugegriffen 28. Aug. 2013
- Interrogare GmbH (Hrsg.): Trendstudie 2012: Smartphones und Tablets dominieren die private Mediennutzung – Apps bevorzugt. <http://www.interrogare.de/news/news-detailansicht/trendstudie-2012-smartphones-und-tablets-dominieren-die-private-mediennutzung-apps-bevorzugt/da3452c23f29844a70a7897ffb5feb0c/> (2012). Zugegriffen 20. Mai 2013
- Kersken, S.: Kompendium der Informationstechnik – EDV-Grundlagen, Programmierung, Mediengestaltung. <http://openbook.galileocomputing.de/kit/itkomp11001.htm> (2004). Zugegriffen 20. Mai 2013
- Kluczniok, J.: Android Update-Guide: Was kann welche Version? <http://www.netzwelt.de/news/84963-android-update-guide-welche-version.html> (2012). Zugegriffen 20. Mai 2013
- Massachusetts Institute of Technology (Hrsg.): FAQ – Explore MIT App inventor. <http://appinventor.mit.edu/explore/content/faq.html> (2013). Zugegriffen 20. Mai 2013
- Massachusetts Institute of Technology (Hrsg.): Tutorials – HelloPurr. <http://appinventor.mit.edu/explore/content/hellopurr.html> (2013a). Zugegriffen 21. Mai 2013
- Microsoft Inc. (Hrsg.): Einführung in die Microsoft Office (2007) Open XML-Dateiformate. [http://msdn.microsoft.com/library/office/aa338205\(v=office.12\).aspx](http://msdn.microsoft.com/library/office/aa338205(v=office.12).aspx) (2006). Zugegriffen 20. Mai 2013
- Microsoft Inc. (Hrsg.): Windows phone SDK 8.0. <http://www.microsoft.com/en-us/download/details.aspx?id=35471> (2013). Zugegriffen 20. Mai 2013
- Neward, T.: Using Java to crack Office 2007. <http://www.infoq.com/articles/cracking-office-2007-with-java> (2007). Zugegriffen 20. Mai 2013
- Oracle Inc. (Hrsg.): How to use the system tray. <http://docs.oracle.com/javase/tutorial/uiswing/misc/systemtray.html> (2013). Zugegriffen 20. Mai 2013
- Pakalski, I.: Google verändert Zählsystem, Android 4.1 legt deutlich zu. <http://www.golem.de/news/android-verbreitung-google-veraendert-zaehlsystem-android-4-1-legt-deutlich-zu-1304-98484.html> (2013). Zugegriffen 21. Mai 2013
- Parker, J.: 5 years in: The evolution of the iPhone OS. [http://reviews.cnet.com/8301-19512\\_7-57463858-233/5-years-in-the-evolution-of-the-iphone-os/](http://reviews.cnet.com/8301-19512_7-57463858-233/5-years-in-the-evolution-of-the-iphone-os/) (2012). Zugegriffen 18. Mai 2013
- Post, U.: Android Apps entwickeln. Galileo Press, Bonn (2012)
- Rodewig, K.M., Wagner, C.: Apps entwickeln für iPhone und iPad. [#dodtp856d8bd6-bac7-44c4-ab9a-668b2dd46ef4](http://openbook.galileocomputing.de/apps_entwickeln_fuer_iphone_und_ipad/apps_01_002.html) (2012). Zugegriffen 18. Mai 2013

- Rohles, B.: Entstehung einer mobilen Applikation für mittelständische Unternehmen. <http://jorni.de/wp-content/uploads/2012/05/app-entstehung-media-day-2012.pdf> (2012). Zugegriffen 20. Mai 2013
- Roth, F.: JAVA SAX Parser Beispiel/Tutorial. <http://blog.mynotiz.de/programmieren/java-sax-parser-tutorial-773/> (2010). Zugegriffen 20. Mai 2013
- Rühl, C., Schenkel, T.: Best Practices für die Entwicklung mobiler Unternehmens-Apps. <http://www.heise.de/developer/artikel/Best-Practices-fuer-die-Entwicklung-mobiler-Unternehmens-Apps-1627012.html> (2012). Zugegriffen 20. Mai 2013
- Schaffry, A.: Die Zukunft mobiler Anwendungen. <http://www.computerwoche.de/a/die-zukunft-mobiler-anwendungen/2910477> (2013). Zugegriffen 28. Aug. 2013
- Schnabel, P. (Hrsg.): IEEE 802.11/WLAN-Grundlagen. <http://www.elektronik-kompendium.de/sites/net/0610051.htm> (2013a). Zugegriffen 20. Mai 2013
- Schnabel, P. (Hrsg.): WLAN-Übertragungstechnik. <http://www.elektronik-kompendium.de/sites/net/0907101.htm> (2013b). Zugegriffen 20. Mai 2013
- Sikora, A.: Physikalische Schicht – Frequenzsprungverfahren. [http://www.tecchannel.de/netzwerk/wlan/401459/bluetooth\\_grundlagen\\_herkunft\\_und\\_funktionsweise/index4.html](http://www.tecchannel.de/netzwerk/wlan/401459/bluetooth_grundlagen_herkunft_und_funktionsweise/index4.html) (2008). Zugegriffen 20. Mai 2013
- Strang, T., Lichtenstein, M.: Smart Mobile Apps – Ein Spagat zwischen Stil, Performanz und Benutzerfreundlichkeit. In: Linnhoff-Popien, C. (Hrsg.) Smart Mobile Apps – Mit Business-Apps ins Zeitalter mobiler Geschäftsprozesse. Springer, Heidelberg (2012).
- teltarif.de Onlineverlag GmbH (Hrsg.): Windows Phone: Microsofts Neuanfang im Smartphone-Markt. <http://www.teltarif.de/handy/betriebssysteme/windows-phone-7.html> (2013). Zugegriffen 20. Mai 2013
- Telzerow A. (Hrsg.): Die verschiedenen Arten und Reichweiten von Bluetooth. <http://www.computerbild.de/artikel/cb-Ratgeber-Handy-Alles ueber-Bluetooth-3177119.html> (2008). Zugegriffen 20. Mai 2013
- The Apache Software Foundation (Hrsg.): POI-HSLF – A Guide to the PowerPoint File Format. <http://poi.apache.org/slideshow/ppt-file-format.html> (2013). Zugegriffen 20. Mai 2013
- Theiss, B.: Windows Phone 8 im Überblick. <http://www.connect.de/ratgeber/windows-phone-8-im ueberblick-1457236.html> (2013). Zugegriffen 20. Mai 2013
- Thuy, L. G.: Simple Android an Java Bluetooth Application. <http://luugiathuy.com/2011/02/android-java-bluetooth/> (2011). Zugegriffen 20. Mai 2013
- University of Arizona (Hrsg.): Finding your MAC (Media Access Control) address. <http://security.arizona.edu/macaddress> (2013). Zugegriffen 20. Mai 2013
- Vogel, L.: Using the Android action bar (ActionBar) – tutorial. <http://www.vogella.com/articles/AndroidActionBar/article.html> (2013a). Zugegriffen 20. Mai 2013
- Vogel, L.: Android development tutorial. <http://www.vogella.com/articles/Android/article.html> (2013b). Zugegriffen: 20 Mai 2013
- Vogel, L.: Android ListView – tutorial. <http://www.vogella.com/articles/AndroidListView/article.html> (2013c). Zugegriffen 20. Mai 2013
- Walter, M.: Bluetooth. GRIN, München (2003)
- Yiu, S.: My Android tutorials: How to use Timer and TimerTask – Part 1. <http://writecodeeasy.blogspot.de/2012/08/androidtutorial-timer-p1.html> (2012). Zugegriffen 20. Mai 2013

Dennis Christmann

*Entwicklung eines Ratingmodells /-Tools zur Bewertung von mobilen Applikationen*

---

## Zusammenfassung

Der App-Markt wird immer größer. Um sich bei dieser Vielfalt an Apps heutzutage zurechtzufinden, greifen viele Nutzer auf Tests und Rezensionen zurück. Doch die Bewertungen in den App-Stores sind meist nicht sehr detailliert und somit entsteht der Bedarf an einem ausführlicheren Bewertungsmodell. Der nachfolgende Business Case handelt von der Entwicklung eines Rating-Tools zur Bewertung von mobilen Applikationen. Im ersten Schritt wird das Bewertungsmodell entwickelt, auf dessen Basis im Anschluss das Rating-Tool in Java entwickelt wird.

---

## 9.1 Motivation und Zielsetzung

Der Markt für mobile Applikationen (Apps) boomt und die Anzahl der verfügbaren Apps steigt stetig. Laut einer Studie der App-Analytiker von Distimo waren im Januar 2012 fast 480.000 Apps im App-Store von Apple vorhanden.<sup>1</sup> Zu Beginn des Jahres 2012 hat auch der Google-Play-Store den Meilenstein mit 400.000 Apps erreicht.<sup>2</sup> Alleine in Deutschland

---

<sup>1</sup> Vgl. Chip 2012.

<sup>2</sup> Vgl. Distimo 2012.

sind, laut BITKOM, 2011 rund 962 Mio. Apps heruntergeladen worden. Dies entspricht einem Wachstum von 249 % gegenüber dem Vorjahr.<sup>3</sup>

Doch hohe Downloadzahlen und Popularität bedeuten nicht direkt den Erfolg einer App. Wie Studien aus den vergangenen Jahren zeigen, ist der Lebenszyklus einer App auf einem Smartphone begrenzt. Die Firma Localytics untersuchte 2011 Tausende von Apps. Ihre Untersuchungen ergaben, dass 28 % der Apps, die im Laufe des Jahres 2010 heruntergeladen wurden, von ihren Nutzern nur einmal verwendet worden sind.<sup>4</sup> Der erste Eindruck zählt und demnach kommen mindestens 20 % der Apps nicht über die erste Nutzung hinaus. Doch auch im Falle, dass der erste Eindruck überzeugt, spricht dies nicht für eine viel höhere Lebenserwartung einer App. Die folgende Abb. 9.1 zeigt das Ergebnis einer weiteren Studie von Flurry, damals Pinch Media, aus dem Jahr 2009.

Die Studie zeigt, dass es nur minimale Unterschiede in der Nutzungsdauer von kostenlosen und kostenpflichtigen mobilen Applikationen gibt. Bereits nach wenigen Tagen fällt die Anzahl der wiederkehrenden Nutzer deutlich und die meisten Apps kommen nicht über eine Lebenserwartung von 30 Tagen hinaus.

Demnach scheint es, dass die meisten Nutzer auf Dauer nicht mit den Apps zufrieden sind. Entweder wird die App nur einmal verwendet, da sie z. B. nicht das erfüllt, was von ihr erwartet wurde, oder sie verliert nach kurzer Zeit für den Anwender ihren Nutzen. Zwar gibt es in den App-Stores knappe Rezensionen von Käufern mit einer groben Bewertung, doch diese scheinen nicht sehr aufschlussreich bzw. hilfreich zu sein, da viele Nutzer, laut dieser Studie, schon nach dem ersten Starten nicht mehr überzeugt von der App sind. Das ist bei einer kostenlosen App zu verkraften, bei einer kostenpflichtigen schon weniger. Daher entsteht hier der Bedarf an einem Bewertungsmodell, das mobile Applikationen detaillierter bewertet, als es z. B. in den App-Stores getan wird. Damit könnte sich der potenzielle Nutzer, vor dem Kauf einer App, genau über die einzelnen wichtigen Kriterien informieren und wie die App darin bewertet wird.

Das vorliegende Kapitel beschreibt die Entwicklung eines Rating-Tools für mobile Applikationen. Demnach verfolgt dieses Kapitel im Kern zwei Ziele. Das erste Ziel ist die Entwicklung eines Ratingmodells zur Bewertung von mobilen Applikationen. Daraus resultierend wird anschließend, zum Bewerten der Apps in der Praxis, ein Rating-Tool in der Programmiersprache Java entwickelt, das auf diesem Ratingmodell basiert.

---

## 9.2 Grundlegendes zum Projekt Rating-Tool

In diesem Abschnitt wird Grundlegendes zum Projekt Ratingmodell /-Tool erläutert. Zu Beginn wird das Vorgehensmodell für die Entwicklung des Rating-Tools vorgestellt. Des Weiteren soll in diesem Abschnitt auf die Java Database Connectivity (JDBC) eingegangen werden. Zum Abschluss werden die Tools zum Erstellen des Datenbankservers vorgestellt.

---

<sup>3</sup> Vgl. Bitkom 2012.

<sup>4</sup> Vgl. Localytics 2012.

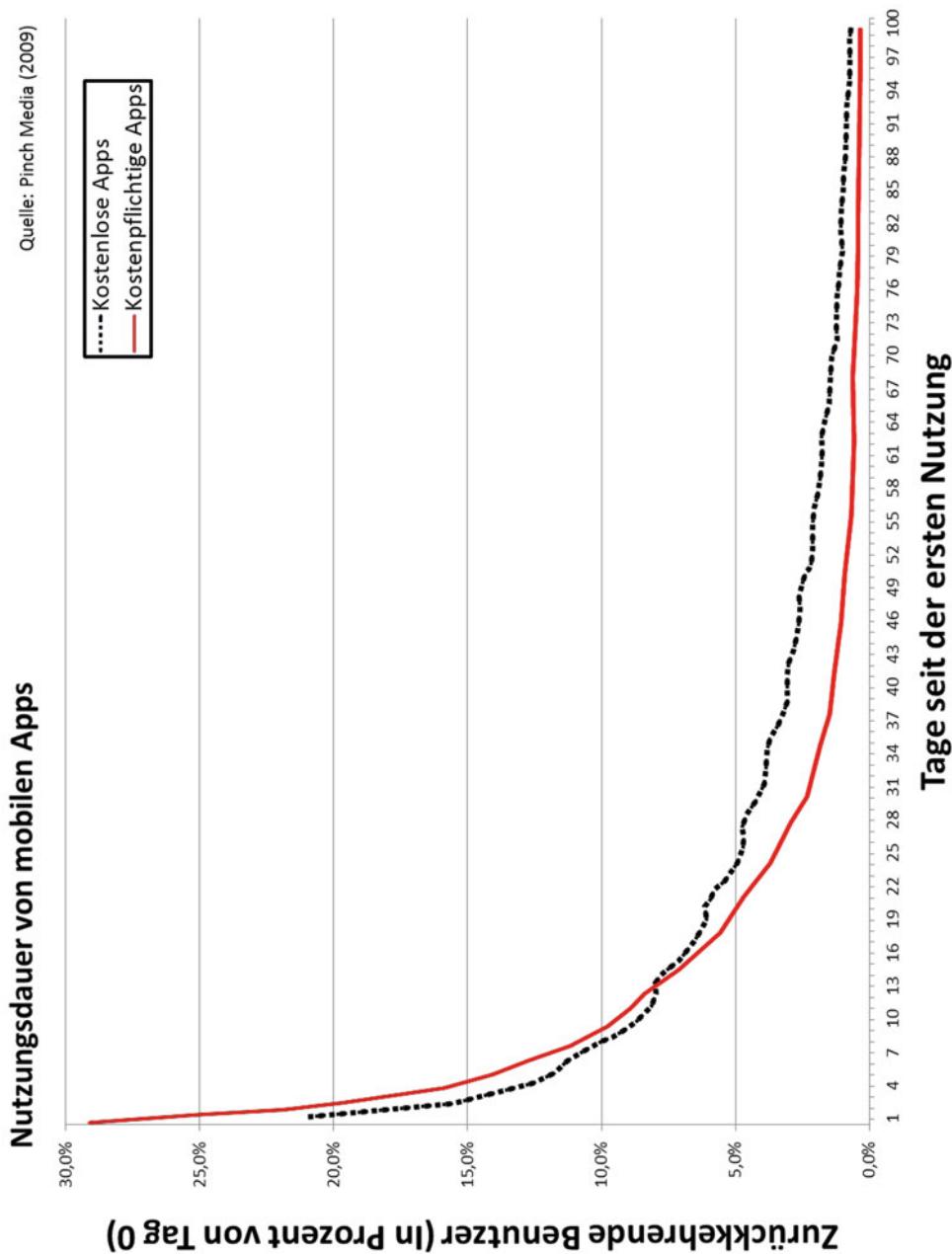
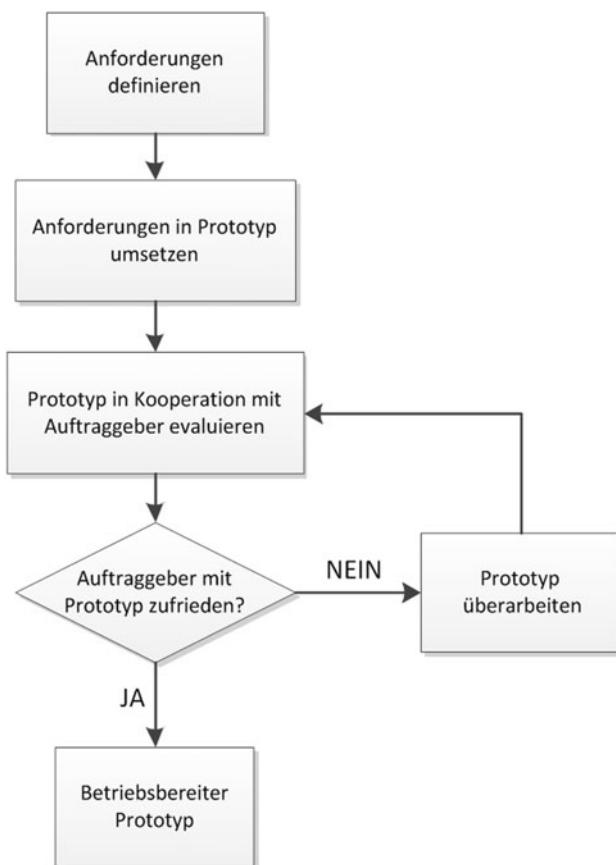


Abb. 9.1 Nutzungsdauer von mobilen Applikationen. (Eigene Erstellung, in Aplehnung an Yardley 2009)

**Abb. 9.2** Das Prototypingmodell



### 9.2.1 Das Vorgehensmodell

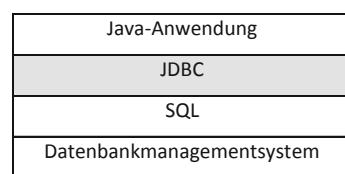
Zu Beginn eines Softwareentwicklungsprozesses ist es wichtig, einen Ablaufplan zu entwerfen. Ein solcher Ablaufplan wird Vorgehensmodell genannt.<sup>5</sup> Im Laufe der Zeit haben sich bestimmte Vorgehensmodelle bewährt, die die Planung und Durchführung einer Softwareentwicklung unterstützen. Im Allgemeinen legen Vorgehensmodelle eine genaue Abfolge von Schritten fest, an denen sich der Entwickler orientieren kann. In der Praxis lässt sich jedoch nicht jedes Vorgehensmodell direkt anwenden, sondern muss an die Situation des Projektes angepasst werden.<sup>6</sup> Für dieses Projekt wurde das Prototypingmodell ausgewählt. Der Ablauf des Prototypingmodells wird in Abb. 9.2 dargestellt.

