Information Engineering und IV-Controlling
Hrsg.: Franz Lehner, Stefan Eicker, Ulrich Frank,
Erich Ortner, Eric Schoop

Brigitte Eller

Usability Engineering in der Anwendungsentwicklung

Systematische Integration zur Unterstützung einer nutzerorientierten Entwicklungsarbeit

GABLER

RESEARCH

Brigitte Eller

Usability Engineering in der Anwendungsentwicklung

GABLER RESEARCH

Information Engineering und IV-Controlling

Die Schriftenreihe präsentiert aktuelle Forschungsergebnisse der Wirtschaftsinformatik sowie interdisziplinäre Ansätze aus Informatik und Betriebswirtschaftslehre. Ein zentrales Anliegen ist dabei die Pflege der Verbindung zwischen Theorie und Praxis durch eine anwendungsorientierte Darstellung sowie durch die Aktualität der Beiträge. Mit der inhaltlichen Orientierung an Fragen des Information Engineerings und des IV-Controllings soll insbesondere ein Beitrag zur theoretischen Fundierung und Weiterentwicklung eines wichtigen Teilbereichs der Wirtschaftsinformatik geleistet werden.

Brigitte Eller

Usability Engineering in der Anwendungsentwicklung

Systematische Integration zur Unterstützung einer nutzerorientierten Entwicklungsarbeit

Mit einem Geleitwort von Prof. Dr. Erich Ortner

GABLER

RESEARCH

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

Dissertation Technische Universität Darmstadt, 2009

D 17

1. Auflage 2009

Lektorat: Claudia Jeske | Stefanie Loyal

Gabler ist Teil der Fachverlagsgruppe Springer Science+Business Media.
www.gabler.de

Umschlaggestaltung: KünkelLopka Medienentwicklung, Heidelberg
Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier
Printed in Germany

ISBN 978-3-8349-2114-7

Geleitwort

Von manchen Menschen wird die Informationstechnologie (IT) mit ihren ubiquitären Anwendungsgebieten nicht nur als Chance für den weiteren Fortschritt, sondern auch als Belastung oder gar als Bedrohung empfunden. Wird sie die Individuen entmündigen oder wird sie ihre Lebensqualität erhöhen? Eine Entwicklung zum Positiven kann hier nur durch möglichst breites Verstehen der Technologiefortschritte und damit einer nachhaltigen Vermittlung von „Positionen“ und „Konstruktion“ zwischen den Betroffenen gelingen. Diese Aufgabe kommt seit einigen Jahren auch dem informatischen Forschungsgebiet Usability Engineering (Gebrauchstauglichkeit von Informationstechnologien) zu, welches in der Arbeit von Frau Eller einem bedeutenden und vor allem auch praxisrelevanten Fortschritt zugeführt wurde.

Das vorliegende Buch führt in didaktisch gelungener Weise in das Usability Engineering sowie das sonstige „Drumherum“ ein und versucht in klar verständlicher Form aufzuklären. Es eignet sich wegen seiner systematischen Information sowohl für den fachlich schon tiefer Informierten als auch für diejenigen, die als Entscheidungsträger etwas mehr über die (sprach-)theoretischen Hintergründe des Usability Engineering wissen sollten. Diese Weiterentwicklung des Usability Engineering kann gerade dem, der den sprachtheoretischen Grundlagen – beispielsweise von Schema und Ausprägung – etwas ferner steht, Nutzen bringen.

Dazu macht Frau Eller einige Prinzipien der benutzerfreundlichen Anwendungsentwicklung auf der Basis einer gemeinsamen Sprache explizit. Zu ihnen gehören beispielsweise eine „transsubjekte“, d.h. nicht nur auf den eigenen Vorteil bedachte, klare und verständliche Kommunikation zwischen Nutzer- und Technologiewelt in den Unternehmen, sowie eine auf „verträglichen Zwecken“ beruhende Entwicklung von Lösungen des Einsatzes neuer Informationstechnologien. Diese Prinzipien, die aus der sprachbasierten Informatik, wie sie von Ortner und Wedekind vertreten wird, herrühren, münden in ein um Usability Engineering Aspekte erweitertes Multipfad-Vorgehensmodell, mit dem alle Unternehmen sich systematisch auf den neuen Erfolgskurs – ein konsequentes, von alles Beteiligten im Unternehmen praktiziertes Service- und Prozessdenken – begeben können.

Der Autorin, Brigitte Eller, ist es gelungen, einen gut verständlichen „state of the art“ sowie seine Weiterentwicklung zu einer interdisziplinären „Vermittlungsmethodologie“ zwischen Anwenderwelt und Informationstechnologiewellt – allgemein als Usability Engineering bezeichnet – zu schreiben. Der Nutzen dieses Buches würde deutlich gesteigert, wenn es die Verantwortlichen in Wirtschaft und Verwaltung selber lesen

und erst dann an ihre Fachleute weitergeben würden. Der etwas umfangreiche aber präzise Titel „Usability Engineering in der Anwendungsentwicklung / Systematische Integration zur Unterstützung einer nutzerorientierten Entwicklungsarbeit“ sollte gerade diesen Personenkreis nicht davon abhalten, auf diesem Gebiet auch etwas Fachwissen zu erwerben, obwohl solches – wie ich aus Wissenschaftskreisen weiß – gelegentlich die Entscheidungsfreude eher trübt als fördert.