Zu Beginn werden die vom Auftraggeber gestellten Anforderungen identifiziert. Der Entwickler arbeitet mit dem Auftraggeber zusammen, bis alle grundlegenden Anforde-

<sup>5</sup> Vgl. Lehner et al. 2008, S. 150.

<sup>6</sup> Vgl. Schatten et al. 2010, S. 65.

**Abb. 9.3** Schichtenmodell einer DB-Java-Anwendung.  
(Eigene Erstellung, in Anlehnung an Abts 2010, S. 27)



rungen bekannt sind. Anschließend wird der erste funktionsfähige Prototyp entwickelt, der den definierten Anforderungen entspricht.<sup>7</sup> Danach wird der Prototyp in Zusammenarbeit mit dem Auftraggeber evaluiert, um herauszufinden, ob die Anforderungen erfüllt wurden. Hier können nun Verbesserungs- oder Änderungswünsche geäußert werden. Nachdem eventuelle Änderungswünsche umgesetzt wurden, wird der Prototyp erneut in Kooperation mit dem Auftraggeber evaluiert. Dieser Vorgang wird sich so oft wiederholen, bis der Prototyp an die Bedürfnisse und Vorstellungen des Auftraggebers angepasst und betriebsbereit ist.<sup>8</sup>

## 9.2.2 Java Database Connectivity

Die Java Database Connectivity (JDBC) ist die Standard-Java-Schnittstelle für den Zugriff auf relationale Datenbanken durch SQL. JDBC ist eine Sammlung von Klassen, die im Paket „java.sql“ zusammengestellt sind. Mithilfe dieser Klassen können Verbindungen zu Datenbanken (DB) aufgebaut, SQL-Befehle verschickt und Abfragen im Programm verarbeitet werden. Java-Anwendungen, die durch diese Schnittstelle auf eine Datenbank zugreifen, besitzen keinen datenbankspezifischen Code. Dadurch kann das Datenbanksystem (z. B. MySQL) einfach ausgetauscht werden, ohne das Programm ändern zu müssen. JDBC stellt somit eine neutrale Datenbankschnittstelle zur Verfügung. Abbildung 9.3 zeigt, dass JDBC eine separate Schicht zwischen dem Java-Programm und SQL bildet.<sup>9</sup>

Im Falle des Rating-Tools wird die JDBC im Clientprogramm genutzt, das lokal (oder später über einen Server) auf eine Datenbank zugreift. Ein solcher Aufbau wird zweistufige Architektur genannt, wie Abb. 9.4 zeigt.

Ein Datenbankmanagementsystem (DBMS) besitzt eine eigene vom Hersteller bereitgestellte Zugriffsschnittstelle. Dieser sogenannte JDBC-Treiber übersetzt die JDBC-Befehle in DBMS-Befehle des jeweiligen Datenbankservers. Für jedes DBMS wird ein eigener Treiber eingesetzt.<sup>10</sup> Den Treiber für MySQL, den MySQL Connector, stellt Oracle auf der SQL-Homepage zur Verfügung.<sup>11</sup>

<sup>7</sup> Vgl. Laudon et al. 2010, S. 572.

<sup>8</sup> Vgl. Laudon et al. 2010, S. 572.

<sup>9</sup> Vgl. Abts 2010, S. 27.

<sup>10</sup> Vgl. Abts 2010, S. 27f.

<sup>11</sup> Vgl. Oracle 2013a.

**Abb. 9.4** JDBC-2-Schichten-Architektur. (Eigene Erstellung, in Anlehnung an Abts 2010, S. 27)



### 9.2.3 XAMPP und phpMyAdmin

XAMPP<sup>12</sup> ist eine kostenlos erhältliche Sammlung freier Software. Sie ermöglicht ein einfaches Installieren sowie Konfigurieren des Webservers Apache, der Datenbank MySQL sowie der Sprachen Perl und PHP (Hypertext Preprocessor). Ziel ist es, eine einfache und schnelle Installation zu ermöglichen, da das Aufsetzen und Konfigurieren eines Servers und von Datenbanken recht aufwendig ist. XAMPP ist am besten für Entwickler geeignet, da sie hiermit schnell ein kompaktes Testsystem erhalten.<sup>13</sup> Zusätzlich enthält XAMPP einige nützliche Tools, darunter auch phpMyAdmin.

phpMyAdmin ist ebenfalls eine freie PHP-Anwendung zur Administration von MySQL-Datenbanken. Die Nutzung erfolgt über HTTP (Hypertext Transfer Protocol) mit einem Browser, z. B. Mozilla Firefox. XAMPP und phpMyAdmin werden später im Rahmen der Entwicklung des Rating-Tools verwendet, um einen MySQL-Datenbankserver lokal auf dem Rechner zu erstellen.

<sup>12</sup> X steht für verschiedene Betriebssysteme, A für den Webserver Apache, M für die Datenbank MySQL und P für die Skriptsprachen Perl und PHP.

<sup>13</sup> Vgl. Apache 2013.

## 9.3 Das Bewertungsmodell des Rating-Tools

Nachdem die Grundlagen für die Entwicklung des Ratingmodells /-Tools erläutert sind, kann mit der eigentlichen Aufgabe des Projektes begonnen werden. Dieses Kapitel beschreibt das Bewertungsmodell für mobile Applikationen. Auf diesem Modell wird anschließend das im nächsten Kapitel zu entwickelnde Rating-Tool aufgebaut.

### 9.3.1 Anforderungen an das Modell

Das Bewertungsmodell hat das Ziel, mobile Applikationen aus verschiedenen Kategorien zu bewerten. Ausgehend von der Motivation aus Abschn. 9.1 werden folgende Anforderungen an das Modell gestellt:

- **verschiedene App-Kategorien:**

Mobile Applikationen werden in den App-Stores in verschiedene Bereiche, je nach ihrer Funktion, kategorisiert. Das Bewertungsmodell soll möglichst alle App-Kategorien umfassen, die in App-Stores vorhanden sind und angeboten werden.

- **zwei Bewertungsbereiche:**

Die Bewertung soll sich in zwei Bereiche gliedern. D. h. es soll unterschieden werden zwischen allgemeinen Bewertungskategorien, die für alle Apps gelten, und spezifischen Bewertungskategorien, die je nach App-Kategorie relevante Kriterien bewerten.

- **Sterne-Bewertungssystem**

Es soll ein Sterne-Bewertungssystem (5 Sterne) verwendet werden, um die Apps zu bewerten. Aus den einzelnen Bewertungen der Bewertungskategorien soll ebenfalls eine Durchschnittswertung für die App gebildet werden, z. B. eine Durchschnittswertung in Höhe von 3,5 Sternen. Dies ermöglicht einen direkten Vergleich von Wertungen verschiedener Apps.

### 9.3.2 App-Kategorien

Wie in den Anforderungen festgelegt, soll das Ratingmodell möglichst alle App-Kategorien beinhalten, die in den App-Stores verwendet werden. Die App-Stores von Apple und Google, die aufgrund ihres Erfolgs und ihrer Größe als Maßstab dienen, unterteilen ihre Apps in die, in Abb. 9.5 angeführten Kategorien.

Da einige Kategorien aus den App-Stores die gleichen Apps umfassen und sich nur im Namen unterscheiden, werden diese für das Ratingmodell in passende Kategorien integriert (siehe Abb. 9.6).

Die Kategorie Software-Demos entfällt, da nur fertige Apps in das Modell aufgenommen werden. Demnach wird das Ratingmodell folgende 24 App-Kategorien (siehe Abb. 9.7) enthalten:

<i>Apple Store</i>	<i>Google Play</i>
<ul style="list-style-type: none"> <li>• Bücher</li> <li>• Wirtschaft</li> <li>• Kataloge</li> <li>• Bildung</li> <li>• Unterhaltung</li> <li>• Finanzen</li> <li>• Spiele</li> <li>• Gesundheit und Fitness</li> <li>• Lifestyle</li> <li>• Medizin</li> <li>• Musik</li> <li>• Navigation</li> <li>• Nachrichten</li> <li>• Zeitungskiosk</li> <li>• Foto und Video</li> <li>• Produktivität</li> <li>• Referenz</li> <li>• Soziale Netze</li> <li>• Sport</li> <li>• Reisen</li> <li>• Dienstprogramme</li> <li>• Wetter</li> </ul>	<ul style="list-style-type: none"> <li>• Bücher &amp; Nachschlagewerke</li> <li>• Büro</li> <li>• Comics</li> <li>• Effizienz</li> <li>• Finanzen</li> <li>• Fotografie</li> <li>• Gesundheit &amp; Fitness</li> <li>• Kommunikation</li> <li>• Lernen</li> <li>• Lifestyle</li> <li>• Live-Hintergründe</li> <li>• Medien &amp; Videos</li> <li>• Medizin</li> <li>• Musik &amp; Audio</li> <li>• Nachrichten &amp; Magazine</li> <li>• Personalisierung</li> <li>• Reisen &amp; Lokales</li> <li>• Shopping</li> <li>• Software &amp; Demos</li> <li>• Soziale Netzwerke</li> <li>• Sport</li> <li>• Tools</li> <li>• Unterhaltung</li> <li>• Verkehr</li> <li>• Wetter</li> <li>• Widgets</li> </ul>

**Abb. 9.5** App-Kategorien von Apple und Google

**Abb. 9.6** Zusammenfassung einiger App-Kategorien

<b>Kategorie</b>	<b>Beinhaltet</b>
Referenz & Nachschlagewerke	Kataloge
Bildung	Lernen
Gesundheit	Fitness, Medizin
Nachrichten und Magazine	Zeitungskiosk, Nachrichten
Produktivität	Effizienz
Personalisierung	Live-Hintergründe
Wirtschaft	Büro
Bücher	Comics
Dienstprogramme	Tools

**Abb. 9.7** Die App-Kategorien des Ratingmodells



### 9.3.3 Allgemeine Informationen

Das Ratingmodell soll nicht nur die Bewertungen von Apps ermöglichen bzw. erfassen. Es ist sinnvoll, auch zusätzliche grundlegende Informationen zu den bewerteten Apps aufzunehmen. Daher ist der erste Schritt vor der eigentlichen Bewertung, allgemeine Informationen über die zu testende App anzugeben. Diese Informationen sollen alle für den Nutzer relevanten Daten zu einer App enthalten. Da die entsprechenden Angaben aus den jeweiligen App-Stores entnommen werden, dienen diese ebenfalls der Orientierung. Im Google Play Store und im Apple App-Store werden beispielsweise folgende Angaben gemacht (siehe Abb. 9.8):

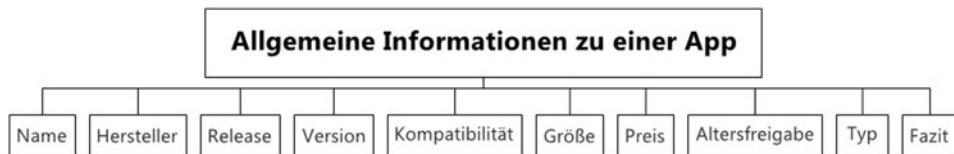
**Abb. 9.8** App-Infos bei Google Play und im Apple-Store. (Google 2012a; Apple 2013)



Wie in Abb. 9.8 einzusehen ist, werden im Google Play Store neben der Kundenbewertung auch Daten wie Erscheinungsdatum, die Versionsnummer, die erforderliche Android-Version, die App-Kategorie sowie die Größe, der Preis und die Inhaltseinstufung angegeben. Im App-Store von Apple wird zusätzlich der Entwickler aufgeführt, der jedoch im Google Play Store auch an anderer Stelle einsehbar ist. Die allgemeinen Informationen dienen dazu, dem potenziellen Käufer grundlegende Informationen zu der App zu liefern. Im Ratingmodell werden dementsprechend folgende Angaben (siehe Abb. 9.9) als allgemeine Informationen zu einer App deklariert.

Bei jedem App-Test müssen grundlegende Angaben gemacht werden:

- der **Name** der App
- der **Hersteller**
- das Erscheinungsdatum (**Release**)



**Abb. 9.9** Allgemeine Informationen des Ratingmodells

**Abb. 9.10** Altersfreigaben im Ratingmodell

Altersfreigabe	Einstufung der App-Stores	
	Apple	Google
Keine Einschränkung	4+	Keine Beschränkung
Niedrige Stufe	9+	Niedrige Stufe
Mittlere Stufe	12+	Mittlere Stufe
Hohe Stufe	17+	Hohe Stufe

- die **Größe** in Megabyte
- die **Versionsnummer**
- der **Preis**
- Die **Kompatibilität** beschreibt die Plattform, auf der eine App betrieben werden kann, z. B. Android oder iOS.
- Ist eine App kostenlos, wird unter Preis dementsprechend „Kostenlos“ eingetragen.

Bezüglich der **Alterseinstufung** gibt es in den App-Stores unterschiedliche Herangehensweisen. Während Google zwischen niedriger, mittlerer und hoher Einstufung des Inhalts unterscheidet<sup>14</sup>, werden bei Apple die Altersstufen „4+“, „9+“, „12+“, „17+“, „No Rating“ verwendet.<sup>15</sup> Um beide Alterskennzeichnungen berücksichtigen zu können, sollen die in Abb. 9.10 dargestellten Altersstufen in diesem Ratingmodell verwendet werden.

Der Punkt „**Typ**“ der allgemeinen Informationen führt noch eine weitere Unterscheidung durch. Hiermit sollen, sofern möglich, die mobilen Apps in ihrer Art der „Geschäftsbeziehung“ klassifiziert werden. Die Auswahlmöglichkeiten sind Folgende:

- Business to Business (B2B)
- Business to Consumer (B2C)
- Business to Employee (B2E)
- Consumer to Consumer (C2C)

Zu den allgemeinen Informationen gehört abschließend ein **Fazit**, welches in Textform wichtige Punkte zur Bewertung der App, z. B. Vor- und Nachteile, zusammenfasst. Der Umfang des Fazits liegt im Ermessen des Anwenders und kann von einem kurzen Satz bis zu einem längeren Text variieren.

<sup>14</sup> Vgl. Google 2012b.

<sup>15</sup> Vgl. Techcrunch 2012.

**Abb. 9.11** Allgemeine Informationen zum „WhatsApp Messenger für Android“

<b>Name</b>	WhatsApp Messenger für Android
<b>Kategorie</b>	Kommunikation
<b>Typ</b>	C2C
<b>Hersteller</b>	WHATSAPP INC.
<b>Release</b>	2013-04-11
<b>Version</b>	2.9.5196
<b>Kompatibilität</b>	ab Android 2.1
<b>Größe</b>	9,7 MB
<b>Preis</b>	Kostenlos
<b>Altersfreigabe</b>	Mittlere Stufe
<b>Fazit</b>	Tolle Messenger App!

Abschließend soll ein Beispiel (siehe Abb. 9.11) gegeben werden, wie die allgemeinen Informationen zur Messenger-App „WhatsApp“ aussehen würden. Die Angaben zu Hersteller, Release, Version, Größe, Preis und Altersfreigabe sind aus dem Google Play Store entnommen.<sup>16</sup>

### 9.3.4 Allgemeine Bewertungskategorien

Die Anforderungen geben vor, dass das Ratingmodell zwei Bewertungsbereiche besitzen soll. Dabei soll der erste Bereich allgemeingültige Kriterien bewerten. Diese Bewertungskategorien bilden den ersten Schritt in der Bewertung einer App. Gemäß den Anforderungen sind daher Bewertungskategorien gesucht, die generell für alle Apps gelten, unabhängig von der jeweiligen App-Kategorie. Abbildung 9.12 zeigt dementsprechend Kriterien, die mobile Apps gemeinsam haben.

Resultierend aus den Kriterien aus Abb. 9.12, ergeben sich fünf allgemeine Bewertungskategorien. Die Kriterien Design und Darstellung des Inhalts werden zu einer Kategorie zusammengefasst. Die fünf Bewertungskategorien sind:

- Design/Darstellung,
- Layout,
- Bedienung,
- Performance und
- Stabilität.

Auf den folgenden Seiten werden diese Kategorien erläutert und es soll eine Orientierung vorgegeben werden, was in den einzelnen Kategorien bewertet werden soll.

Für den ersten Eindruck einer App sind das **Design** und die **Darstellung** ausschlaggebend. Jede App besitzt ihr eigenes Design und ihre eigene Art, den Inhalt darzustellen. Es

<sup>16</sup> Vgl. Google 2012a.



**Abb. 9.12** Gemeinsame Kriterien für Apps

soll geprüft werden, wie die grafische Benutzeroberfläche gestaltet ist. Bei der Bewertung dieser Kategorie sollen vor allem folgende Fragen eine Rolle spielen:

- Wie ist das Design gestaltet? Ansprechend, originell und stilvoll oder nur zweckmäßig?
- Sind die Farben passend zum Thema der App? Spricht das Design und die Darstellung die Zielgruppe an?
- Ist die Schrift gut lesbar?
- Sind alle Elemente des Designs, z. B. Icons und Buttons, selbsterklärend, sodass intuitives Arbeiten ermöglicht wird?

Das **Layout** spielt ebenfalls eine wichtige Rolle im Aufbau einer App. Hier geht es darum, zu bewerten, wie der Inhalt und die einzelnen Screens der App angeordnet sind und wie die Menüführung gestaltet ist. Für einen Nutzer der App ist es wichtig, dass die App übersichtlich ist und alle Informationen auf dem Screen sinnvoll angeordnet sind. Da die Bildschirme von Smartphones in ihrer Größe variieren, ist es wichtig, eine sinnvolle Anordnung zu finden.

- Sind die einzelnen Screens übersichtlich gestaltet und nicht zu überladen?
- Wie schnell findet man sich zurecht?
- Ist das Menü übersichtlich und nachvollziehbar?
- Sind die Symbole, Fenster, Infoboxen, Buttons sinnvoll angeordnet?

Dies sind beispielhafte Fragen, die bei der Bewertung berücksichtigt werden sollen.

Da mobile Applikationen vor allem auf Endgeräten mit Touchscreens laufen, spielt die **Bedienung** eine ebenso wichtige Rolle. Über ein schlechteres Design und ein unübersichtliches Layout kann unter Umständen hinweggeschaut werden. Doch wenn die Bedienung einer mobilen App nicht überzeugt, ist es relativ unwahrscheinlich, dass die App langfristig genutzt wird. In dieser Kategorie, soll folgendes geprüft werden:

- Wie gut funktioniert die Bedienung der einzelnen Funktionen?
- Reagieren alle Buttons ohne Verzögerung? Vor allem sollte hier die Benutzerfreundlichkeit bewertet werden, z. B. ob die Menüführung einfach zu verstehen ist und sich der Nutzer gut zurechtfindet.

- Sind manche Funktionen der App nur umständlich zu erreichen?
- Wie sind die Funktionen in das Handy integriert (z. B. Zugriff auf Kamera)?

Bei Spielen sollte hier vor allem die Umsetzung der Steuerung bewertet werden. Bei Apps, die als Messenger dienen oder bei denen viele Texteingaben erfolgen, ist darauf zu achten, ob z. B. eine QWERTZ-Tastatur<sup>17</sup> angeboten wird und wie gut diese umgesetzt wurde.

Die Kategorie **Performance** bewertet die Geschwindigkeit und Systemauslastung der App. Es sollte auch der Akkuverbrauch überprüft werden. Natürlich verbrauchen aufwendigere Apps im Multimediacbereich mehr Ressourcen und somit mehr Energie. Falls aber z. B. eine einfache „Stoppuhr-App“ viele Ressourcen verbraucht, dann liegt dies an einer ineffizienten Programmierung und muss abgewertet werden. Zentrale Fragen sind hier:

- Wie lange dauert das Starten der App gemessen an ihrer Größe?
- Wie hoch ist der Akkuverbrauch?
- Wie ist die Systemauslastung im Betrieb oder im Hintergrund/Stand-by-Modus?
- Bei Spielen stellt sich die Frage, ob es im Spielablauf Performanceprobleme gibt und ob das Spiel langsamer wird, da z. B. das System überfordert ist.

In der letzten Kategorie **Stabilität** soll bewertet werden, wie stabil die App im Dauerbetrieb oder im Hintergrund läuft. Eine App, die ständig hängen bleibt oder sogar das System zum Absturz bringt, liefert dem Anwender keinen Nutzen. Folgende Fragen sollten bei der Bewertung dieser Kategorie beachtet werden:

- Gibt es Systemhänger?
- Stürzt die Anwendung ab oder friert sogar das ganze System ein?

### 9.3.5 Spezielle Bewertungskategorien

Diese Bewertungskategorien bilden den zweiten Schritt der Bewertung. Sie dienen zur Bewertung der jeweiligen Details einer App. Die speziellen Bewertungskategorien können sich je nach App-Kategorie unterscheiden, da z. B. E-Books oder Spiele nicht mit den gleichen Kriterien bewerten werden können wie Apps aus der Kategorie Kommunikation. Einige App-Kategorien lassen sich unter gleichen Kriterien bewerten, bei anderen sind wiederum eigene Kriterien nötig. In diesem Abschnitt werden die unterschiedlichen speziellen Bewertungskriterien der App-Kategorien vorgestellt.

---

<sup>17</sup> Tastaturbelegung, bei der die sechs Buchstaben der obersten Buchstabenzeile mit den Buchstaben „Q“, „W“, „E“, „R“, „T“ und „Z“ belegt sind.

### 9.3.5.1 Bildung

Diese Kategorie soll Apps umfassen, die zur Bildung bzw. Weiterbildung beitragen. Dazu gehören z. B. Apps wie Vokabeltrainer oder Apps für die Fahrschule. Anwendungen aus dieser Kategorie sollen durch die folgenden Kriterien zusätzlich bewertet werden:

#### 1. Schwierigkeitsgrad

Hier soll grob der Schwierigkeitsgrad angegeben werden, um einzugrenzen für welche Nutzer die App geeignet ist bzw. welcher Wissensstand für die App benötigt wird. Als Auswahlmöglichkeiten werden die Stufen Einsteiger, Fortgeschrittene und Experten vorgegeben.

#### 2. Interaktivität

Durch diese Kategorie soll die Interaktivität der App bewertet werden. Dient die App nur zum „Nachlesen“ von Beispielen oder gibt es auch Übungen, bei denen der Wissensstand des Nutzers in einem Kontext geprüft wird, z. B. das Einsetzen von Wörtern bei einem Vokabeltrainer. Zentrale Fragen sind:

- Wie interagiert der Nutzer mit der App?
- Gibt es Übungen, in denen mit der App interagiert wird, oder dient die App letztendlich nur als Nachschlagewerk zum behandelten Thema?
- Wird der Wissensstand des Nutzers auch durch Beispiele oder Fragerunden geprüft?

#### 3. Feedback

Das Feedback für den Anwender einer Bildungs-App spielt ebenfalls eine wichtige Rolle. Hier soll bewertet werden, in welchem Maße der Nutzer Rückmeldung bekommt, z. B. wie gut sein Wissensstand ist oder wie gut er bei einer Fragerunde geantwortet hat.

#### 4. Nutzwert

Der Nutzwert bewertet den Nutzen der App und wie sinnvoll (langfristig) sie für den Benutzer ist. Zentrale Fragen dazu sind:

- Erfüllt die App auf Dauer ihren Zweck bzw. ihr Ziel?
- Wie hoch ist letztendlich der Nutzen, der aus der Verwendung der App entsteht?

### 9.3.5.2 Bücher

In diese Kategorie fallen Bücher, Comics und E-Books aller Art. Den Inhalt von Büchern zu bewerten ist sehr vielschichtig. Doch da dieses Bewertungsmodell kein Modell zum Bewerten von Büchern sein soll, wird sich in dieser Kategorie auf folgende Angaben beschränkt:

#### 1. Inhalt

Die Bewertung des Inhalts unterscheidet sich je nach Buchtyp. Bei Romanen zum Beispiel kann hier eine grobe Wertung darüber abgegeben werden, wie gut das Buch unterhält. Bei Sachbüchern soll bewertet werden, wie gut der Themenbereich abgedeckt wurde, ob der Text verständlich aufbereitet ist und durchgehend ein roter Faden vorhanden ist.

## 2. Genre & Autor

Wird ein E-Book bewertet, können zusätzlich noch Angaben zum Genre (z. B. Roman oder Sachbuch) und dem Namen des Autors gemacht werden. Diese sind keine direkten Bewertungskategorien, sondern nur weitere Informationen, und gehen somit auch nicht in die Wertung mit ein.

### 9.3.5.3 Nachrichten & Magazine

Diese Kategorie umfasst Apps für Nachrichten, Zeitungen und Magazine. Diese sollen nach folgenden Kriterien bewertet werden.

#### 1. Aufmachung

Das Kriterium „Aufmachung“ bewertet die Art, wie die einzelnen Inhalte bzw. Artikel präsentiert werden:

- Wie werden der Inhalt bzw. die einzelnen Artikel dargestellt?
- Werden moderne Webtechnologien verwendet?
- Werden anderen Medien (z. B. Videos) sinnvoll genutzt?

#### 2. Genre

Auch in dieser Kategorie soll eine Angabe zur Genrerichtung des Magazins oder der Zeitung gegeben werden, z. B. Modemagazin oder Boulevardzeitung.

#### 3. Aktualität

Die Aktualität spielt bei Nachrichten und Zeitungen eine große Rolle. Denn schließlich zieht der Anwender keinen Nutzen aus einer solchen App, wenn es keine regelmäßigen neuen Artikel gibt. Im Grunde stellt sich hier eine zentrale Frage:

- Wie oft werden die Nachrichten, Artikel, Inhalte aktualisiert?

### 9.3.5.4 Referenz & Nachschlagewerke

In die Kategorie „Referenz & Nachschlagewerke“ fallen Apps verschiedener Arten. Diese enthält vor allem Nachschlagewerke, z. B. Duden, Wörterbücher, Lexika oder Kataloge. Zur Bewertung dieser Kategorie sind folgende Bewertungskategorien und Angaben zu verwenden:

#### 1. Thema

Die Angabe des Themas bzw. der Themenrichtung, mit der sich diese App befasst, dient als weitere Informationsquelle für den potenziellen Nutzer.

#### 2. Umfang

Der Umfang einer App dieser Kategorie ist maßgebend für den Nutzen. Daher soll hier bewertet werden, wie umfangreich die Informationen sind, die diese App bietet.

#### 3. Aufmachung

Hier richtet sich die Bewertung, ähnlich wie bei der Kategorie „Nachrichten & Magazine“ (siehe Abschn. 9.3.5.3), auf die Qualität und die Art, wie der Inhalt der App präsentiert wird.

#### 4. Aktualität

Ebenfalls stellt sich bei Nachschlagewerken die Frage, ob ihre Daten aktuell sind oder ob regelmäßig Aktualisierungen, sofern nötig, durchgeführt werden.

##### 9.3.5.5 Spiele

In dieser Kategorie sollen Spiele-Apps nach folgenden Kriterien bewertet werden:

###### 1. Genre

Spiele werden in verschiedene Genres unterteilt. Dementsprechend soll das Genre der Spieleanwendung, z. B. Action, Taktik oder Strategie, ebenfalls angegeben werden.

###### 2. Grafik/Sound

Die Grafik und der Sound sind wichtige Bestandteile eines Spiels. Sie werden beispielsweise dazu verwendet, um Atmosphäre aufzubauen. Folgende Fragen spielen bei der Bewertung eine zentrale Rolle:

- Ist die Grafikgestaltung ansprechend? Für die Zielgruppe geeignet?
- Passen der Sound und die Grafik zum Spielszenario?

###### 3. Spaßfaktor

Der Spaßfaktor stellt im Grunde den Nutzwert der Kategorie Spiele dar. Hier soll der Unterhaltungswert, den das Spiel insgesamt besitzt, bewertet werden.

###### 4. Umfang

Die Spielzeit ist maßgebend für einen dauerhaften Spielspaß. Zentrale Fragen, die bei der Bewertung beachtet werden sollten, sind Folgende:

- Wie lange beträgt die Spielzeit?
- Ist die Spielzeit angemessen für ein kostenloses bzw. kostenpflichtiges Spiel?
- Wie hoch ist der Wiederspielwert?

##### 9.3.5.6 Sonstige App-Kategorien

Bisher wurden spezielle Bewertungskriterien für fünf von den gesamten 24 App-Kategorien definiert. Die restlichen Kategorien sind in Abb. 9.13 dargestellt.

Für diese „sonstigen“ App-Kategorien ist es möglich, gemeinsame App-Kriterien aufzustellen. Die folgende Abb. 9.14 zeigt gemeinsame Kriterien der restlichen 19 App-Kategorien.

Daher sollen Apps, die den Kategorien in Abb. 9.13 angehören, durch die folgenden Kriterien bewertet werden.

###### 1. Funktionsumfang

Beim Funktionsumfang soll generell bewertet werden, wie groß der Umfang der Möglichkeiten ist, den die App bietet. Eine zentrale Frage hierbei ist, erstens ob sie ihren eigentlichen Zweck erfüllt und zweitens ob sie noch zusätzliche sinnvolle Funktionen oder Extras enthält, z. B. Komfortfunktionen oder Möglichkeiten zur Personalisierung.

###### 2. Sicherheit

Sobald Nutzerdaten verwendet, abgespeichert, ausgetauscht oder veröffentlicht werden, spielt die Sicherheit dieser Daten eine wichtige Rolle. Diese Kategorie betrifft vor allem

**Abb. 9.13** Sonstige App-Kategorien

- Dienstprogramme
- Finanzen
- Foto & Video
- Gesundheit
- Kommunikation
- Lifestyle
- Musik
- Navigation
- Personalisierung
- Produktivität
- Reisen
- Shopping
- Soziale Netzwerke
- Sport
- Unterhaltung
- Verkehr
- Wetter
- Widgets
- Wirtschaft

Apps für soziale Netzwerke oder Messenger-Apps. Bei der Bewertung der Sicherheit einer App sollten folgende Fragen beachtet werden:

- Werden beim Nutzen der App Nutzerdaten verwendet, abgespeichert oder ausgetauscht?
- Was passiert mit den Nutzerdaten? Ist es nachvollziehbar?
- Gibt es Einstellungen, mit denen die Privatsphäre geschützt werden kann?
- Gibt es AGB und Datenschutzrichtlinien?

### 3. Werbung

In den meisten Apps ist Werbung integriert. Sei es beim Start, vor jedem Beenden oder auch während der Bedienung kann sie durchgehend präsent sein. In dieser Kategorie soll bewertet werden, wie umfangreich bzw. wie aufdringlich die Werbung ist. Die Werbung soll nie den eigentlichen Inhalt der App überdecken. Es sollte zwischen kostenlos und kostenpflichtig differenziert werden, da bei kostenlosen Apps Werbung akzeptabel ist.

Natürlich hängt es vom Empfinden des Nutzers ab, ob Werbung stört oder nicht, doch es gibt auch Fälle, bei denen klickbare Werbebanner implementiert sind. Falls auf dementsprechende Werbefallen geklickt wird, ist es möglich, dass z. B. ein kostenpflichtiges Abonnement angenommen wird oder Daten gesammelt werden.<sup>18</sup> Zentrale Fragen sind:

- Wie umfangreich ist die eingesetzte Werbung?

---

<sup>18</sup> Vgl. Bleich 2010.



**Abb. 9.14** Kriterien für die restlichen App-Kategorien

**Abb. 9.15** Legende zum Bewertungssystem

Sterne	Bewertung	Note	Rating ( $\emptyset$ )
★★★★★	mangelhaft	5,0	< 1,5 Sterne
★★★★★	ausreichend	4,0	1,5 – 2,4 Sterne
★★★★★	befriedigend	3,0	2,5 – 3,4 Sterne
★★★★★	gut	2,0	3,5 – 4,4 Sterne
★★★★★	sehr gut	1,0	4,5 – 5,0 Sterne

- Wie wird die Werbung eingesetzt? Aufdringlich oder dezent?
- Gibt es Werbefallen?

#### 4. Personalisierung

Diese Bewertungskategorie bewertet die Möglichkeiten, die App nach den Ansprüchen oder Gewohnheiten des Nutzers anzupassen. Die Möglichkeit zum Speichern von Profilen und Einstellungen gehört dazu sowie die Möglichkeit, Teile der visuellen Darstellung der App zu ändern. Zentrale Fragen sind hier:

- Gibt es Möglichkeiten, das Aussehen/den Skin der App oder Teile davon anzupassen/zu verändern?
- Ist es möglich, Benutzerprofile anzulegen oder Präferenzen zu speichern?

#### 5. Nutzwert

Auch bei diesen Apps soll der Nutzen für den Anwender bewertet werden. Dazu werden dieselben Nutzwertkriterien beachtet wie bereits in der Kategorie Bildung, im Abschn. 9.3.5.1.

### 9.3.6 Bewertungssystem

Nachdem die Bewertungskategorien festgelegt wurden, fehlt dem Ratingmodell abschließend ein Bewertungssystem. Wie schon zuvor bei den Anforderungen erwähnt, wurde ein Fünf-Sterne-Bewertungssystem (siehe Abb. 9.15) ausgewählt.

Abbildung 9.15 zeigt die Legende des Bewertungssystems. Dementsprechend wird beispielsweise eine Kategorie, die vier Sterne erhält, als „gut“ bewertet und entspricht einer Note von 2,0.

Im Rating-Tool soll es vereinzelt auch möglich sein, keine Bewertung abzugeben. Kein Stern bedeutet, dass diese Kategorie nicht bewertet ist, denn es ist möglich, dass zum Beispiel eine App nicht in der Kategorie Sicherheit bewertet werden kann. Zum Beispiel ist bei einer einfachen Wecker-App keine Sicherheitsrelevanz vorhanden.

Aus den einzelnen Bewertungen soll auch ein Rating gebildet werden, d. h. wie viele Sterne im Durchschnitt bewertet wurden, damit die Apps miteinander verglichen werden können. Dieses Rating wird später in der Datenbank den allgemeinen Informationen hinzugefügt. Für das Rating wird das arithmetische Mittel aus den einzelnen Bewertungen errechnet. Die Berechnung findet folgendermaßen statt:

Das **arithmetische Mittel** (auch *Durchschnitt*) ist ein Mittelwert, der als Quotient aus der Summe aller beobachteten Werte und der Anzahl der Werte definiert ist.<sup>19</sup> Daher werden alle einzelnen Bewertungen (Sterne) addiert und durch die Anzahl der Bewertungskategorien dividiert. Das Rating könnte dann z. B. einen Wert von 3,25 Sternen ergeben. Laut der Legende zum Bewertungssystem in der Spalte Rating entspricht dies der Bewertung „befriedigend“ und einer Note 3,0.

Das erste Ziel dieses Kapitels ist hiermit erreicht. Es wurde ein Ratingmodell entwickelt, das mobile Applikationen verschiedener Kategorien der App-Stores durch unterschiedliche Kriterien bewerten kann.

Alle Anforderungen, die an das Ratingmodell gestellt wurden, sind erfüllt, da es:

- verschiedene App-Kategorien aus den App-Stores umfasst,
- allgemeine Informationen einer App aufnimmt,
- zwei Bewertungsbereiche besitzt (allgemeine und spezielle Bewertungskategorien) und
- ein Fünf-Sterne-Bewertungssystem besitzt.