Prof. Dr. Erich Ortner
Entwicklung von Anwendungssystemen
Technische Universität Darmstadt

Vorwort

Die Notwendigkeit, Benutzer bereits früh in den Softwareentwicklungsprozess einzubeziehen, gewinnt an Bedeutung. Produkteigenschaften werden dadurch beeinflusst, eine spätere Kostenexplosion für ein (abermaliges) Re-Engineering eines Anwendungssystems kann vermieden werden. Diese Situation schafft Nährboden und Raum für neue Konzepte zur in der Entwicklungsarbeit, wie die Agile Entwicklung das End-User-Development, um nur zwei Ausprägungen zu nennen. Teildisziplinen wie das Requirements Engineering etablieren sich. Das Usability Engineering rückt ins Zentrum der Vermittlung zwischen der Welt der Informationstechnologie und der Anwenderwelt.

Bereits Mitte der 1990er Jahre widmete die Software-Industrie der Usability und dem User Centered Design zunehmende Aufmerksamkeit. Dennoch gab es bisher kein Konzept, in dem das Usability Engineering konzeptionell und semantisch in die Entwicklung von Anwendungssystemen integriert ist, sodass ein Entwickler daraus direkt systematische und methodische Unterstützung erhalten kann. Die vorliegende Positionierung des Usability Engineering in Verbindung mit der sprachkritischen Entwicklungsarbeit eröffnet neue Perspektiven in der Zusammenführung unterschiedlicher Disziplinen über den gesamten Entwicklungszyklus.

Mit der sprachkritischen Rekonstruktion werden Probleme der Praxis von Unklarheiten bereinigt. Der Benutzer ist unmittelbar beteiligt. Das Ergebnis ist die Beschreibung eines Anwendungssystems in nomierter Sprache. Das gestattet den unmittelbaren Übergang zur entwicklersprachlich ausgerichteten Modellierung und Implementierung. Mit der Stützung der Nutzer wird das Anwendungssystem in den Anwendungsbereich zurückgeführt und stabilisiert. Die Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik folgt dem konstruktivistischen Ansatz, „aus dem Gebrauch – für den Gebrauch“. Sie pflegt einen konstruktiven und zugleich pragmatischen Umgang mit Sprache. Die semantische Integration über Sprache ist Grundlage für die konzeptionelle, technische und praktische Annäherung zwischen der IT-Welt und der Anwender-Welt. Die dargestellte nahtlose Integration verspricht, in ihrer Wirkung nachhaltig zu sein.

Das Buch richtet sich gleichermaßen an Praktiker und Theoretiker. Der vorliegende Ansatz kann branchenunabhängig zum Einsatz gelangen und bringt wichtige Impulse für die benutzerorientierte Entwicklungsarbeit. Es wird gezeigt, wie eine gemeinsame

Sprache für den Anwendungsbereich systematisch geschaffen werden kann. Diese gemeinsame Sprache ist Vermittler zwischen Beteiligten und Disziplinen. Für entwicklungsrelevante Probleme wird damit das erforderliche Verständnis erreicht. Die planmäßige Entwicklungsarbeit wird weiter unterstützt durch das erweiterte Multipfad-Vorgehensmodell. Es erlaubt ein situativ adaptierbares und damit flexibles Vorgehen während des Entwicklungsprozesses, fördert die Einbeziehung der Benutzer und gewährleistet gleichzeitig systematisches Vorgehen. Methodenbeschreibungen innerhalb des vorgeschlagenen Ordnungsrahmens begünstigen die Vernetzung der Entwicklungsarbeiten.

An dieser Stelle danke ich all jenen Menschen herzlich, die zum Gelingen dieser Arbeit beigetragen haben. Allen voran gilt dieser Dank meinem Doktorvater, Herrn Prof. Dr. Erich Ortner, für die fachliche und wissenschaftliche Betreuung über viele Jahre. Seine konstruktiv-kritische Sichtweise auf die Entwicklungsarbeit und die damit verbundenen Anregungen und Diskussionen trugen wesentlich zum Erfolg dieser Arbeit bei. Danken möchte ich auch Herrn Prof. Dr. Ulrich Frank für die Übernahme des Korreferats.

Dem Vorstand und dem gesamten Team der DEG Rhein Main e.G. danke ich dafür, dass sie praktische Erfahrungen zum vorliegenden Konzept ermöglicht haben. Den Studentinnen und Studenten, die sich in Praxisprojekten engagiert haben, gebührt Dank dafür, dass sie durch ihre Arbeit und viele Hinweise meine wissenschaftliche Arbeit vorangetrieben haben. Meine Freundin Annemarie Müller hat den gesamten Text korrekturgelesen und mit zahlreichen Vorschlägen dessen Verständlichkeit verbessert.

Besonders danke ich meinen Kindern Irmgard und Oskar sowie meinem Mann Norbert. Ihre Geduld und ihr Verständnis waren mir eine unschätzbare Hilfe.

Brigitte Eller

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abb.	Abbildung
ACM	Association for Computing Machinery
AG	Aktiengesellschaft
ANSI/SPARC	American National Standards Institute / Standard Planning and Requirement Committee
ARIS	Architektur integrierter Informationssysteme
AUCSL	Agile User Centered Software LifeCycle
Aufl.	Auflage
BCG	Boston Consulting Group
BMM	Business Motivation Model
BPEL	Business Process Execution Language
BPM	Business Process Management
BPML	Business Process Modeling Language
BPMN	Business Process Modeling Notation
BWL	Betriebswirtschaftslehre
bzw.	beziehungsweise
C4ISR	Command, Control. Computers, Communications, Intelligence, Surveillance and Reconnaissance
ca.	zirka
CHI	Computer-Human-Interaction
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
Cobit	Control Objectives for Information and Related Technology
CORBA	Common Object Request Broker Architecture
COSO	Committee of Sponsoring Organizations of the Treadway Commission
CRUISER	Cross Discipline User Interface and Software Engineering LifeCycle
CSCW	Computer Supported Cooperative Work oder Computer Supported Collaborative Work
C-TAM-TBP	Combined Technology and Planned Behavior Model
CURE	Center for Usability Research and Engineering
C #	C-Sharp (Programmiersprache für die .NET Plattform)
DarWin	Darmstädter Wissensmanagement-Konzept
DATech	Deutsche Akkreditierungsstelle Technik e.V.
d.h.	das heißt

DIN	Deutsches Institut für Normung
DSDM	Dynamic System Development Method
EC	European Commission
EJB	Enterprise Java Beans
EN	Europäische Norm
ERP	Enterprise Resource Planning
€	Euro
FEAF	Federal Enterprise Architecture Framework
GABEK	Ganzheitliche Bewältigung von Komplexität
ggf.	gegebenenfalls
GOMS	Goals, Operators, Methods and Selection rules
HCI	Human Computer Interaction
HGB	Handelsgesetzbuch
IBM	International Business Machines Corporation
IDT	Innovation Diffusion Theory
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
INTERREG	Gemeinschaftsinitiative des Europäischen Fonds für regionale Entwicklung (EFRE) zur Förderung der Zusammenarbeit zwischen den Regionen der Europäischen Union
IS	Informationssystem bzw. Information System
ISO	International Organization for Standardization
i.S.v.	im Sinne von
IT	Informationstechnologie
i.ü.S.	im übertragenen Sinn
i.w.S.	im weiteren Sinn
J2EE	Java Plattform Enterprise Edition
JBOSS	ein in der Programmiersprache Java geschriebener, freier Anwendungsserver
KMU	Kleine und mittlere Unternehmen
LNCS	Lecture Notes in Computer Science
M1, M2, M3	Bezeichnungen der Ebenenstruktur der Metamodellierung im Rahmen der MDA
MaxDB	Name eines relationalen Datenbankmanagementsystems der Firma SAP
MCI	Management Center Innsbruck
MDA	Model Driven Architecture
MI	Man Machine Interface
MM	Motivational Model
MPCU	Model of PC Utilisation

MySQL	freies relationales Datenbankverwaltungssystem, ursprünglich von der schwedischen Firma MySQL AB entwickelt
o.a.	oben angeführt(e)
OMG	Object Management Group
OSM	Organizational Structure Metamodel
PostgreSQL	freies, objektrelationales Datenbankmanagementsystem
ProCEM®	Process-Centric Enterprise Modeling & Management
RUP	Rational Unified Process
S.	Seite
s.o.	siehe oben
SAP	Systemanalyse und Programmentwicklung
SBVR	Semantics of Business Vocabulary and Business Rules
SCT	Social Cognitive Theory
SEI	Software Engineering Institute
SIGCHI	Special Interest Group on Computer-Human-Interaction
SOA	Service-orientierte Architektur steht auch synonym für die Serviceorientierung in Organisationen generell
sog.	sogenannt
SWEBOK	Guide to the Software Engineering Body of Knowledge
Tab.	Tabelle
TAM	Technology Acceptance Model
TOGAF	The Open Group Architecture Framework
TPB	Theory of Planned Behavior
TR	Technical Report
TRA	Theory of Reasoned Action
TS	Technische Spezifikation
u.a.	unter anderem bzw. unter anderen
u.a.m.	und andere mehr bzw. und anderes mehr
UCD	User Centered Design
UE	Usability Engineering
überarb.	überarbeitete
UI	User Interface
UML	Unified Modeling Language
UMM	Usability Maturity Model
US	United States
USECON	Name eines SpinOffs des renommierten Forschungszentrums CURE
usw.	und so weiter
UTAUT	Unified Theory of Acceptance and Use of Technology

V1, V2, ...	Nummerierung von Vorgangstypen im Multipfad-Vorgehensmodell
vgl.	vergleiche
V-Modell	Vorgehensmodell
WinRelan®	Windows® Relationen Analyse
XML	Extensible Markup Language
XP	Extreme Programming
XU	Extreme Usability
z.B.	zum Beispiel

1 Einleitung

1.1 Ausgangssituation und Problemstellung

Die Auseinandersetzung der Softwareingenieure mit Benutzern von Anwendungssystemen erfolgt in der Softwareentwicklung zuverlässig nach Fertigstellung des Systems, d.h. in der Testphase, oder im Zuge der Einführung des Systems in der *Anwendungsumgebung*. Für die Benutzer ist dies oft die erste Konfrontation mit einer neuen Software. Sie werden dann für die neue Software oder für die Handhabung von Systemanpassungen geschult und vor vollendete Tatsachen gestellt. Im Vorfeld der Einführung waren Softwareingenieure und bestenfalls Repräsentanten der Anwendungsbereiche am Projekt beteiligt, nicht aber die eigentlichen Benutzer.

Werden Mitarbeiter nicht in ein Entwicklungsprojekt miteinbezogen und somit als Benutzer vor vollendete Tatsachen gestellt, suchen sie (trotz vorheriger Schulung) in der Einführungsphase nach gewohnten Menüpunkten, Funktionen, Inhalten und Abläufen. Die einzelnen Arbeitsschritte sind möglicherweise nicht mehr so angeordnet, wie sie das gewohnt waren. Es erscheinen Fehlermeldungen, die sie nicht einzuordnen wissen. Die im System angebotene Hilfe empfinden sie als unzureichend bzw. zu zeitaufwändig. Diese Mitarbeiter werden zu Daueranrufern bei der Hotline und fühlen sich bald als Versuchskaninchen. Für ihre Arbeit benötigen sie mit der neuen Software länger, nicht zuletzt wegen der vielen Kontakte zur Hotline. Die Akzeptanz für ein Anwendungssystem sinkt nach Einführung zusehends, die Qualität der Arbeit sinkt womöglich ebenfalls. Diese Aspekte – und es ließen sich diese Geschichten noch erweitern – führen zu Unzufriedenheit, Ineffizienz und möglicherweise auch zu Ineffektivität. Die Gebrauchstauglichkeit (Usability) des Systems ist nicht ausreichend. Mit Folgekosten in oft nicht unerheblichem Ausmaß ist zu rechnen (Donahue, Weinschenk & Nowicki, 1999).