---

## 9.4 Das Rating-Tool

Dieser Abschnitt beschreibt die Entwicklung des Rating-Tools in Java. Zu Beginn werden die vorbereitenden Maßnahmen beschrieben, d. h. es werden die Anforderungen an das Tool definiert, sowie das Anlegen der MySQL-Datenbank beschrieben. Anschließend wird die Umsetzung des Rating-Tools im Prototypenprozess (siehe Abschn. 9.2.1) beschrieben und nach Fertigstellung zusammen mit dem Ratingmodell getestet.

### 9.4.1 Anforderungen an das Rating-Tool

Das Rating-Tool basiert auf dem, im vorherigen Abschnitt entworfenen, Bewertungsmodell für mobile Applikationen. Die zukünftigen App-Tests sollen in der Praxis über

---

<sup>19</sup> Vgl. Statista 2012.

das Rating-Tool durchgeführt werden können. Die Ergebnisse der Tests sollen anschließend vom Rating-Tool gespeichert werden. Zum Speichern der Testergebnisse muss daher eine App-Datenbank bestehen. Es wurde sich mit dem Auftraggeber darauf geeinigt, eine MySQL-Datenbank anzulegen. Des Weiteren soll es über das Rating-Tool möglich sein, die Inhalte der Datenbank zu betrachten. Somit werden zusammenfassend folgende Anforderungen an das Tool gestellt:

- Das Anlegen von App-Bewertungen über das Graphical User Interface (GUI) des Rating-Tools, entsprechend dem Bewertungsmodell
- Das Speichern der App-Bewertungen in einer MySQL-Datenbank (lokal)
- Die Möglichkeit zur Einsicht in die Inhalte der App-Datenbank (Tabellen)

#### 9.4.2 Anlegen der App-Datenbank

Bevor die Prototypentwicklung des Rating-Tools beginnt, wird die App-Datenbank mithilfe der Programme XAMPP und phpMyAdmin erstellt. Die Datenbank dient zur Speicherung der Ergebnisse der App-Tests, die über das Rating-Tool durchgeführt werden. Zuerst wird das XAMPP Control Panel gestartet. Über das Control Panel werden per Klick auf die entsprechenden Start-Buttons zuerst der Apache-Server und anschließend die MySQL-Datenbank gestartet. Die Installation und Konfigurierung übernimmt XAMPP. Sobald die Statusanzeige für den Apache-Server und MySQL den Status „Running“ meldet, ist der Server lokal auf dem Rechner erstellt und die MySQL-Datenbank kann verwendet werden. Anschließend kann per Internetbrowser phpMyAdmin über die URL (Uniform Resource Locator) „<http://localhost/phpmyadmin/>“ aufgerufen werden (siehe Abb. 9.16).

Über die Startseite von phpMyAdmin ist es direkt möglich, eine neue Datenbank anzulegen (siehe Abb. 9.17).

Die Datenbank wird als „appdatenbank“ bezeichnet und erzeugt. Automatisch wird nach der Erstellung die nächste Seite angezeigt, auf der die Tabellen der Datenbank angelegt werden. Die erste Tabelle, die erstellt wird (siehe Abb. 9.18), soll die allgemeinen Informationen beinhalten.

Ist die Tabelle erstellt, werden im nächsten Schritt die einzelnen Spalten der Tabelle angelegt und ihre Datentypen sowie deren Zeichlänge definiert (siehe Abb. 9.19).

Die Struktur der Tabelle, die die allgemeinen Informationen enthält, wird in Abb. 9.20 dargestellt.

Nachdem die erste Tabelle angelegt ist, werden auf dieselbe Weise die anderen Tabellen erstellt. Aufgrund der Unterschiede zwischen den App-Kategorien ist es nicht sinnvoll, alle Angaben in eine Tabelle zu integrieren. Daher wird die App-Datenbank folgendermaßen aufgebaut sein:

Für die allgemeinen Informationen werden, ebenso wie für die Kategorien Bildung, Bücher, Spiele, Nachrichtung & Magazine, Referenz & Nachschlagewerke, jeweils eigene Tabellen gebildet. Die restlichen Kategorien, die dieselben Kriterien besitzen, werden

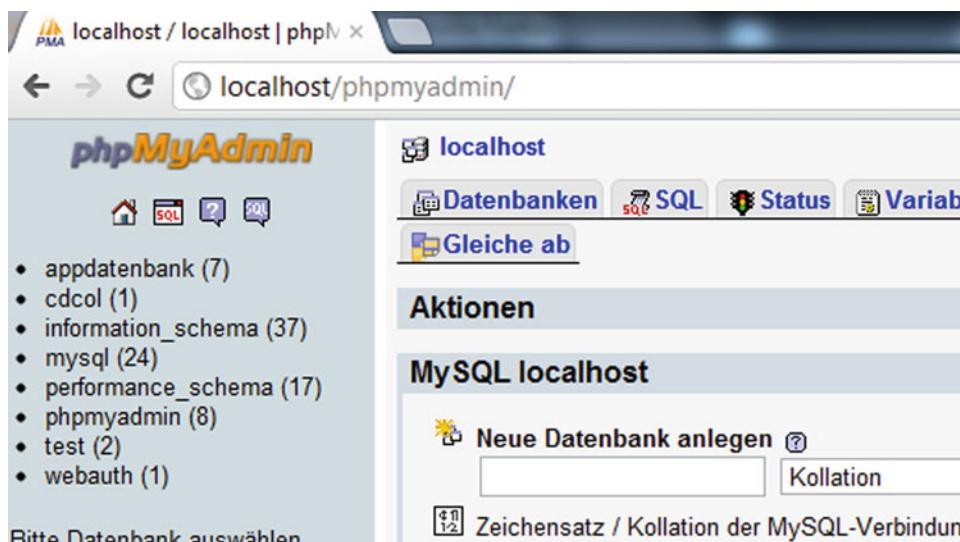


Abb. 9.16 Aufrufen von phpMyAdmin

Abb. 9.17 Anlegen der App-Datenbank

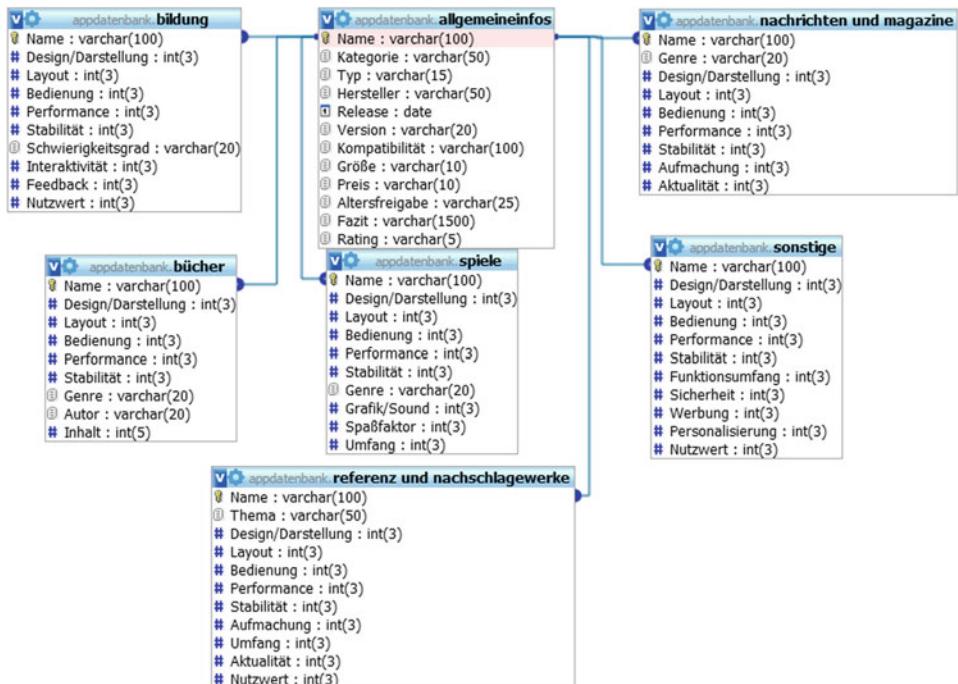
Abb. 9.18 Anlegen einer Tabelle

Feld	Typ	Länge/Set <sup>1</sup>
Name	VARCHAR	100

Abb. 9.19 Anlegen der Spalte „Name“

in der Tabelle „sonstige“ gespeichert. Der Name einer App wird als Primärschlüssel verwendet, um die einzelnen Apps eindeutig zu identifizieren. Die jeweiligen Angaben und Bewertungskategorien bilden die einzelnen Spalten bzw. Attribute. Die Spalte Rating der Tabelle „allgemeineinfos“, ist der Platzhalter für die Durchschnittswertung einer App (siehe Abb. 9.21).

	Feld	Typ	Kollation	Attribute	Null	Standard	Extra
<input type="checkbox"/>	<b>Name</b>	varchar(100)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Kategorie</b>	varchar(50)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Typ</b>	varchar(15)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Hersteller</b>	varchar(50)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Release</b>	date			Nein	Kein	
<input type="checkbox"/>	<b>Version</b>	varchar(20)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Kompatibilität</b>	varchar(100)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Größe</b>	varchar(10)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Preis</b>	varchar(10)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Altersfreigabe</b>	varchar(25)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Fazit</b>	varchar(1500)	latin1_swedish_ci		Nein	Kein	
<input type="checkbox"/>	<b>Rating</b>	varchar(5)	latin1_swedish_ci		Nein	Kein	

**Abb. 9.20** Struktur einer Tabelle**Abb. 9.21** Die App-Datenbank

Die Abb. 9.21 zeigt den finalen Aufbau und die Struktur der App-Datenbank in den einzelnen Tabellen. Untereinander sind sie durch den Primärschlüssel „Name“ verknüpft. Die Struktur der Tabellen basiert auf den Angaben und Bewertungskategorien des Ratingmodells. Somit ist die Datenbank angelegt und kann mit Daten gefüllt werden. Dazu wird das Rating-Tool verwendet werden. Dessen Entwicklung und Test wird für den verbleibenden Teil dieses Kapitels den Schwerpunkt bilden.

### 9.4.3 Realisierung des Rating-Tools

Der Prototypentwicklungsprozess beginnt mit der Definition der Anforderungen. Dieser Schritt wurde in Abschn. 9.4.1 bereits durchgeführt. Im nächsten Schritt werden diese Anforderungen in einem ersten Prototyp umgesetzt. Dieser Abschnitt handelt von der Realisierung des Rating-Tools in der Programmiersprache Java. Es wird nicht auf alle Einzelheiten der entwickelten Klassen und Methoden eingegangen, sondern nur auf die wichtigsten Elemente und Abläufe.

#### 9.4.3.1 Entwicklung des Prototyps

Der erste Schritt in der Entwicklung des ersten Prototyps ist das Designen des GUI (engl. Graphical User Interface). Über diese Benutzeroberfläche muss es möglich sein, die für die App-Bewertung nötigen Informationen einzugeben, damit diese an die MySQL-Datenbank weitergegeben werden können. Des Weiteren soll das Tool den Inhalt der Datenbank bzw. der Tabellen anzeigen können. Zum Erstellen einer grafischen Benutzeroberfläche bietet Java mit der „Swing“-Klasse eine Bibliothek zur Programmierung von GUIs.<sup>20</sup> Zum Erstellen des GUI wird der Netbeans GUI Builder verwendet. Mit ihm ist es möglich, Benutzeroberflächen auf einem Designfeld, per „Drag-and-drop“ der Elemente (z. B. Buttons, Textfelder), zu erstellen. Der Code wird automatisch erstellt.<sup>21</sup>

##### 9.4.3.1.1 Das Graphical User Interface des Rating-Tools

Die Hauptaufgabe des Rating-Tools ist das Bewerten der Apps. Darum besitzt die Startseite bzw. Hauptseite der Benutzeroberfläche des Rating-Tools die Elemente, um die Bewertung der Apps durchzuführen. Dieses Hauptfenster (siehe Abb. 9.22) wurde, orientiert am Bewertungsmodell, in drei Bereiche (Panels) gegliedert.

Auf dem oberen Panel befinden sich die Java-Swing-Elemente, „JTextfields“ und „JComboBoxen“, um die allgemeinen Informationen anzugeben. „JTextfields“ sind Textfelder, in denen die Angaben eingetragen werden können.<sup>22</sup> Eine „JCombo-Box“ ist ein Drop-down-Auswahlmenü, über die z. B. die App-Kategorie ausgewählt werden kann.<sup>23</sup>

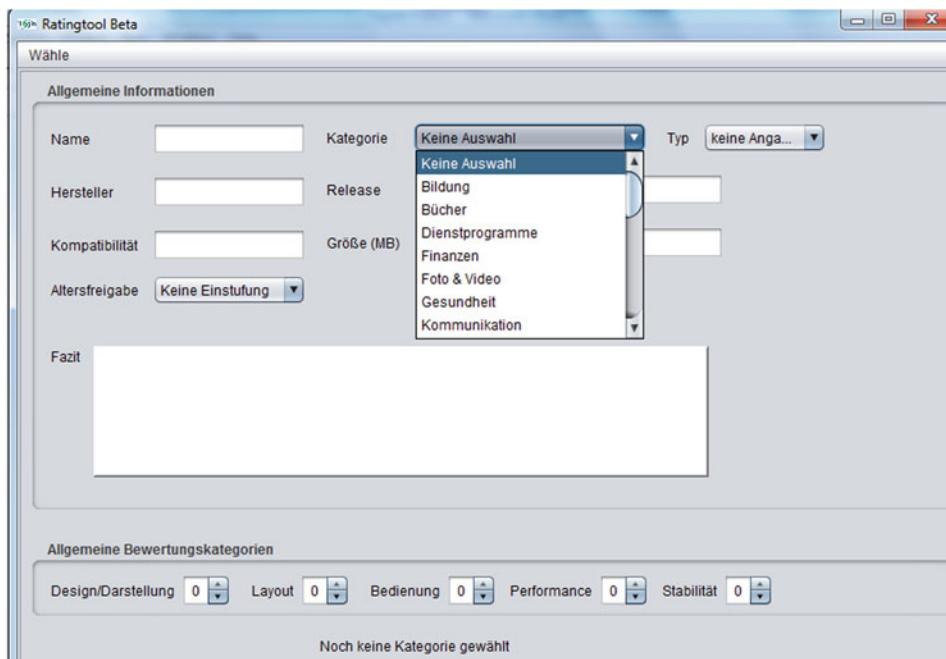
---

<sup>20</sup> Vgl. Oracle 2013b.

<sup>21</sup> Vgl. Oracle 2013c.

<sup>22</sup> Vgl. Oracle 2013d.

<sup>23</sup> Vgl. Oracle 2013e.



**Abb. 9.22** GUI des Rating-Tool-Prototyps

**Abb. 9.23** Panel der Kategorie Spiele

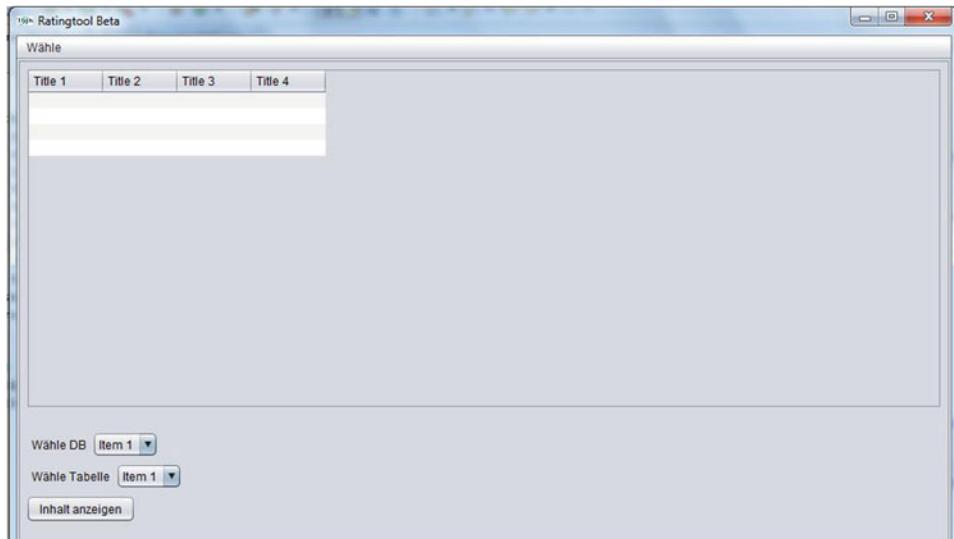


Im mittleren Bereich können die Bewertungen zu den allgemeinen Bewertungskategorien abgegeben werden. Hierfür werden „JSpinner“ eingesetzt. Das sind Eingabefelder mit zwei kleinen Pfeilen, um die Werte innerhalb des Feldes zu verändern.<sup>24</sup> Die Zählwerte der Felder werden, entsprechend des Bewertungssystems, auf null bis fünf „Sterne“ festgelegt. Das untere Panel für die speziellen Bewertungskategorien soll sich, je nach Kategorie, die im oberen Panel ausgewählt wurde, dynamisch ändern. Für jede Kategorie wird ein eigenes Panel entworfen. Die „sonstigen“-Kategorien nutzen dasselbe Panel. Je nach Kategorie ändert sich jedoch die Beschriftung des Panels (siehe Abb. 9.23).

<sup>24</sup> Vgl. Oracle 2013f.

```
public static String getNameField() {return NameField.getText();}
```

**Abb. 9.24** Die Methode „getNameField“



**Abb. 9.25** Anzeige der Tabellen

Hier wird zur Abgabe der Bewertung ebenfalls „JSpinner“ verwendet und je nach Kategorie weitere Auswahlmenüs oder Textfelder. Auf diesen Panels befindet sich jeweils ein Button zum Speichern bzw. Absenden der Daten an die Datenbank.

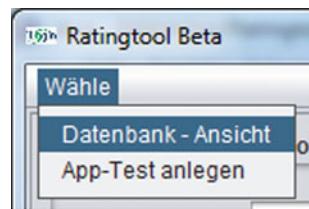
Selbst implementiert werden die „Getter“ für die Elemente. Das sind Zugriffsmethoden für jedes einzelne Textfeld, Auswahlmenü und „JSpinner“, die die dort eingetragenen Werte zurückliefern. Damit können die Nutzereingaben innerhalb des Programms verwendet werden, um die SQL-Befehle zu generieren. Die „Getter“ sind alle gleich aufgebaut. Der „Getter“ für das Textfeld, in dem der Name der App eingegeben wird, wurde z. B. wie in Abb. 9.24 dargestellt, implementiert.