Zunehmend wurde die Forderung lauter, Benutzer bereits viel früher in den Softwareentwicklungsprozess einzubeziehen, um Produkteigenschaften beeinflussen zu können und damit spätere Kostenexplosion z.B. für ein (abermaliges) Re-Engineering eines (Anwendungs-)Systems zu vermeiden (z.B. Mayhew, 1999a). Diese Forderungen schafften Nährboden und Raum für neue Konzepte der Softwareentwicklung wie z.B. die Agile Entwicklung (z.B. Cockburn, 2003; Agile Alliance, 2008) aber auch das *End-User-Development* (z.B. Liebermann, Paterno & Wulf, 2006). Ebenso entwickelten sich „neue“ Disziplinen wie z.B. das Requirements Engineering (z.B. Schienmann, 2002; Rupp, 2007) und das Usability Engineering (z.B. Mayhew, 1999a; Rosson & Carroll, 2002; DATech, 2008). In allen genannten Konzepten und Disziplinen spielt die

Integration der künftigen Benutzer in die Entwicklung einer Anwendung eine zentrale Rolle. Eine systematische und nachhaltig wirksame Integration hat aber bis dato nicht stattgefunden. Im Gegenteil, es hat den Anschein, dass jede Disziplin eigenständig Fuß fassen will.

Es ist längst widerlegt, dass Aktivitäten des Usability Engineering zu Mehrkosten in der Entwicklungsarbeit führen (z.B. Landauer, 1995; Donahue et al., 1999). Je früher in einem Projekt Usability-Methoden eingesetzt werden, desto weniger Kosten entstehen, da die meisten Probleme schnell erkannt werden und in frühen Phasen mit wenig Aufwand zu beheben sind (Jacobsen, 2006).

Mayhew (1999a, S. 431 u. Synopsis) betont, dass aufgabenspezifisch ausgerichtete und in eine existierende Software Engineering Methodologie nahtlos integrierte Aufgaben des Usability Engineering extrem produktiv sein würden:

> A commitment to usability in user interface design and development offers enormous benefits, including greater user productivity, more competitive products, lower support costs, and a more efficient development process.

Trotzdem die Software-Industrie bereits Mitte der 1990er Jahre der Usability und dem User Centered Design zunehmende Aufmerksamkeit widmete (z.B. Butler, 1996; Trenner & Bawa, 1998), blieb die Integration von benutzerzentrierten Methoden in bestehende Entwicklungslebenszyklen bis heute eine erhebliche Herausforderung. In der Betriebswirtschaft ist die systematische Beeinflussung und Steuerung von Produktkosten in einem sehr frühen Stadium der Produktentwicklung etabliert und wird z.B. unterstützt durch die Methoden des Target Costing und der dynamischen Lebenszykluskostenrechnung.[1] Im Vergleich dazu erfreut sich die systematische Beeinflussung von Systemeigenschaften zur Verbesserung der Akzeptanz und Gebrauchstauglichkeit von einem sehr frühen Stadium der Anwendungsentwicklung bis zur Einführung und Nutzung eines Anwendungssystems einer bis dato noch zu geringen Aufmerksamkeit (Nielsen, 1993, S. 3). Es ist aber grundsätzlich nicht neu, dass versucht wird, das Software-Engineering und die Aktivitäten der Human-Computer-Interaction zu integrieren (z.B. Daimler Benz, 1998; Gundelsweiler, Memmel & Reiterer, 2004; Battle & Lockheed, 2005; Blomkvist, 2005; Gulliksen, Göransson, Boivie,

1 Target Costing ist ein vor ca. 35 Jahren bei Toyota in Japan entwickelter Ansatz des Kostenmanagements, der bestehende Instrumente, wie z.B. den Transaktionskostenansatz, als zentrales Konzept einer ökonomischen Organisationstheorie und effizienzorientiertes Entscheidungskriterium (Picot, 1982, S. 268; Picot & Dietl, 1990, S. 178), mit der (dynamischen) Lebenszykluskostenrechnung (Britzelmaier & Eller, 2004) sowie mit einer konsequenten Marktorientierung des Unternehmens (Britzelmaier, 1999, S. 41-43; Horvath, Gleich & Voggenreiter, 2001, S. 235) kombiniert.

Persson, Blomkvist & Cajander, 2005; Seffah, Desmarais & Metzker, 2005b). Dennoch gibt es bisher kein Modell, in dem das Usability Engineering konzeptionell und semantisch in die Entwicklung von Anwendungssystemen integriert ist, sodass einem Entwickler daraus direkte, systematische und methodische Unterstützung erwachsen kann.

Die Notwendigkeit der Integration des Usability Engineering in die Anwendungssystementwicklung ist längst erkannt (z.B. Boehm, 1991; Landauer, 1995; Mayhew, 1999a; Donahue, 2001; Seffah et al., 2005a). Dieser Herausforderung wollen wir uns aus Sicht der sprachbasierten Informatik nähern und einen Ansatz für eine systematische Integration des Usability-Engineering in die Anwendungssystementwicklung aufzeigen, der sowohl eine inhaltliche als auch eine strukturelle Zusammenführung unterstützt. In Erweiterung zu bereits genannten Notwendigkeiten soll damit auch die Etablierung eines *Acceptance Engineering* voranzutreiben sein. Die Kernfragen zur vorliegenden Problemstellung lauten also:

- Wie kann das Usability-Engineering innerhalb der Anwendungssystementwicklung systematisch unterstützt werden?
- Wie kann ein Vorgehensmodell (Prozessmodell) derart (flexibel) gestaltet werden, dass diese Integration hinreichend institutionalisiert ist und dennoch situativ adaptiert werden kann?
- Stellt eine systematische Integration des Usability Engineering in die Entwicklung von Anwendungssystemen eine Verbesserung für den Entwicklungsprozess dar und zwar im Hinblick auf die Steuerung von Produktakzeptanz und Gebrauchstauglichkeit?