Die Abb. 9.25 zeigt das Design des zweiten Fensters des Rating-Tools, das den Inhalt der Tabellen darstellen soll.

Über die Auswahlmenüs im unteren Bereich wird zuerst die Datenbank, dann die Tabelle ausgewählt, die angezeigt werden soll. Bei Klick auf den Button „Inhalt anzeigen“ soll dann der Inhalt der ausgewählten Tabelle im oberen Bereich in einer „JTable“ (Tabelle) angezeigt werden.<sup>25</sup> Um zwischen den einzelnen Fenstern der Oberfläche zu wechseln, dient der Punkt „Wähle“ in der Menüleiste am oberen Rand (siehe Abb. 9.26).

<sup>25</sup> Vgl. Oracle 2013g.

**Abb. 9.26** Wechseln zwischen den Fenstern



**Abb. 9.27** Die Klasse MySQLConnection

MySQLConnection	
<u>con : Connection</u>	
<u>conStatus : boolean</u>	
<u>+connect() : void</u>	
<u>+disconnect() : void</u>	
<u>+sendQuerys(SQLString1 : String,SQLString2 : String) : void</u>	

**Abb. 9.28** Aufbau der URL

`jdbc:<Protokoll>:<Datenbank-URL>`

#### 9.4.3.1.2 Verbindung zur Datenbank

Nachdem die GUI entworfen ist, kann mit der Implementierung der Funktionen begonnen werden. Bevor Datensätze in die Datenbank eingefügt werden können, muss zuerst eine Verbindung zur Datenbank hergestellt werden. Hier kommt die JDBC und der MySQL-Treiber, die in Abschn. 9.2.2 erwähnt wurden, zum Einsatz. Die Verbindung zur Datenbank wird über die Java-Klasse „MySQLConnection“ geregelt (siehe Abb. 9.27). Sie besitzt die Methoden, um die Verbindung zur Datenbank herzustellen, zu beenden und SQL-Befehle zu versenden.

Über die Methode „connect“ wird die Verbindung zur Datenbank hergestellt. Dazu muss zuerst der JDBC-Treiber für MySQL geladen werden. Anschließend wird die Verbindung zur Datenbank durch Aufruf der Methode „DriverManager.getConnection“ hergestellt. Dieser Methode werden drei Parameter übergeben: die URL der Datenbank, der Benutzername und das Passwort. Die URL ist wie in Abb. 9.28 ersichtlich, aufgebaut.

In diesem Falle wird als Protokoll „MySQL“ für die URL „LocalHost“ angegeben, da der Server lokal auf dem Rechner läuft. Für die Datenbank wurde über phpMyAdmin ein Benutzer angelegt, der die entsprechenden Zugriffsrechte besitzt. Der Benutzername und das jeweilige Passwort müssen ebenfalls angeben sein, damit eine Verbindung zur App-Datenbank hergestellt werden kann (siehe Abb. 9.29).

Die Methode „sendQuerys“ ermöglicht das Senden von SQL-Befehlen an die Datenbank. Es werden immer zwei SQL-Befehle verschickt. Der Erste beinhaltet die allgemeinen Informationen, der Zweite enthält die Bewertungen. Daher besitzt die Methode zwei Parameter, die beim Aufruf übergeben werden müssen (siehe Abb. 9.30).

Zuerst wird überprüft, ob eine Verbindung besteht. Dann wird ein Statement-Objekt erzeugt, das benötigt wird, um die SQL-Befehle zu versenden. Die String-Parameter, mit

```
con = DriverManager.getConnection( "jdbc:mysql://localhost/AppDatenbank", "DC", "test" );
```

**Abb. 9.29** Herstellen der Verbindung

```
public static void sendQuerys(String SQLString1, String SQLString2) {
    try {
        if (conStatus != false){
            Statement stmt = con.createStatement();
            stmt.executeUpdate(SQLString1);
            stmt.executeUpdate(SQLString2);
    }
}
```

**Abb. 9.30** Senden der SQL-Abfragen

```
public static void disconnect() throws SQLException {

    if (con != null)
        con.close();
}
```

**Abb. 9.31** Schließen der Verbindung

denen diese Methode aufgerufen wird, enthalten die SQL-Befehle. Durch Aufruf von „stmt.executeUpdate“ werden beide SQL-Befehle nacheinander über das Statement-Objekt per JDBC an die Datenbank geschickt.

Die Methode „disconnect“ (siehe Abb. 9.31) beendet die Verbindung mit der Datenbank durch den Aufruf von „con.close()“.

#### 9.4.3.1.3 Wechseln der GUI-Fenster und Panels

Das entwickelte Ratingmodell bietet für die App-Kategorien unterschiedliche Bewertungskriterien an. Daher muss sich, bei Auswahl einer Kategorie, das Panel für die speziellen Wertungskategorien anpassen, damit dem Nutzer die passenden Kategorien angezeigt werden.

Diese Aufgabe erledigt die Klasse „PanelSwitch“ (siehe Abb. 9.32) mit ihrer Methode „changePanel“. Falls im Auswahlmenü für die App-Kategorien ein Element geändert wird, werden von der Java-Runtime Events (Ereignisse) gemeldet. Eines dieser Events meldet das ausgewählte Element in der Liste. Dieses Events („ItemEvent“) wird sich hier zunutze gemacht. Wird ein Event durch die Auswahl einer Kategorie ausgelöst, überprüft die Klasse das ausgewählte Item („getSelectedItem“).

Wie im Programmcode in Abb. 9.33 zu erkennen ist, wird mit mehreren If-Schleifen überprüft, welche Kategorie ausgewählt wurde, um das passende Panel aufzurufen. Dazu wird das Card-Layout genutzt. Hier werden die verschiedenen Panels wie Spielkarten übereinander gelegt, jedoch ist immer nur eines von ihnen sichtbar bzw. oben auf dem „Karten-Stapel“.<sup>26</sup> In einem Card-Layout wird jedem der Panels ein Name zugeordnet,

---

<sup>26</sup> Vgl. Oracle 2013h.

**Abb. 9.32** Die Klasse „PanelSwitch“

PanelSwitch
+changeToPanel1() : void
+changeToPanel2() : void
+changePanel() : void

z. B. „SpielePanel“, mit dem es durch „cl.show“ aufgerufen werden kann. Dieses Card-Layout wird ebenfalls zum Wechseln zwischen der Datenbankansicht und dem Fenster zum Anlegen der App-Bewertung verwendet. Die beiden Methoden „changeToPanel1“ (App-Test) und „changeToPanel2“ (Datenbankansicht) erledigen das Aufrufen der Fenster, je nachdem welcher Button in der Menüleiste (siehe Abb. 9.26) geklickt wurde. Der Wechsel wird auch hier durch Aufruf von „cl.show“ und der Angabe des jeweiligen Namens des Fensters, z. B. „DBViewPanel“ für die Datenbankansicht, durchgeführt.

#### 9.4.3.1.4 Generieren der SQL-Befehle

Die über die GUI angegebenen App-Daten sollen per SQL an die Datenbank weitergeleitet werden. Eine Methode zum Versenden von SQL-Befehlen wurde über die Klasse „MySQLConnection“ bereits implementiert. Bevor jedoch die SQL-Befehle versendet werden können, müssen diese anhand der eingegebenen Daten erst generiert werden. Dazu dient die Klasse „createQuerys“ (siehe Abb. 9.34).

Da sich die einzelnen SQL-Befehle für die Kategorien unterscheiden, wurden sechs Methoden implementiert. D. h. es wurden jeweils eine eigene Methode für die Kategorien Bildung, Bücher, Spiele, Nachrichten & Magazine, Referenz & Nachschlagewerke und die sonstigen Kategorien erstellt. Die Methoden sind in ihrem Aufbau grundsätzlich baugleich. Im Folgenden wird der Aufbau am Beispiel der Kategorie Bildung erklärt (siehe Abb. 9.35).

Im Code der Abb. 9.35 werden zuerst jeweils zwei String-Variablen initialisiert. Die Erste wird den SQL-Befehl zum Speichern der Daten in die Tabelle „allgemeineinfos“ enthalten, die Zweite den INSERT-Befehl, der für das Einfügen der Eingabedaten in die Tabelle „Bildung“ sorgt. Anschließend wird, wie in Abb. 9.36 einzusehen, der erste SQL-INSERT-Befehl generiert.

Beim Erstellen des Befehls kommen die „Getter“ der GUI-Elemente zum Tragen. Diese liefern die benötigten Werte (VALUES) um den SQL-Befehl zu vervollständigen. An letzter Stelle des ersten SQL-Befehls wird auch das **Rating** als Wert geliefert. Dieses wird anhand der vorhandenen Bewertungen berechnet. Die Implementierung der Berechnung folgt im weiteren Verlauf (siehe Abschn. 9.4.3.1.5).

Abbildung 9.37 zeigt den zweiten SQL-INSERT-Befehl. Er speichert die einzelnen Bewertungen in die Tabelle „Bildung“ ab. Auch hier liefern die „Getter“ der GUI-Elemente die einzelnen Werte zurück, um den Befehl zu bilden. Zum Abschluss werden die beiden SQL-Befehle („updateString1“, „updateString2“) durch Aufruf der „sendQuerys“-Methode (siehe Abb. 9.30) an die Datenbank verschickt.

```

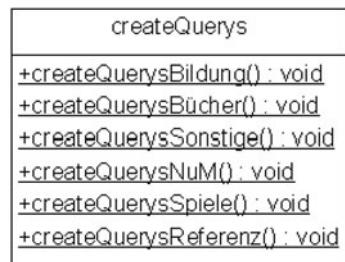
public static void changePanel(ItemEvent e){
    CardLayout cl = (CardLayout) (RatingtoolGUI.SpezBewKatPanel.getLayout());
    JComboBox selectedChoice = (JComboBox) e.getSource();

    if (selectedChoice.getSelectedItem().equals("Spiele")) {
        System.out.println("Spiele");
        cl.show(RatingtoolGUI.SpezBewKatPanel, "SpielePanel");
    }
    else if (selectedChoice.getSelectedItem().equals("Bildung")) {
        System.out.println("Bildung");
        cl.show(RatingtoolGUI.SpezBewKatPanel, "BildungPanel");
    }
}

```

**Abb. 9.33** Auszug aus der „changePanel“-Methode

**Abb. 9.34** Die Klasse „createQuerys“



**Abb. 9.35** Initialisieren der Parameter

```

public static void createQuerysBildung(){
    String updateString1;
    String updateString2;
}

```

#### 9.4.3.1.5 Berechnung des Ratings

Wie bereits erwähnt, soll zu jeder App auch eine Durchschnittswertung gebildet werden, damit sie mit anderen Apps verglichen werden kann. Dieses Rating wird in der Datenbank am Ende der Tabelle „allgemeineinfos“ gespeichert. Die Klasse „calculateRating“ (siehe Abb. 9.38) wurde implementiert, um die Durchschnittswertung zu berechnen und für die SQL-Befehle zurückzuliefern. Das Rating wird dann, wie im vorherigen Abschnitt erwähnt, in den SQL-Befehl integriert (siehe Abb. 9.36).

Die ersten fünf Methoden aus dem Klassendiagramm gelten für die Kategorien Bildung, Bücher, Nachrichten & Magazine, Spiele, Referenz & Nachschlagewerke sowie für die sonstigen Kategorien. Die Methode „rating“ führt die Berechnung der Durchschnittswertung aus. Am Beispiel der „sonstigen Kategorien“ soll die Berechnung des Ratings veranschaulicht werden.

Im Codeauszug in Abb. 9.39 wird zuerst eine Integer-Variable namens „divisor“ deklariert. Dieser entspricht der Anzahl der Bewertungskriterien der jeweiligen Kategorie. Die sonstigen Kategorien werden durch zehn Kriterien bewertet, fünf allgemeine und fünf spezielle. Danach werden über die „Getter“-Methoden alle Bewertungen geliefert, addiert

```

updateString1 = "INSERT INTO `appdatenbank`.`allgemeineinfos` " +
    "(`Name`, `Kategorie`, `Typ`, `Hersteller`, `Release`, `Version`, " +
    "`Kompatibilität`, `Größe`, `Preis`, `Altersfreigabe`, `Fazit`, `Rating`)" +
    "VALUES ('" +
    +RatingtoolGUI.getNameField()+"', '"+ +
    RatingtoolGUI.getCategoryChooser().toString()+"', '"+ +
    RatingtoolGUI.TypAuswahl().toString()+"', '"+ +
    RatingtoolGUI.getHerstellerField()+"', '"+ +
    RatingtoolGUI.getReleaseField()+"', '"+ +
    RatingtoolGUI.getVersionField()+"', '"+ +
    RatingtoolGUI.getKompaField()+"', '"+ +
    RatingtoolGUI.getSizeField)+"', '"+ +
    RatingtoolGUI.getPreisField)+"', '"+ +
    RatingtoolGUI.getAgeChooser().toString()+"', '"+ +
    RatingtoolGUI.getFazitField)+"', '"+ +
    calculateRating.RatingBildung()+"')";
```

**Abb. 9.36** Generieren des ersten SQL-Befehls

```

updateString2= "INSERT INTO `appdatenbank`.`bildung`" +
    "(`Name`, `Design/Darstellung`, `Layout`, `Bedienung`, `Performance`, `Stabilität`, " +
    "`Schwierigkeitsgrad`, `Interaktivität`, `Feedback`, `Nutzwert`)" +
    "VALUES ('" +
    +RatingtoolGUI.getNameField()+"', '" +
    +RatingtoolGUI.getDesignBew()+"', '" +
    +RatingtoolGUI.getLayoutBew()+"', '" +
    +RatingtoolGUI.getBedienungBew)+"', '" +
    +RatingtoolGUI.getPerformanceBew)+"', '" +
    +RatingtoolGUI.getStabilitätBew)+"', '" +
    +RatingtoolGUI.getSchwChooser)+"', '" +
    +RatingtoolGUI.getInterBew)+"', '" +
    +RatingtoolGUI.getFeedBew)+"', '" +
    +RatingtoolGUI.getNutzBew)+"');"
```

**Abb. 9.37** Generieren des zweiten SQL-Befehls

**Abb. 9.38** Die Klasse  
„calculateRating“

calculateRating
+RatingBildung(): String
+RatingBücher(): String
+RatingSonstige(): String
+RatingNuM(): String
+RatingSpiele(): String
+RatingReferenz(): String
+rating(): String

und in der Variable „sum“ (Summe) gespeichert. Falls eine Kategorie mit null bewertet ist, gilt diese als nicht relevant für die Bewertung und der Divisor wird entsprechend um eine Einheit verringert.

```
public static String RatingSonstige(){

    int divisor=10;
    double sum= Integer.parseInt(RatingtoolGUI.getDesignBew())
        +Integer.parseInt(RatingtoolGUI.getLayoutBew())
        +Integer.parseInt(RatingtoolGUI.getBedienungBew())
        +Integer.parseInt(RatingtoolGUI.getPerformanceBew())
        +Integer.parseInt(RatingtoolGUI.getStabilitätBew())
        +Integer.parseInt(RatingtoolGUI.getFunktBew())
        +Integer.parseInt(RatingtoolGUI.getSichBew())
        +Integer.parseInt(RatingtoolGUI.getWerbeBew())
        +Integer.parseInt(RatingtoolGUI.getPersoBew())
        +Integer.parseInt(RatingtoolGUI.getNutzwBew());
```

**Abb. 9.39** Addition der Bewertungen

```
public static String rating(int divisor, double sum){

    double erg=sum/divisor;

    System.out.println(sum);
    System.out.println(sum+ " : "+divisor+ " = "+erg);
    DecimalFormat df = new DecimalFormat("0.00");
    String rating = df. format(erg);
    return rating;
```

**Abb. 9.40** Berechnen des Ratings

Die Variablen „divisor“ und „sum“ werden abschließend an die Methode „rating“ übergeben. Hier wird das arithmetische Mittel, d. h. die durchschnittliche Anzahl an Sternen, berechnet. Abbildung 9.40 zeigt die Umsetzung dieser Funktion.

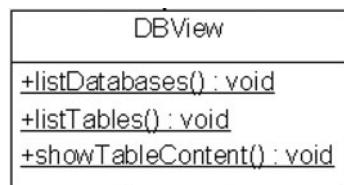
Durch Angabe eines „DecimalFormat“ wird das Ergebnis auf zwei Stellen gerundet. Über den Befehl „Return“ wird abschließend das Ergebnis („erg“) der Division der Bewertungen (Anzahl an Sternen) und dem Divisor (Anzahl an Bewertungskategorien) zurückgeliefert.

#### 9.4.3.1.6 Anzeige der Datenbanktabellen

Eine weitere Anforderung an das Rating-Tool ist es, den Inhalt der einzelnen Tabellen der Datenbank darzustellen. Das Fenster zum Anzeigen der Tabellen wurde bereits im Zusammenhang mit dem GUI dargestellt (siehe Abb. 9.25). Das Auslesen und Anzeigen der Tabelleninhalte erledigt die Klasse „DBView“ (siehe Abb. 9.41), die jedoch auf das Verbindungsmanagement der Klasse „MySQLConnection“ angewiesen ist.

Ist eine Verbindung zur Datenbank hergestellt, wird als erstes über die Methode „listDatabases“ der MySQL-Server nach Datenbanken untersucht. Alle verfügbaren Datenbanken werden im Drop-down-Menü aufgelistet (siehe Abb. 9.42). Wird eine Datenbank vom

**Abb. 9.41** Die Klasse „DBView“



**Abb. 9.42** Die Drop-down-Menüs



Nutzer ausgewählt, werden durch Aufruf von „listTables“ alle vorhandenen Tabellen im zweiten Auswahlmenü angezeigt.

Sind sowohl eine Datenbank und eine Tabelle ausgewählt, kann der Inhalt der Tabelle über den Button „Inhalt anzeigen“ abgerufen werden. Der Button löst den Aufruf der Methode „showTableContent“ aus. Deren Ablauf wird in Abb. 9.43 dargestellt.

Zu Beginn wird, wie bei der „sendQuerys“-Methode, ein Statement-Objekt erzeugt, über das die SQL-Abfrage verschickt wird. Dann werden zwei String-Variablen deklariert: Die Variable „db“ für die ausgewählte Datenbank und die Variable „tbl“ für die ausgewählte Tabelle. Mit der Variable „db“ wird festgelegt, welche Datenbank verwendet werden soll („USE + db“). Die Variable „tbl“ wird zur Erstellung der SQL-Abfrage verwendet (SELECT \* FROM tbl). Anschließend wird die SQL-Abfrage generiert, z. B. fragt „SELECT \* FROM Bildung“ alle Daten aus der Tabelle Bildung ab. Das Ergebnis dieser Abfrage soll im nächsten Schritt in einer Tabelle (JTable) angezeigt werden. Den Programmcode hierzu zeigt die folgende Abb. 9.44.

Jedes „JTable“ benötigt ein Table Model, um Daten anzeigen zu können. Es werden zwei Vektoren erzeugt. Ein Vektor ist eine lineare Liste von Elementen, ähnlich einem Array. Sie werden für die Spaltennamen („clmHeader“) und für die Inhalte der Tabelle („dataVector“) benötigt. Die Inhalte einer Tabelle, die durch die SQL-Abfrage geschickt werden, sind in der Variable „TblData“ enthalten. Mit diesen Daten wird das Table Model gefüllt werden. Beispielhaft wird angenommen, dass die SQL-Abfrage folgende Daten (siehe Abb. 9.45) liefert.

Das Füllen der Tabelle mit den Daten wurde mit zwei For-Schleifen und einer While-Schleife gelöst. Der implementierte Code, um die Daten in die Tabelle zu übergeben, ist in Abb. 9.46 dargestellt.

Der Ablauf des Programmcodes aus Abb. 9.46 ist wie folgt:

**Zeile 1–4:** Der Integer-Variablen „clmCnt“ wird die Anzahl der Spalten übergeben, die die abgefragte Tabelle enthält. Die Spaltenanzahl wird in der For-Schleife als Testausdruck dienen. Damit wird sichergestellt, dass alle Spalten übernommen werden. Die For-Schleife

```

static void showTableContent(){

    try{

        Statement stmt = MySQLConnection.con.createStatement();
        String db = (String)RatingtoolGUI.jComboBox1.getSelectedItem();
        if (db == null || db == "")      return;
        stmt.execute("USE " + db);

        String tbl = (String)RatingtoolGUI.jComboBox2.getSelectedItem();
        if (tbl == null || tbl == "")   return;
        ResultSet tblData = stmt.executeQuery("SELECT * FROM " + `"+tbl+"`");
    }
}

```

**Abb. 9.43** Die Methode „showTableContent“**Abb. 9.44** Das Table Model

```

tblDataModel = new DefaultTableModel();
Vector clmHeader = new Vector();
Vector dataVector = new Vector();

```

**Abb. 9.45** Beispiel zur SQL-Abfrage

Name	Kategorie
WhatsApp	Kommunikation
LernApp	Bildung

fügt alle Spaltenüberschriften („getColumnName“) in die „JTable“ ein. Im Beispiel sind dies die Spalten „Name“ und „Kategorie“.

**Zeile 6–7:** Die While-Schleife durchläuft das Ergebnis der SQL-Abfrage Zeile für Zeile und erzeugt für jede Zeile einen neuen Vektor („rowVector“).

**Zeile 8–11:** Durch die zweite For-Schleife werden dem „rowVector“ die einzelnen Spalten der aktuellen Zeile hinzugefügt. Sind alle Spalten hinzugefügt, wird der Inhalt dieser Zeile in den „DataVector“ angehängt. Im Beispiel hätte es zwei Durchläufe gegeben, da zwei Zeilen Inhalt in der Tabelle vorhanden sind.

**Zeile 14:** Am Ende des Ablaufs der Methode sind im „clmHeader“ die Spaltenüberschriften und im „DataVector“ der gesamte Inhalt der abgefragten Tabelle enthalten. Diese werden abschließend dem Table Model zugeordnet.

Zum Abschluss wird, wie in Abb. 9.47 einzusehen ist, dieses Table Model der Tabelle („JTable1“) übergeben. Außerdem wird ein „RowSorter“ hinzugefügt, damit die Spalten sortiert werden können. Durch Aufruf von „updateUI“ aktualisiert sich das „JTable“ und zeigt die abgefragten Daten an.

Das folgende Aktivitätsdiagramm (siehe Abb. 9.48) veranschaulicht nochmals den zuvor geschilderten Ablauf in Zeile 6–11 (siehe Abb. 9.46).

#### 9.4.3.1.7 Die Klasse „Start“

Die Klasse „Start“ (siehe Abb. 9.49) beinhaltet die Startparameter des Rating-Tools. Die „Main“-Methode erzeugt über den Konstruktor, der alle Komponenten initialisiert, ei-

```

1  ResultSetMetaData rsmd = tblData.getMetaData();
2      int clmCnt = rsmd.getColumnCount();
3      for(int i = 1; i <= clmCnt;i++)
4          clmHeader.addElement(rsmd.getColumnName(i));
5
6  while(tblData.next()){
7      Vector rowVector = new Vector();
8      for(int i = 1; i <= clmCnt; i++){
9          rowVector.addElement(tblData.getString(i));
10     }
11     dataVector.addElement(rowVector);
12 }
13
14 tblDataModel.setDataVector(dataVector,clmHeader);

```

**Abb. 9.46** Die Tabelleninhalte dem „JTable“ übergeben

```

RatingtoolGUI.jTable1.setModel(tblDataModel);
RatingtoolGUI.jTable1.setAutoCreateRowSorter(true);
RatingtoolGUI.jTable1.updateUI();

```

**Abb. 9.47** Table Model an „JTable“ übergeben

ne neue GUI-Oberfläche und macht diese sichtbar. Ohne „setVisible(true)“ würde die Oberfläche zwar erzeugt, jedoch nicht angezeigt werden (siehe Abb. 9.50).

#### 9.4.3.2 Evaluation des Prototyps

Nachdem die Anforderungen in einem ersten Prototyp umgesetzt sind, folgt der nächste Schritt im Vorgehensmodell, die Evaluation des Prototyps. In einem Abstimmungsmeeting mit dem Auftraggeber wurden der erste Prototyp sowie die Datenbank vorgestellt und evaluiert. Das Meeting hat ergeben, dass der Prototyp grundsätzlich den Anforderungen entspricht, jedoch noch folgende zusätzliche Erweiterungen vorgenommen werden sollen:

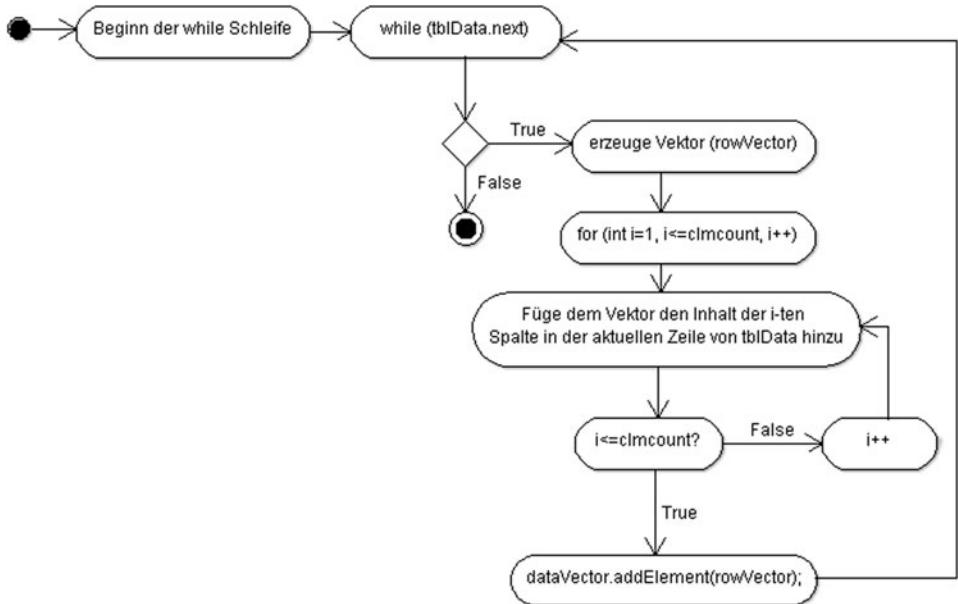
- Implementierung eines Dialogs zum Einloggen in die Datenbank
- Implementieren eines Dialogfensters zur Meldung der erfolgreichen bzw. nicht erfolgreichen Speicherung in der Datenbank

Entsprechend diesen Vorgaben soll der Prototyp abschließend im Folgenden überarbeitet werden.

#### 9.4.3.3 Überarbeiten des Prototyps

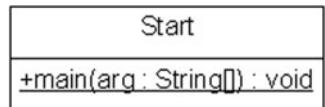
##### 9.4.3.3.1 Log-in-Dialog

Als erster Schritt in der Überarbeitung des Prototyps wird das Log-in-Fenster entworfen (siehe Abb. 9.51). Es besitzt ein Panel, auf dem sich drei Textfelder für die Eingaben befinden, sowie zwei Buttons zum Einloggen und Abbrechen des Vorgangs.



**Abb. 9.48** Auslesen des Tabelleninhaltes

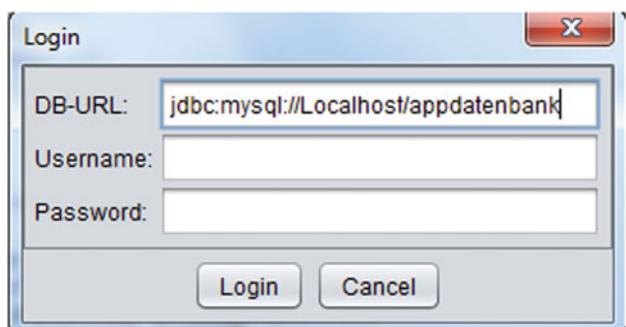
**Abb. 9.49** Die Klasse „Start“



**Abb. 9.50** Der Aufruf des Konstruktors

```
new RatingtoolGUI().setVisible(true);
```

**Abb. 9.51** Der Log-in-Dialog



Zum Erzeugen des Log-in-Dialogs wird eine neue gleichnamige Klasse (siehe Abb. 9.52) implementiert.

**Abb. 9.52** Die Klasse „LoginDialog“

LoginDialog
+LoginDialog()
+initLogin() : void
+getPassword() : String
+getURL() : String
+getUsername() : String

**Abb. 9.53** Menüpunkt „Datenbank“



Die Methode „initLogin“ dient zum Erzeugen und Anzeigen des Log-in-Dialogs. Ihr erster Aufruf wird direkt nach dem Starten des Rating-Tools durchgeführt, wodurch der Nutzer aufgefordert wird, die Verbindungsdaten anzugeben. Zusätzlich wurden dem GUI die Menüpunkte „Verbinden“ und „Ausloggen“ hinzugefügt, über die jederzeit die Verbindung zur Datenbank hergestellt und beendet werden kann (siehe Abb. 9.53).

Die Methoden „getPassword“, „getURL“ und „getUsername“ liefern die Verbindungsdaten zurück, die vom Nutzer eingegeben wurden. Diese werden von der Klasse „MySQLConnection“ zum Verbindungsauflauf benötigt. Zum Abschluss muss daher die Klasse „MySQLConnection“ überarbeitet werden. Bisher waren die URL, der Benutzername sowie das Passwort im Code hinterlegt. Doch diese Angaben sollen über das neue Log-in-Fenster vom Anwender selbst eingegeben werden. Hierfür werden die eben genannten Methoden, wie in Abb. 9.54 einzusehen, zum Verbindungsauflauf eingesetzt.

Bei einem Verbindungsauflauf zur Datenbank werden von nun an die Daten verwendet, die über das Log-in-Fenster eingegeben werden.

#### 9.4.3.3.2 Feedbackmeldungen

Bisher wird dem Anwender, nachdem er die Daten eines App-Tests an die Datenbank geschickt hat, keinerlei Rückmeldung gegeben, ob das Abspeichern erfolgreich oder nicht erfolgreich war. Er muss daher selbst überprüfen, ob die Daten angekommen sind. Diese Tatsache soll sich mit der Implementation eines Dialogfensters, das Feedback über den Speichervorgang gibt, ändern. Das Java-Paket „javax.swing“ besitzt eine Klasse namens „JOptionPane“. Die Klasse kann Pop-up-Dialoge erzeugen, um z. B. Informationen oder Fehlermeldungen anzuzeigen. Diese Klasse wird genutzt, um Meldungen darüber zu liefern, ob das Speichern der Daten erfolgreich war oder fehlgeschlagen ist. Um eine entsprechende Meldung zu liefern, werden für beide Möglichkeiten jeweils ein Dialogfenster

```
try {
    con = DriverManager.getConnection(
        "+LoginDialog.getURL()+"", ""+LoginDialog.getUsername()+"", ""+LoginDialog.getPassword()+"");
```

**Abb. 9.54** Anpassen des Verbindungsaufbaus

### Programmcode:

```
JOptionPane.showMessageDialog(null,
    "Die Daten der App "+RatingtoolGUI.getNameField()+" " +
    "wurden erfolgreich in der App-Datenbank gespeichert!",
    "Speichern erfolgreich",
    JOptionPane.INFORMATION_MESSAGE);
```

### Anzeige der Meldung:



**Abb. 9.55** Die Erfolgsmeldung

(„JOptionPane“) hinzugefügt, das über den Status des Speicherns der Daten Rückmeldung gibt. Die Implementierung der Erfolgsmeldung sowie ein Beispiel dieser Meldung ist in Abb. 9.55 dargestellt.

Falls der Speichervorgang nicht erfolgreich durchgeführt wurde, soll ebenfalls eine Meldung angezeigt werden. Wenn die JDBC einen Fehler bei der Interaktion mit einer Datenbank feststellt, wird dies durch eine SQL-Exception gemeldet. Über die Methode „getMessage“ einer SQL-Exception kann eine Beschreibung des Fehlers abgerufen werden. Diese Möglichkeit wird genutzt, um im Dialogfenster eine Fehlermeldung anzuzeigen. Sowohl die Implementierung als auch die Meldung selbst wird in Abb. 9.56 dargestellt.

In diesem Fall wird in Abb. 9.56 die Fehlermeldung „Duplicate Key“ angezeigt. Das bedeutet, dass der Name dieser App bereits in der Datenbank vorhanden ist und somit ein Test für diese App bereits angelegt wurde.

#### 9.4.3.4 Fertigstellung des Rating-Tools

Der nach den Vorgaben überarbeitete Prototyp des Rating-Tools wurde anschließend in einem weiteren Meeting mit dem Auftraggeber evaluiert. Diese abschließende Evaluierung ergab, dass der Prototyp in dieser Form die Anforderungen des Auftraggebers erfüllt und betriebsbereit ist. Somit kann das Rating-Tool fertiggestellt werden. Alle entwickelten Java-Klassen werden im Paket „iLogin\_Ratingtool“ zusammengefasst. Ge-

## Programmcode:

```
} catch (SQLException e) {  
    System.out.println(e.getMessage());  
    JOptionPane.showMessageDialog(null,  
        "Fehlermeldung: "+e.getMessage(), "FEHLER",  
        JOptionPane.ERROR_MESSAGE);
```

## Anzeige der Meldung:



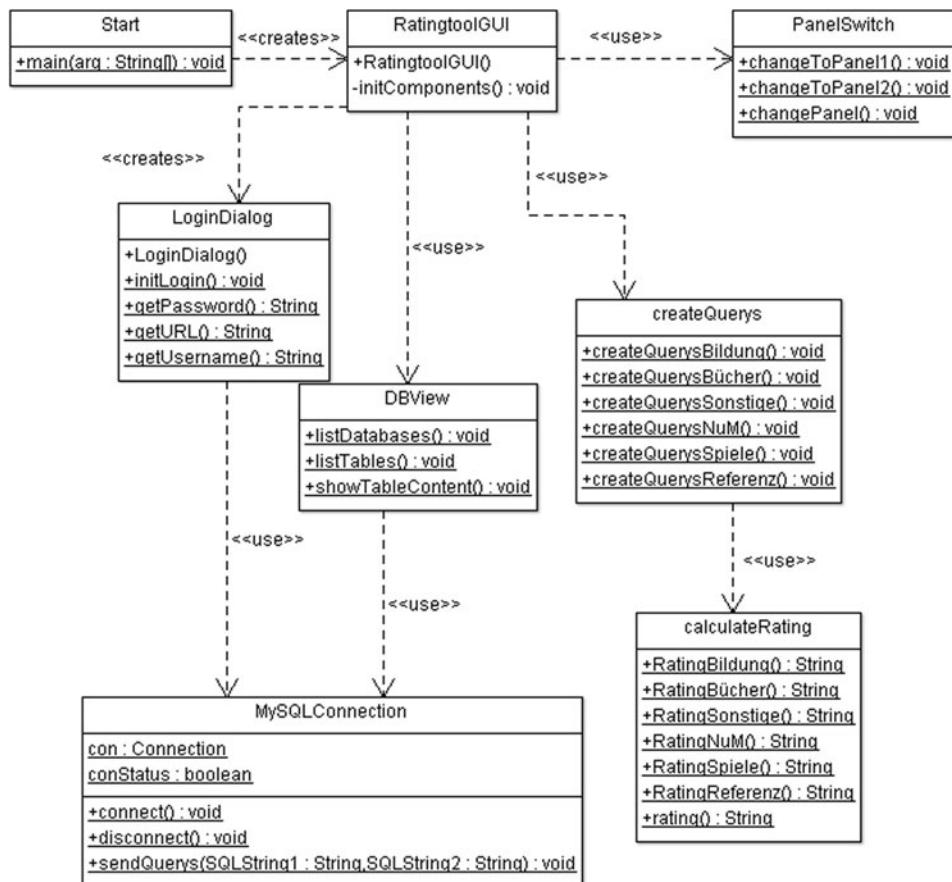
**Abb. 9.56** Die Fehlermeldung

startet wird die Anwendung über eine lauffähige JAR-Datei (Java Archive) mit dem Titel „iLogin-Ratingtool“, in der das gesamte Paket hinterlegt ist. Abschließend stellt das folgende UML-Diagramm in Abb. 9.57 die Abhängigkeiten der Klassen des Rating-Tools untereinander dar.