Die Anwendungsentwicklung findet im Rahmen von Mensch, Organisation und Technik statt. Für das Herangehen an die vorliegenden Fragestellungen liegt der Fokus auf der nutzerzentrierten Entwicklungsarbeit, also auf dem Menschen. Technische Aspekte der Integration des Usability Engineering werden ebenso ausgeklammert wie der organisationale Aspekt z.B. in Ausprägung der Beeinflussung organisationaler Akzeptanz. Integration und Akzeptanz im Hinblick auf den Menschen steht im Vordergrund, im Sinne von „Fitting the task to the human“, wie Kroemer & Grandjean (2003) das Credo für ergonomische Gestaltung im beruflichen Umfeld bezeichnen. Im Sinne von Mayhew geht es um eine „nahtlose“ Integration des Usability Engineering in die Entwicklungsarbeit zwecks Arbeitsgestaltung zur Aufgabenerfüllung mittels Anwendungssystemen.

Für die Bewältigung der vorliegenden Herausforderungen werden nachfolgend Arbeitsprinzipien festgelegt sowie Aufbau und Gang der Arbeit skizziert. Mit der Fundierung der Arbeit im Konstruktivismus ist die Grundlegung der methodischen Vor-

gehensweise bestimmt. Mit der Grundlegung im Konstruktivismus erfolgt zudem die Einordnung der vorliegenden Arbeit in Relation zu den „Hauptströmungen“ der Systementwicklung in der Wirtschaftsinformatik, dem gestaltungs- bzw. gebrauchsorientierten Ansatz und dem verhaltensorientierten Ansatz (vgl. Abschnitt 1.4.1, S. 8f).

1.2 Arbeitsprinzipien

1.2.1 Transdisziplinarität

> Transdisziplinarität ist die Kurzformel für einen weitreichenden Vorgang: Transdisziplinäre Wissenschaft verlässt ihren angestammten akademischen Ort, geht an die Orte der gesellschaftlichen Praxis, tritt in die soziale Wirklichkeit, um hier Zustände und Vorgänge zu entdecken, zu erkennen, die der wissenschaftlichen Rationalität im Allgemeinen verschlossen sind. Wissenschaft begibt sich hier in eine ungewohnte, subtile Ordnung, muss ihre Erwartungen, ihren Habitus, ihre Sprache dieser Ordnung anpassen. In der sozialen Wirklichkeit gerät Wissenschaft in jene Verteilungskämpfe; sie mischt sich ein, wird politisch, hat sich in ungewohnter Weise zu legitimieren, wird wirksam in der Gestaltung gesellschaftlicher Normen.
>
> (Nicolini, Freyer, Zinggl & Treusch-Dieter, 2002)

Sowohl bei der Anwendungssystementwicklung als auch beim *Usability Engineering*, verstanden als Teilbereich des *Acceptance Engineering*, handelt es sich um transdisziplinäre Wissenschaftsbereiche, denn ohne in die soziale Wirklichkeit des Arbeitsgeschehens einzutreten, ist ein entdecken und erkennen der für die Entwicklung und die Schaffung von *Akzeptanz* und Gebrauchstauglichkeit relevanten Zustände und Vorgänge undenkbar. Letztere mischen sich als subtiles Ordnungsgefüge ein in die Gestaltung der Arbeit (Hacker, 1987). Dementsprechend sind Erwartungen, Auftreten und Sprache dem jeweiligen Gefüge (Anwendungsbereich) anzupassen. Die geschaffenen, gebrauchstauglichen und akzeptierten Anwendungssysteme manifestieren schließlich organisationale Normen.

Als Terminus der neueren Wissenschaftstheorie dient der Begriff der *Transdisziplinarität* zur Charakterisierung von Forschungsformen, die problembezogen über die – historisch bedingt divergente – fachliche und disziplinäre Konstitution der Wissenschaft hinausgehen, vor allem hinein in die soziale Wirklichkeit, wie es Nicolini (2001) zum Ausdruck bringt. Im Gegensatz zur Interdisziplinarität hält die Transdisziplinarität nicht an den Fachgrenzen sowie Grenzen zwischen Disziplinen fest. Es sollte damit keine Gefahr drohen, dass Grenzen von Fächern und Disziplinen zu Erkenntnisgren-

zen werden (Mittelstraß, 2004c, S. 329). Mit dem Prinzip der *Transdisziplinarität* als Entwicklungs- und Forschungsprinzip soll eine Erweiterung der wissenschaftlichen Wahrnehmungsfähigkeit und Problemlösungskompetenz, trotz großer Spezialisierung in manchen Bereichen, gewährleistet bleiben. In Bezug auf die geltende Praxis hat diese Aussage präskriptiven Charakter. Der deskriptive Charakter der Transdisziplinarität von Forschungsvorhaben äußert sich in der Beschreibung von beteiligten Disziplinen, basierend auf ihrem historischen Konstitutionszusammenhang, mit Ausdruck der erreichten Spezialisierung. Erst wenn es gelingt, die deskriptive und präskriptive Sicht im Zeitverlauf zu vereinen, kann Forschungs- und Entwicklungsarbeit der Transdisziplinarität im beschriebenen Sinn gerecht werden. Ein Ausgangspunkt dieser möglichen Entwicklung für die Praxis der Wirtschaftsinformatik ist ihr interdisziplinäres Profil (vgl. z.B. Hansen & Neumann, 2005; Lassmann, 2006).