### 9.4.3.5 Test des Rating-Tools

Nachdem die Entwicklung abgeschlossen ist, soll sich dieser Abschnitt zum Abschluss dem Test des Rating-Tools widmen. Hier soll eine App beispielhaft bewertet werden, um anschließend die Daten über das Rating-Tool in die Datenbank einpflegen zu können. Zum Test steht die App „WhatsApp“ zur Verfügung. Der „WhatsApp“-Messenger ist ein Multiplattform-Messenger für iPhones, BlackBerry, Android- und Nokia-Smartphones. Mit „WhatsApp“ kann der Nutzer kostenlos Nachrichten verschicken und mit anderen Nutzern der App chatten. Daher wird die App in die Kategorie „Kommunikation“ eingeordnet. Als Beziehungstyp wird C2C ausgewählt, da zwei Nutzer miteinander in Verbindung treten und miteinander kommunizieren. Zur Verfügung steht die iPhone-Version der App, daher wird ein Test speziell für diese angelegt. Die allgemeinen Informationen zur App sehen folgendermaßen aus (siehe Abb. 9.58):

Ein Fazit zur App folgt am Ende der Bewertung. Der nächste Schritt ist die eigentliche Bewertung, beginnend mit den allgemeinen Bewertungskategorien.



**Abb. 9.57** Das Klassendiagramm des Rating-Tools

**Abb. 9.58** Allgemeine Informationen zur App

Name	WhatsApp Messenger für iPhone
Kategorie	Kommunikation
Typ	C2C
Hersteller	WHATSAPP INC.
Release	2012-12-07
Version	2.8.7
Kompatibilität	iOS 4.3
Größe	11.5 MB
Preis	0,89 €
Altersfreigabe	Niedrige Stufe

Design/Darstellung ★★★★★ Layout ★★★★★ Bedienung ★★★★★

Performance ★★★★★ Stabilität ★★★★★

**Abb. 9.59** Allgemeine Bewertungskategorien von „WhatsApp“

„WhatsApp“ besitzt ein ansprechendes, für Chatprogramme standardmäßiges **Design** und ein übersichtliches **Layout**. Alle Symbole und Elemente des Designs sind selbsterklärend, sodass ein neuer Nutzer sich gut zurechtfindet. Die **Bedienung** reagiert flüssig. Das Tippen von Textnachrichten ist einfach und mit etwas Übung schnell möglich. Aufgrund des sehr guten Eindrucks während des Tests wird die App in allen drei Kategorien mit fünf Sternen bewertet. In den Kategorien **Performance** und **Stabilität** gibt es ebenfalls keinerlei Abzüge. Die App startet schnell und ist sofort betriebsbereit. Während der Testphase gab es keine Abstürze oder Hänger. Demzufolge wird die App in den allgemeinen Bewertungskategorien wie in Abb. 9.59 dargestellt, bewertet.

Als Nächstes folgt die Bewertung der speziellen Bewertungskategorien. Der **Funktionsumfang** der App lässt keine Wünsche offen. Zusätzlich zur Möglichkeit, reine Textnachrichten zu verschicken, bietet „WhatsApp“ die Möglichkeit, über mehrere Plattformen (Android, iOS, Blackberry OS, Symbian) Nachrichten zu verschicken. Die App ermöglicht Gruppenchats und besitzt die Funktion Push-Benachrichtigung, d. h. neue Nachrichten werden, wie bei einer SMS (Short Message Service), per Signalton oder Vibration gemeldet. „WhatsApp“ ermöglicht neben dem Verschicken von Textnachrichten auch das Senden von Bildern, Sound und Videodateien. Zusätzlich lassen sich Kontakte sowie der aktuelle Standort in Form von Google-Maps-Daten schicken. Aufgrund dessen wird die Kategorie Funktionsumfang mit fünf Sternen bewertet.

Im Bereich **Sicherheit** gibt es jedoch Abzüge. Zu Beginn muss der Nutzer seine Handynummer angeben, damit per SMS der Account freigeschaltet werden kann. Dann durchsucht „WhatsApp“ selbstständig das Handy nach Kontakten und deren Handynummern, um zu erkennen, wer ebenfalls „WhatsApp“ verwendet. Die App überträgt allerdings alle gespeicherten Telefonnummern auf die eigenen Server und anonymisiert diese nicht. Des Weiteren verschlüsselt „WhatsApp“ die Nachrichten nicht. Daher können diese in unverschlüsselten und öffentlichen WLAN-Netzwerken leicht abgefangen und mitgelesen werden.<sup>27</sup> Eine Anwendung namens „WhatsApp-Sniffer“ kam vor Kurzem für genau diesen Zweck auf den Markt, wurde aber schnell wieder von Google entfernt.<sup>28</sup> Aufgrund der Menge an persönlichen Daten, die ausgetauscht werden, und der Unklarheit darüber, wie sicher diese sind, wird die Kategorie „Sicherheit“ mit drei Sternen als befriedigend bewertet.

Der „WhatsApp“-Messenger ist nicht nur sehr preiswert, sondern auch komplett **werbefrei**. Daher erhält er auch in diesem Bereich die Bestnote. Weiterhin ist es möglich, einige

<sup>27</sup> Vgl. Focus 2013.

<sup>28</sup> Vgl. Heise 2013.

Funktionsumfang ★★★★☆ Sicherheit ★★★★☆ Werbung ★★★★☆

Personalisierung ★★★★☆ Nutzwert ★★★★☆

**Abb. 9.60** Spezielle Bewertungskategorien von „WhatsApp“

persönliche Einstellungen vorzunehmen, um es an die eigenen Vorlieben anzupassen, z. B. ein eigenes Profilbild auswählen, individuelle Hintergrundbilder und Nachrichtentöne einstellen, den eigenen Status angeben (z. B. verfügbar, beschäftigt) und das Erstellen einer Favoritenliste der Kontakte.

Die Kategorie **Personalisierung** wird daher mit vier Sternen bewertet. Der Grund für den Abzug ist, dass eine Möglichkeit vermisst wird, den Skin, das Aussehen der Elemente und die generelle optische Darstellung zu verändern, z. B. Farbe der Taskleiste oder der Sprechblasen. Abschließend wird der **Nutzwert** mit vier Sternen als „gut“ bewertet, da der langfristige Nutzen der App stark davon abhängt, wie viele der Freunde und Bekannte ebenfalls „WhatsApp“ verwenden. Daraus ergeben sich folgende Bewertungen in den speziellen Bewertungskategorien, die in Abb. 9.60 dargestellt sind.

Zum Abschluss einer Bewertung wird ein kurzes Fazit zur App verfasst:

#### Testfazit zu „WhatsApp“

„WhatsApp“ ist ein sehr guter Multiplattform-Messenger, der Sie kostenfrei mit anderen Nutzern der App chatten lässt. Vor allem für SMS-Vielschreiber ist die App hervorragend, da einiges an Kosten gespart werden kann. Vorausgesetzt ist natürlich ein WLAN-Zugang oder eine Internetflatrate. Außerdem durchsucht „WhatsApp“ automatisch Ihre Kontaktliste und listet alle Nutzer der App auf. In puncto Sicherheit sollte man sich jedoch im Klaren sein, dass Zugriff auf die persönlichen Daten gewährt werden muss und es noch unklar ist, was mit diesen Daten geschieht. Wird dies akzeptiert, bekommt man jedoch für weniger als einen Euro eine sehr gute Messenger-App mit vielen Funktionen. Vorausgesetzt genügend Freunde und Bekannte besitzen diese App ebenfalls, da nur dann die App ihren Nutzen entfalten kann.

Nachdem der Test durchgeführt wurde, können die Angaben im Rating-Tool gemacht werden. Dazu wird zuerst das Tool gestartet und sich über das Log-in-Fenster (siehe Abb. 9.51) in die Datenbank eingeloggt. Dann können über die Textfelder die Daten angegeben werden. Über die Auswahlmenüs werden ebenfalls die entsprechenden Einträge ausgewählt (siehe Abb. 9.61). Sobald eine Kategorie ausgewählt wurde, erscheint das Panel für die speziellen Bewertungskategorien.

Danach werden für die einzelnen Kategorien über die „JSpinner“ die Bewertungen abgegeben (siehe Abb. 9.62).

Allgemeine Informationen

Name	WhatsApp Messenger	Kategorie	Kommunikation	Typ	C2C
Hersteller	WHATSAPP INC.	Release	2012-12-07	Version	2.8.7
Kompatibilität	iOS 4.3	Größe (MB)	11.5	Preis (€)	0.89
Altersfreigabe	Niedrige Stufe				
Fazit	Vor allem für SMS-Vielschreiber ist die App hervorragend, da einiges an Kosten gespart werden kann. Vorausgesetzt ist natürlich ein WLAN-Zugang oder eine Internetflatrate. Außerdem durchsucht WhatsApp automatisch ihre Kontaktliste und listet alle Nutzer der App auf. In Punkt Sicherheit sollte man sich jedoch im Klaren sein, dass Zugriff auf die persönliche Daten besteht. Wird dies akzeptiert, bekommt man jedoch weniger als einen Euro eine sehr gute Messenger-App. Vorausgesetzt, genügend Freunde & Bekannte besitzen diese App ebenfalls, da nur dann die Anwendung von Nutzen ist.				

**Abb. 9.61** Eintragen der allgemeinen Informationen

Allgemeine Bewertungskategorien

Design/Darstellung	5	Layout	5	Bedienung	5	Performance	5	Stabilität	5
--------------------	---	--------	---	-----------	---	-------------	---	------------	---

Bewertungskategorien : Kommunikation

Funktionsumfang	5
Sicherheit	3
Werbung	5
Personalisierung	4
Nutzwert	4

**Speichern**

**Abb. 9.62** Abgeben der Bewertungen

Um den App-Test in der Datenbank zu speichern, muss der „Speichern“-Button gedrückt werden. Bei erfolgreicher Speicherung wird die entsprechende Meldung geliefert (siehe Abb. 9.63).

Über die integrierte Datenbankansicht kann abschließend überprüft werden, ob alle Daten richtig abgespeichert worden sind. Über den Menüpunkt „Wähle“ wird die Datenbankansicht ausgewählt. In den Auswahlmenüs (siehe Abb. 9.25) wird die „appdatenbank“



**Abb. 9.63** Meldung über erfolgreiche Speicherung

Name	Kategorie	Typ	Hersteller
WhatsApp Messenger für iPhone	Kommunikation	C2C	WHATSAPP INC.

**Abb. 9.64** Ausschnitt der Anzeige der „WhatsApp“-Daten

und die Tabelle „allgemeineinfos“ ausgewählt. Nachdem der „Inhalt anzeigen“-Button gedrückt wurde, zeigt das Rating-Tool den Inhalt der Tabelle an, wie in Abb. 9.64 dargestellt.

Wie vom betriebsbereiten Prototyp erwartet, wurden die Daten der App erfolgreich in der App-Datenbank abgespeichert. Mit der Anwendung des Ratingmodells und dem erfolgreichen Test des Rating-Tools ist hiermit auch die Realisierung des Rating-Tools abgeschlossen. Folglich wurde auch das zweite und letzte Ziel dieses Kapitels erreicht.

---

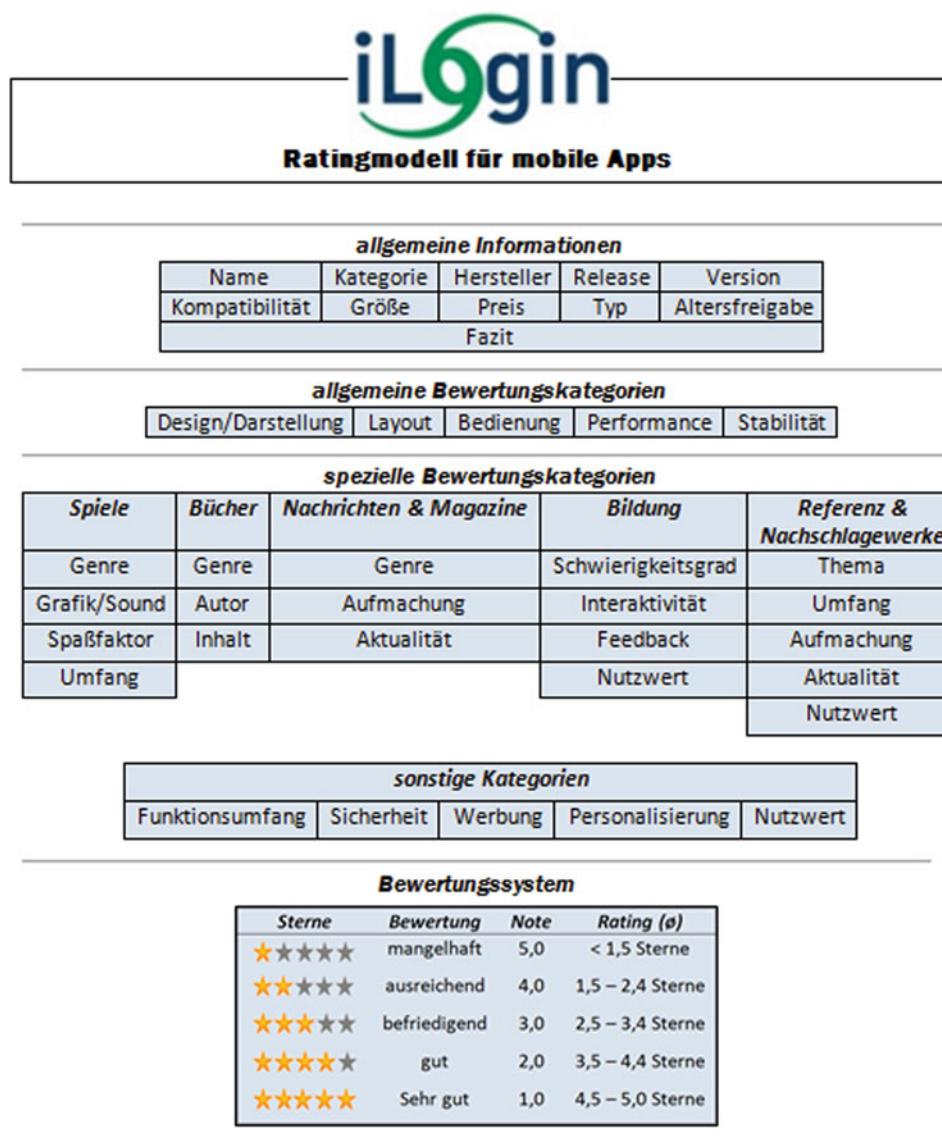
## 9.5 Fazit und Ausblick

### 9.5.1 Fazit

Ausgangssituation dieses Kapitels war der Bedarf an einem Bewertungsmodell, das mobile Applikationen detaillierter bewertet, als es in den App-Stores der Fall ist. Das erste Ziel dieses Kapitels war es demnach ein Ratingmodell zu entwickeln, mit dem mobile Applikationen verschiedener Kategorien bewertet werden können. Die nachfolgende Abb. 9.65 zeigt eine Übersicht über das entwickelte Ratingmodell.

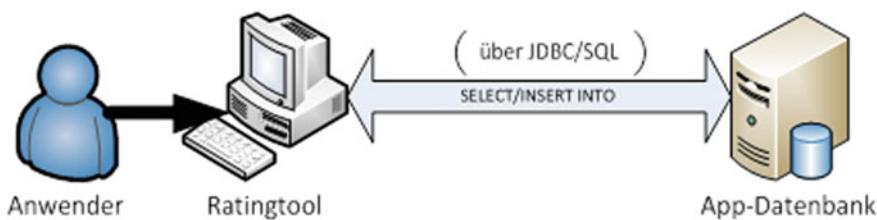
Die Anforderungen (siehe Abschn. 9.3.1), die zu Beginn bzgl. der Bewertungsbereiche, App-Kategorien und des Bewertungssystems an das Modell gestellt wurden, sind erfüllt worden. Das Ratingmodell umfasst:

- 24 unterschiedliche App-Kategorien
- die Aufnahme allgemeiner Daten zu einer App
- einen allgemeinen Bewertungsbereich, geltend für alle Apps
- einen speziellen Bewertungsbereich für unterschiedliche Kategorien
- ein Fünf-Sterne-Bewertungssystem



**Abb. 9.65** Das Ratingmodell für mobile Applikationen

Das entwickelte Ratingmodell ermöglicht es, die mobilen Apps in fünf allgemeinen und, je nach Kategorie, drei bis fünf speziellen Bewertungskategorien zu bewerten. Zusätzlich wird auch ein Fazit zu jeder getesteten App verfasst. Somit kann dem potenziellen Käufer einer App eine Hilfestellung bei seiner Entscheidung gegeben werden. Die Entwicklung des Ratingmodells war jedoch nicht das einzige Ziel dieses Kapitels. Basierend auf dem



**Abb. 9.66** Informationsaustausch zwischen den Komponenten

Bewertungsmodell sollte zusätzlich ein Rating-Tool in Java erstellt werden, mit dem die Apps in der Praxis bewertet werden können. Das Rating-Tool wurde in einem Prototypenprozess entwickelt. Insgesamt wurden zwei Prototypen erstellt. Der erste Prototyp erfüllte zwar die Anforderungen, doch nach der ersten Evaluation wurden minimale Erweiterungen implementiert. Der überarbeitete Prototyp des Rating-Tools wurde anschließend für betriebsbereit und einsatzfähig erklärt, da keine weiteren Änderungen gefordert bzw. Wünsche geäußert wurden. Das Rating-Tool interagiert, wie vom Auftraggeber gefordert (siehe Abschn. 9.4.1), mit einer MySQL-Datenbank, in der die App-Testdaten abgespeichert werden. Diese Kommunikation veranschaulicht die folgende Abb. 9.66.

Der Anwender gibt die Bewertung einer App über das Rating-Tool ab. Das Rating-Tool speichert anschließend über die JDBC und per SQL-Befehl „INSERT INTO“ die Daten in die App-Datenbank ab. Mit dem Rating-Tool ist es ebenfalls möglich, die Inhalte der Datenbank abzufragen und einzusehen. Somit wurden die beiden eingangs gesetzten Ziele, die Konzeption und Entwicklung eines Ratingmodells/Tools zur Bewertung von mobilen Applikationen, erreicht.

## 9.5.2 Ausblick

Zum Abschluss soll noch ein Ausblick gegeben werden über einerseits eine Frage, die bei der zukünftigen Verwendung des Ratingmodells eine Rolle spielen könnte, andererseits die Erweiterungsmöglichkeiten des Rating-Tool betreffend.

### 9.5.2.1 Ansatz zur Vorauswahl der Apps

Nachdem das Ratingmodell erstellt und das Rating-Tool entwickelt worden ist, soll sich am Ende dieses Beitrags noch mit folgender Frage auseinandersetzen, die sich der zukünftige Anwender stellen könnte: **Welche Apps sollen in das Bewertungsmodell aufgenommen werden?**

Aktuell sind Tausende von Apps in den einzelnen App-Stores vorhanden. Daher ist die Frage berechtigt und gleichzeitig nicht ganz einfach zu beantworten. Es stellt sich zugleich die Frage, wie man die „guten Apps“ findet und wie können im besten Falle die schlechten weitestgehend herausgefiltert werden. Einen Ansatz bzw. eine Hilfestellung zum Finden und der Vorauswahl der Apps soll mit den folgenden sechs Punkten gegeben werden.

- **Ranglisten der App-Stores**

Die meisten App-Shops heutzutage, sei es Google Play oder der App-Store von Apple, bieten eine Rangliste an, in der die Apps nach ihrer Popularität bzw. ihren Verkaufszahlen geordnet werden. Ein erster Schritt, um die „guten“ Apps zu finden, wäre es demnach, die vorderen Plätze der „Top-Listen“ zu durchsuchen.

- **Bewertung**

Die Kundenbewertungen und Rezensionen von Nutzern der Apps sollten auch berücksichtigt werden, bevor eine App in das Bewertungsmodell aufgenommen wird.

- **Das Last-but-not-least-Prinzip**

An vorderster Stelle in einer Rangliste stehen die populärsten bzw. erfolgreichsten Apps, doch wie in der Einleitung festgestellt, steht Popularität nicht direkt für den Erfolg bzw. die Qualität einer App. Es stellt sich die Frage, wie die „guten Apps“ gefunden werden können, die weniger erfolgreich sind, da weniger Marketing betrieben wird. Ein Ansatz dafür wäre, in den Ranglisten der App-Stores weiter unten zu suchen, denn auch auf den hinteren Plätzen finden sich von Käufern „gut“ bewertete Apps.

- **Alternative App-Tests**

Es gibt heutzutage alternative Webseiten im Internet, die App-Tests durchführen. Diese sind ebenfalls mögliche Anlaufstellen, um passende Apps zu finden. Beispiele für Homepages, Blogs und Magazine, die Apps testen, sind AndroidPit, Appbrain, aber auch Zeitschriften wie Chip, PC-Welt oder Computerbild.

- **Zufallstests**

Aufgrund der großen Anzahl der existierenden Apps wird es nicht ausbleiben, auch fernab von Top-Listen oder Käuferrezensionen, die auch nicht immer objektiv sind, per Zufall Apps auszuwählen, in das Ratingmodell aufzunehmen und zu bewerten.

- **Kriterien**

Eine weitere Möglichkeit wäre es, die Apps im Voraus zu bewerten und zu filtern. Dafür könnten z. B. Kriterien gefunden werden, die erfüllt werden müssen.

- Mindestbewertung (Rezensionen, alternative Tests)
- Ausreichende Angaben zur App
- Aussagekräftige Screenshots für den ersten Eindruck
- Preis
- Kategorie/Aufgabe

Zum Beispiel könnten nur Apps ausgewählt werden, die eine bestimmte Bewertung nicht unterschreiten (z. B. drei Sterne). Es könnten ebenfalls Apps gefiltert werden, die bestimmte Angaben (z. B. Beschreibung der Funktionen, Inhalte) nicht enthalten oder falls im App-Store keine Screenshots abgebildet sind. Der Preis könnte bei der Auswahl auch eine Rolle spielen. Des Weiteren könnte bei jedem Auswahlverfahren ein Ziel gesetzt werden, nur Apps zu finden, die einer bestimmten Kategorie angehören oder eine bestimmte Aufgabe erfüllen. Damit wäre es ebenfalls möglich, die Suche einzuschränken. Abschließend soll festgehalten werden, dass all diese Punkte nur Ansätze zur Vorauswahl bzw. Pre-Evaluation der Apps darstellen sollen. Es soll als Ausgangspunkt dienen für eine in-

tensivere Auseinandersetzung mit dem Gedanken, die Apps vorher zu selektieren, bevor sie in das Ratingmodell aufgenommen werden. Der nächste Schritt hierbei könnte sein, einen Kriterienkatalog zu erstellen und diesen bei der Suche nach Apps zu berücksichtigen. Um jedoch eine definitive Aussage darüber abgeben zu können, ob eine App gut ist oder nicht, wird letztendlich ein richtiger Test nötig sein. Die Apps anhand von Kriterien auszuwählen, bietet möglicherweise nur einen geringen Mehrwert. Die aussagekräftigsten Kriterien, um eine Grundlage für eine Pre-Evaluation zu besitzen, sind aller Voraussicht nach die Punkte Bewertung, Beschreibung und Screenshots einer App.

### **9.5.2.2 Erweiterungsmöglichkeiten des Rating-Tools**

Auch nachdem die Anforderungen des Auftraggebers erfüllt worden sind, gibt es bei einer Software immer wieder Erweiterungsmöglichkeiten. Im Falle des Ratingmodells gäbe es ebenfalls potenzielle, sinnvolle Erweiterungen. In einem Datenbanksystem gibt es vier grundlegende Datenbankoperationen. Dafür steht das Akronym CRUD (Create, Read, Update und Delete). Diese vier Operationen sind in den meisten Datenbankanwendungen (z. B. phpMyAdmin) implementiert. Das Rating-Tool führt in seiner aktuellen Version zwei dieser Operationen aus, nämlich Create (INSERT INTO) und Read (SELECT). Eine mögliche sinnvolle Erweiterung wäre eine Delete-Operation in das Rating-Tool zu implementieren, sodass bereits vorhandene Datensätze aus der Datenbank entfernt werden können. Falls ein Datensatz einmal gelöscht werden müsste, wäre das Rating-Tool nicht mehr abhängig von anderen Tools wie phpMyAdmin.

Die Möglichkeiten zur Erweiterung des Rating-Tools sind vielfältig, je nachdem welche neuen Anforderungen oder Wünsche des Anwenders existieren. Durch neue Methoden oder Klassen ist es jederzeit möglich, das Rating-Tool um neue Funktionen zu erweitern. Der einzige größere Aufwand könnte entstehen, falls zusätzlich auch etwas an der bisherigen grafischen Oberfläche verändert oder hinzugefügt werden muss. Aktuell erfüllt das Rating-Tool jedoch alle Anforderungen des Auftraggebers. Es soll in Zukunft für die Anwendung des Ratingmodells und die Speicherung der App-Tests in einer MySQL-Datenbank genutzt werden. Der finale Prototyp des Rating-Tools ist dafür bestens geeignet und bildet zusätzlich eine Grundlage für zukünftige Erweiterungen.

---

## **Literatur**

- Abts, D.: Masterkurs Client/Server-Programmierung mit Java, 3. Aufl. Vieweg + Teubner Verlag, Wiesbaden (2010)
- Apache: Apache Friends. <http://www.apachefriends.org/de/index.html> (2013). Zugegriffen: 1. Juli 2013
- Apple Inc.: WhatsApp Messenger for iPhone. <http://itunes.apple.com/us/app/whatsapp-messenger/id310633997?mt=8> (2013). Zugegriffen: 1. Juli 2013
- Bitkom: Fast eine Milliarde App-Downloads allein in Deutschland. [http://www.bitkom.org/71303\\_71298.aspx](http://www.bitkom.org/71303_71298.aspx) (2012). Zugegriffen: 1. Juli 2013

- Bleich: Inkasso auf Fingertipp. <http://www.heise.de/ct/artikel/Inkasso-auf-Fingertipp-1102753.html> (2010). Zugegriffen: 1. Juli 2013
- Chip: App-Stores: Windows Marketplace wächst um 65 %. [http://www.chip.de/news/App-Stores-Windows-Marketplace-waechst-um-65\\_54848492.html](http://www.chip.de/news/App-Stores-Windows-Marketplace-waechst-um-65_54848492.html) (2012). Zugegriffen: 01. Juli 2013
- Distimo: Google Android Market Tops 400,000 Applications. [http://www.distimo.com/blog/2012\\_01\\_google-android-market-tops-400000-applications/](http://www.distimo.com/blog/2012_01_google-android-market-tops-400000-applications/) (2012). Zugegriffen: 01. Juli 2013
- Focus: SMS-Ersatz mit Mängeln. [http://www.focus.de/panorama/diverse/handy-sms-ersatz-mit-maengeln\\_aid\\_788859.html](http://www.focus.de/panorama/diverse/handy-sms-ersatz-mit-maengeln_aid_788859.html) (2013). Zugegriffen: 01. Juli 2013
- Google Inc.: WhatsApp Messenger. <https://play.google.com/store/apps/details?id=com.whatsapp> (2012a). Zugegriffen: 1. Juli 2013
- Google Inc.: Bewertung ihrer App in Google Play Store. <http://support.google.com/googleplay/android-developer/bin/answer.py?hl=de&answer=188189> (2012b). Zugegriffen: 1. Juli 2013
- Heise: Schnüffel-Tool zeigt fremde WhatsApp-Nachrichten an. <http://www.heise.de/securitymeldung/Schnueffel-Tool-zeigt-fremde-WhatsApp-Nachrichten-an-1574066.html> (2013). Zugegriffen: 1. Juli 2013
- Laudon, K., Laudon, J., Schoder, D.: Wirtschaftsinformatik: Eine Einführung, 2. Aufl. Pearson Studium, München (2010)
- Lehner, F., Wildner, S., Scholz, M.: Wirtschaftsinformatik: Eine Einführung, 2. Aufl. Hanser Verlag, München (2008)
- Localytics: First impressions matter!. <http://www.localytics.com/blog/2011/first-impressions-matter-26-percent-of-apps-downloaded-used-just-once/> (2012). Zugegriffen: 1. Juli 2013
- Oracle: Download Connector/J. <http://www.mysql.de/downloads/connector/j/> (2013a). Zugegriffen: 1. Juli 2013
- Oracle: Creating a GUI with JFC/Swing. <http://docs.oracle.com/javase/tutorial/uiswing/> (2013b). Zugegriffen: 1. Juli 2013
- Oracle: Learning Swing with the NetBeans IDE. <http://docs.oracle.com/javase/tutorial/uiswing/learn/index.html> (2013c). Zugegriffen: 1. Juli 2013
- Oracle: How to use textfields. <http://docs.oracle.com/javase/tutorial/uiswing/components/textfield.html> (2013d). Zugegriffen: 1. Juli 2013
- Oracle: How to use combo boxes. <http://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html> (2013e). Zugegriffen: 1. Juli 2013
- Oracle: How to use spinners. <http://docs.oracle.com/javase/tutorial/uiswing/components/spinner.html> (2013f). Zugegriffen: 1. Juli 2013
- Oracle: How to use tables. <http://docs.oracle.com/javase/tutorial/uiswing/components/table.html> (2013g). Zugegriffen: 1. Juli 2013
- Oracle: How to use card layout. <http://docs.oracle.com/javase/tutorial/uiswing/layout/card.html> (2013h). Zugegriffen: 1. Juli 2013
- Schatten, A., et al.: Best Practice Software-Engineering. Spektrum Akademischer Verlag, Heidelberg (2010)
- Statista: Mittelwert und arithmetisches Mittel. <http://de.statista.com/statistik/lexikon/definition/91/mittelwert-und-arithmetisches-mittel/> (2012). Zugegriffen: 1. Juli 2013
- Techcrunch: Here's how iPhone app store ratings work. <http://techcrunch.com/2009/06/29/heres-how-iphone-app-store-ratings-work-hint-they-dont/> 2009-06-29 (2012). Zugegriffen: 1. Juli 2013
- Yardley, G.: AppStore secrets. <http://de.slideshare.net/pinchmedia/iphone-appstore-secrets-pinch-media?type=powerpoint> (2009). Zugegriffen: 1. Juli 2013

---

# Sachverzeichnis

## A

Added-Value, 4  
Analysephase, 167  
Anwendungsentwicklung, mobile  
Anforderungen, 168  
Debugger, 198  
Eigenentwicklungen, 156  
Entwicklungsstrategie, 106  
Entwurf, 174  
Framework, 195  
Honorarkosten, 31  
Kundenprojekte, 156  
Lösungsstrategie, 161  
Strategiedefinition, 50  
Strategiefindung, 37  
Strategiumsetzung, 69  
Vorgehensmodell, 135  
Applikation, mobile, 102, 105, 353  
Aktivitäten, 21  
App-Strategie, 36  
Bewertungskriterien, 369  
Bewertungssystem, 371  
Dienste, 102  
Einführung, 201  
Einführungsprozess, 23  
Einsatzgebiete, 15  
Geschäftsmodell, 73  
hybride, 106  
idealtypischer Lebenszyklus, 154  
Marketingmaßnahmen, 210  
nativ, 106, 186  
Nutzungsdauer, 354  
Nutzungsgrad, 21  
Vertriebsmöglichkeiten, 205  
Vorentscheidungen, 24

## B

webbasiert, 106, 186  
Werbemöglichkeiten, 209  
Benutzeroberfläche, 343  
Betriebssystem, 221, 312  
mobiles, 28, 98  
Android, 99, 314  
iOS, 99, 312  
Marktanteile, 29  
Windows Phone, 100, 314  
Bluetooth, 330  
Brainstorming, 37, 162  
Bring Your Own Device, 8, 218  
Brute-Force-Attacke, 220  
Business Case, 36  
Business Plan, 53  
Business Reengineering, 78  
Business-to-Business, Merkmale, 22  
Business-to-Consumer, Merkmale, 22  
BusinessPlan, 50

## C

Chancen-Risiken-Analyse, 63

## D

Dalvik VM, 317  
Datenmodellierung, 180  
Design Thinking, 50

## E

Einführungsphase, 200  
Einführungsprozess  
Big-Bang-Strategie, 201

- Gestaltung, 204  
Step-by-Step-Strategie, 201  
**E**ndgerät, mobiles, 93  
Architektur, 101  
Aufbau, 101  
Eigenschaften, 100  
Typologisierung, 96  
Entwicklungsphase, 185  
Entwicklungsumgebung, 121  
App Inventor, 123, 190, 326  
Eclipse, 122, 189, 323  
Visual Studio, 126, 191  
Xcode, 124, 189  
EVA-Prinzip, 6
- G**  
Gadget, 2  
Gamification, 77  
Geschäftsmodell, 37  
Grobplanung, 167
- I**  
Informationssystemarchitektur, 69  
IT-Sicherheit, 218
- L**  
Lastenheft, 36, 173, 328
- M**  
Make-or-Buy-Entscheidung, 26  
Malware, 221  
Marktanalyse, 25, 155  
Mind Map, 42, 164  
Mission Statement, 53  
Mobile Applikation, 8  
    Herausforderungen, 10  
Mobile Business, 17  
Mobilität, 94  
    Benutzermobilität, 95  
    Dienstmobilität, 95  
    Endgerätemobilität, 95  
Mock-up, 36  
Modelle, 69  
Modellierungsmethode, 69  
    Business Process Model and Notation, 69,  
        183  
    Entity-Relationship-Modell, 69, 180
- Ereignisgesteuerte Prozesskette, 182  
erweiterte ereignisgesteuerte Prozesskette, 69  
Programmablaufplan, 177  
Struktogramm, 178  
Unified Modelling Language, 69  
Multiperspektive, 163
- N**  
Nutzwertanalyse, 166
- O**  
Online-Stores, 25
- P**  
Pflichtenheft, 36, 173, 328  
Problemanalyse, 155  
    Problembewusstsein, 159  
    Problemfindung, 158  
Programmablaufmodellierung, 177  
Programmiersprachen, 115  
    Arten, 116  
    C, 118  
    C++, 118  
    C#, 118  
    HTML, 120  
    Java, 117  
    Objective-C, 119  
    Perl, 358  
    PHP, 358  
Programmierung, 108, 116, 187  
    Algorithmus, 110  
    Datenobjekte, 112  
    Kontrollstrukturen, 112  
    Wertzuweisung, 112  
Project Charter, 48  
Projektmanagement, 80, 82  
    Projekte, 80  
    Projektleitung, 82  
    Projektlenkung, 171  
    Projektorganisation, 83  
    Projektplanung, 170  
    Ressourcenmanagement, 169  
Protokoll, 164  
Prozessmodellierung, 182
- R**  
Ratingmodell, 354  
Ratingtool, 354, 372

**S**

- Schadsoftware, 232
- Schulungsmaßnahmen, 207
- Software
  - Engineering, 88
  - Werkzeuge, 107
- Freeware, 6
- Klassifizierung, 92
- Open Source, 6
- Shareware, 6
- Software-Life-Cycle, 153
- Softwaretests, 194
  - Durchführung, 194
- SWOT-Analyse, 63

**T**

- Testmanagement, 192
  - Benutzertests, 206
  - Testen, 193
  - Vorgehensweisen, 194
- Trojaner, 220

**V**

- Veröffentlichungsphase, 200

**Virus, 220**

- Vorgehensmodell, 137, 138
  - agiles, 146
  - Bewertungsverfahren, 150
  - Extreme Programming, 147
  - klassisches, 140
  - modernes, 143
  - Rational Unified Process, 144
  - Spiralmodell, 141
  - UCAN-Modell, 149
  - V-Modell, 143
  - Wasserfallmodell, 141

**W**

- Widget, 2
  - Widget-Engine, 2
- Wurm, 220

**X**

- XAMPP, 358

**Z**

- Zertifikate, 246