Den Charakter der vielfältigen Aufgabenstellungen in der Entwicklung von Anwendungssystemen in Verbindung mit einer geforderten Transdisziplinarität der Wirtschaftsinformatik verdeutlicht Ortner (2005, S. 51-53). In diesem Kontext ist mit Transdisziplinarität gemeint, dass die Kooperation verschiedener Disziplinen mit der Zeit zu anderen neuen, methodischen und inhaltlichen Orientierungen in einem gemeinsamen Wissenschaftsgebiet führen kann, womit sich naturgemäß auch fachliche und disziplinäre Orientierungen verändern können. Das Verständnis für Disziplinen schließt hier auch die Anwendungsbereiche in Unternehmen und anderen Organisationen als relevante Wissensgebiete für die Entwicklungsarbeit mit ein. Transdisziplinarität wird dort wirksam, wo alleine fachliche oder disziplinäre Definitionen von Problemlagen und Lösungsansätzen nicht möglich sind. Sie ist daher als ein integratives (zusammenführendes) aber nicht generalisierendes (überwissenschaftliches) Konzept zu verstehen. In diesem Sinne wird Transdisziplinarität als ein Arbeits- und Entwicklungsprinzip dieser Arbeit zugrunde gelegt.

1.2.2 Verständliche Sprache

> Ein guter, verständlicher Text – mit dem Anspruch der Wissenschaftlichkeit – erhält seine Leichtigkeit, seine Schlichtheit aus dem genauen, gerechten Sprachgebrauch, aus der Hingabe an die Sprache, in die der Forschungsgegenstand übersetzt wird. Diese Hingabe ist zugleich eine Hommage an die Leser und Leserinnen, denen ein guter Text gebührt.
>
> (Nicolini et al., 2002)

Nicolini (2001, S. 21) bezeichnet Sprache als Quelle inter- und transdisziplinärer Wissenschaft, als Wurzel der Erkenntnis, Erfahrung und Kommunikation und meint damit nicht eine reduzierte Alltagssprache, wie sie heute mit verkürzten Reden des Alltags

weit verbreitet ist, sondern die gesamte, reich ausgestattete Gebrauchssprache (= Umgangssprache im Sinne von Kamlah, 1975).

Sprache als Quelle und Wurzel wissenschaftlicher Erkenntnis, Erfahrung und Kommunikation umfasst die Gebrauchssprache und in Erweiterung zu Nicolini auch Fachsprachen und Sondersprachen. Als Fachsprachen bezeichnen wir besondere Formen der Ausdrucksweise in einzelnen Disziplinen. Als Sondersprachen können wir Kunstsprachen wie z.B. Diagrammsprachen verstehen. Dieser erweiterte Sprachraum, in dem sich sowohl die Wissenschaft als auch die Anwendungsentwicklung bewegen, bedarf kontinuierlicher Aufmerksamkeit, da Sprache konstitutiv ist. Das heißt, in der Sprache entsteht Wissen und Wirklichkeit (Nicolini, 2001, S. 21). Forschungs- und Entwicklungsarbeit wird als evolutionärer Prozess verstanden, unabhängig davon, welche Disziplinen am jeweiligen Forschungs- bzw. Entwicklungsprozess beteiligt sind. Durch diese Art der Arbeit in einem traditionsfreien Raum, in dem man sich von Gewohntem losspricht, können neue Sinnzusammenhänge gewonnen werden. Dazu sind Sprachkompetenz und Sprachperformanz (i.S.v. Ortner, 2005) gefordert, um die geschaffene Wirklichkeit für jedermann lesbar und verstehbar zu machen.

Die verständliche Sprache wird als Arbeitsprinzip verankert. Sie wird damit nicht nur in der methodischen Vorgehensweise innerhalb der Entwicklung von Anwendungssystemen im Rahmen der sprachbasierten Informatik behandelt (hier ist sie durch die Fundierung im Konstruktivismus inhärent), sondern es besteht die Forderung, dass die vorliegende Arbeit ebenso diesem Prinzip folgt. Die verständliche Sprache als Arbeitsprinzip ist kein Rezept, das sich benützen ließe wie Fertigware. Es ist vielmehr permanente Herausforderung, sich einfach und klar auszudrücken. „…ein wissenschaftlicher Text muss grammatisch richtig und syntaktisch im Gleichgewicht sein, präzise in der Wortwahl und in den Positionierungen, angemessen im Stil, stimmig in der Choreografie.“ (Nicolini, 2001, S. 44-45).

Mit der Anwendung des Arbeitsprinzips der verständlichen Sprache nähern wir uns auch der Sprachkritik Kamlahs (vgl. Abschnitt 2.1.2, S. 17ff). Er forderte die Abschaffung der sogenannten „Bildungssprache“ und begründete dies damit, dass der Gebrauch der bisherigen Bildungssprache enorme Schwierigkeiten der Verständigung bewirkt. Unter diesen Verständigungsschwierigkeiten leider wir auch heute noch. Dies ist „in fast allen Bereichen nicht allein der Philosophie und der Wissenschaft, sondern auch der Literatur, der Kunstkritik, der Politik…“ der Fall. Es handelt sich dabei um jenes Aneinander-Vorbeireden auf noch so hohen Podien von „Podiumsdiskussionen“, jene babylonische Sprachverwirrung gerade unter den „Gebildeten“, den „Intellektuellen“, gegen die wir endlich angehen sollten.

1.3 Aufbau und Gang der Arbeit

Eine Orientierung für das Integrationsvorhaben gibt das Ebenenmodell zur systematischen Integration des Usability Engineering in die Anwendungssystementwicklung (Tab. 1). Das Modell lehnt sich an das 3-Ebenen-Modell nach ANSI-SPARC an. Elemente der internen Ebene bilden die Grundlage für Integrationsarbeiten auf höheren Ebenen. Im Sinne einer konstruktivistischen Grundlegung der Arbeit sind dies Elemente und Erkenntnisse der Formal- und Sprachwissenschaften in Ausprägung der logischen Propädeutik von Kamlah & Lorenzen (1967).[2]

Auf der konzeptionellen Ebene spielen sowohl Erkenntnisse der Natur- und Technikwissenschaften als auch der Sozial- und Kulturwissenschaften eine Rolle. Die Betrachtung von technischen Aspekten zur vorliegenden Problemstellung, wie z.B. die Umsetzung einer Integration auf IT-Plattformen, wird unterbleiben. Das *Enterprise Engineering* wird den Sozial- und Kulturwissenschaften zugeordnet. Es ist in Verbindung mit relevanten Informationstechnologien eine Ausprägung einer systematisch mensch- und organisationszentrierten Anwendungsentwicklung im Sinne des erweiterten Ebenenmodells der Informationsverarbeitung nach Ortner (2003b).

	Orientierungsbereich	**Orientierungsgrundlage**	**Art der Integration**
Externe Ebene	Anwendungsbereiche	Normen aus Legislatur und Praxis	praktisch
Konzeptionelle Ebene	Sozial- und Kulturwissenschaften	Enterprise Engineering	strukturell
	Natur- und Technikwissenschaften	IT-Plattformen und IT-Technologien	technisch
Interne Ebene	Formal- und Sprachwissenschaften	Logische Propädeutik	semantisch

Tab. 1: Ebenen der systematischen Integration des Usability Engineering in die Entwicklung von Anwendungssystemen

Anwendungssysteme werden immer für einen bestimmten Zweck in einem organisationalen Kontext (= Anwendungsbereich) geschaffen. Innerhalb dessen bilden gesetzliche Vorschriften, übergeordnete Normen sowie organisationale Regelwerke (z.B. IT-*Governance*) einen Rahmen, der für die Entwicklungsarbeit maßgebend ist. Organisationale Regelwerke besitzen integrativen Charakter im Übergang von der konzeptionellen zur externen Ebene (Claessens, 2006).

In Abschnitt 0 erfolgt die Positionierung der Arbeit. Die wissenschaftstheoretische Fundierung im Konstruktivismus schafft das Verständnis zur (Re-)Konstruktion von Arbeit als zweckgerichtetes Handeln. Dieses wird in Abschnitt 2 durch relevante

2 Diese wurde später von Lorenzen (2000) in seiner konstruktiven Wissenschaftstheorie aktualisiert.

Grundlagen ergänzt und vertieft. Die eingeführten Grundlagen können gleichermaßen Zweck und Mittel betreffen aber auch selbst Zweck oder Mittel zur Entwicklung von Anwendungssystemen sein.

Die Anwendungsentwicklung ist interdisziplinär. Sie wurzelt u.a. in der Wirtschaftsinformatik, der Informatik und in der Betriebswirtschaft. Mit dem Streben nach Transdisziplinarität überragt sie aber die einfache Zusammenlegung von Wissensgebieten der genannten Disziplinen (vgl. Abschnitt 1.2.14). Die Anwendungsentwicklung aus Sicht der sprachbasierten Informatik (Abschnitt 3) gestattet einen grundlegenden, d.h. in diesem Fall einen sprachbasierten Zugang zur Problemstellung. Dieser Zugang ermöglicht die systematische und benutzerorientierte Integration des Usability Engineering in die Anwendungsentwicklung auf der internen Ebene. Die sprachbasierte Integration wird durch eine strukturelle Integration ergänzt. Schritt für Schritt werden dafür *Modelle* aus dem Usability Engineering und der Akzeptanzforschung mit dem Multipfad-Vorgehensmodell zusammengeführt. Der Fokus liegt dabei auf jenen Phasen, in denen der Anwender im Mittelpunkt stehen soll. Dies sind die frühen Phasen der Anwendungsentwicklung und die Stützung der Nutzer im Rahmen der Einführung eines Systems. Abgerundet wird die Integration über die Zusammenführung von Methoden auf der externen Ebene. Ein Ordnungsrahmen zur integrierten Methodenbeschreibung bietet eine Stützung der Entwickler in der Umsetzung einer nahtlosen Integration. Mit einem Fazit und einem Ausblick (Abschnitt 4) wird die systematische Integration des Usability Engineering in die Anwendungsentwicklung aus Sicht der sprachbasierten Informatik abschließend bewertet und positioniert.

1.4 Fundierung im Konstruktivismus

1.4.1 Gegenüberstellung von Denkansätzen

Für die Gegenüberstellung von Denkansätzen zum Zweck der wissenschaftstheoretischen Fundierung der Arbeit wurde auf einen Vergleich des verhaltensorientierten und des gebrauchsorientierten Forschungsansatzes von March & Smith (1995) sowie Hevner, March, Park & Ram (2004) zurückgegriffen. Hevners Vergleich zwischen dem verhaltensorientierten und dem gebrauchsorientierten Ansatz bildet den Kern der Darstellung in Tab. 2 (S. 9). Die Abgrenzung ist nicht immer so klar, wie dies die tabellarische Illustration vorzugeben vermag. Die gezeigten Standpunkte für den verhaltensorientierten und den gebrauchsorientierten Ansatz entsprechen jenen von Hevner et al. Die Gegenüberstellung wurde erweitert um den konstruktivistischen Forschungsansatz.

Verhaltensorientierte (behavioristische) Ansätze konzentrieren sich auf die Verwendung von existierenden Anwendungssystemen und deren Auswirkung auf Individuen, Gruppen und Organisationen. Der Ansatz ist als eher passiv einzuordnen, da lediglich von einer Erkenntnisgewinnung ausgegangen und eine aktive Einflussnahme nicht berücksichtigt wird. Es wird dabei kaum auf die Potenziale von Technologien zur Lösung von organisatorischen Problemen eingegangen. Auch wurden diese Ansätze im Hinblick auf ihre Praxisrelevanz stark kritisiert. Es war in diesem Zusammenhang sogar von „unangemessenen Forschungsmethoden" oder „mangelnder technologischer Kompetenz" die Rede (Frank, 2003, rezensiert den Artikel von Kock, Gray, Hoving, Klein, Myers & Rockart, 2002). Für die Erkenntnisgewinnung gerade im Zusammenhang mit der Gebrauchstauglichkeit von Anwendungssystemen sind Methoden des Behavioral Science Research aber sehr wohl von Relevanz, da durch die Integration des Usability Engineering durchaus sozial- und kulturwissenschaftliche Ansätze relevant sein können.

	Verhaltensorientierter Forschungsansatz **Behavioral Science Research**	**Gebrauchsorientierter Forschungsansatz** **Design Science Research**	**Methodisch konstruktivistischer Forschungsansatz (Erlanger Schule)** **Methodical Constructivist Epistemological Science Research**
Ursprung	Naturwissenschaften	[Ingenieurwissenschaften][3]	Philosophie, Naturwissenschaften, Sprachwissenschaften
Denkansatz	Das Verstehen des Problems steht im Vordergrund	Problemlösungsorientierter Ansatz	Methodisch, logisch, systematische Selbstreflexion von Handlungen und Denkweisen zum besseren Verstehen von Problemen und zur Problemlösung
Ziele für IS	Untersuchung, Einsatz und Nutzung von IT-Artefakten im Anwendungsumfeld, z.B. in Organisationen	Entwicklung und Evaluation von IT-Artefakten zur Lösung organisatorischer Probleme	Ganzheitliche (sprachbasierte) Rekonstruktion von Anwendungssystemen
Betrachtungsgegenstand	Mensch-Computer-Interaktion	Gestaltung von Artefakten für Anwendungssysteme	ganzheitliche Anwendungsentwicklung umfasst alle ablauf- und aufbaubezogenen Aspekte der Entwicklung

Tab. 2: Gegenüberstellung von Forschungsansätzen

3 Hevner beruft sich auf einen Ursprung in den Ingenieurwissenschaften. Die Einschätzung ist für die Fundierung der Arbeit im Konstruktivismus unbedeutend, sodass keine weitere Erörterung dessen erfolgt.

Im gebrauchsorientierten Forschungsparadigma (synonym: gestaltungorientiertes Forschungsparadigma; Design Science Research) werden Probleme aus der Praxis wahrgenommen und aufgegriffen. Das Wissen und das *Verstehen* des jeweiligen Problems in einem Anwendungsgebiet und auch die Entwicklung des zugehörigen Lösungsansatzes werden erreicht durch die Gestaltung und Anwendung von Artefakten. Eine systematisch, wissenschaftstheoretisch fundierte Untermauerung, wie sie z.B. durch eine konstruktivistische Grundlegung erfolgen könnte, ist bei diesem Ansatz nicht zu erkennen.

Die Unterschiede im methodischen Zugang zu den Forschungsarbeiten rechtfertigen die Einbeziehung von behavioristischen Methoden, insbesondere in Verbindung mit dem Usability Engineering und ansatzweise mit Sichtweisen des Design Science Research. Der Vergleich möglicher Denkansätze für die vorliegende Problemstellung legt nahe, dass ein methodisch-konstruktivistischer Ansatz zu wählen ist, da er, zumindest was die Sicht der Anwendungsentwicklung betrifft, sowohl den Denkansatz des behavioristischen Forschungsansatzes als auch den des gebrauchsorientierten Ansatzes umfasst. Die Gegenüberstellung dieser drei Ansätze (Tab. 2) aus dem Blickwinkel der Entwicklung von Anwendungssystemen verdeutlicht dies. Die Arbeit folgt daher dem methodisch konstruktivistischen Forschungsansatz der Erlanger Schule (kurz: Erlanger Konstruktivismus)

Mit diesem Konstruktivismus als wissenschaftstheoretisches Fundament wird eine systematische, sprachbasierte Verbindung zur Empirie geschaffen, die nicht mit empirisch-behavioristischen Forschungsansätzen, wie sie im angloamerikanischen Raum im Forschungsgebiet für Information Systems (IS) als quantitativ-empirische Ansätze (Mingers, 2003; Andoh-Baidoo, White & Kasper, 2004) vorherrschend sind, vergleichbar ist. Ein Vergleich mit anderen sprachbasierten Ansätzen sowohl des IS-Research als auch der gebrauchs- bzw. gestaltungsorientierten Wirtschaftsinformatik, wie sie Fielenbach & Niehaves (2008) übersichtlich darstellen, wäre durchaus interessant, ist aber nicht Gegenstand dieser Arbeit. Der Erlanger Konstruktivismus wird im folgenden Abschnitt unter Berücksichtigung entwicklungsrelevanter konstruktivistischer Sichtweisen eingeführt.

1.4.2 Erlanger Konstruktivismus

Der Erlanger Konstruktivismus, auch Konstruktivismus der Erlanger Schule oder später Methodischer Konstruktivismus genannt, ist ein Ansatz einer allgemeinen Wissenschaftstheorie. Als Begründer der Erlanger Schule gelten Wilhelm Kamlah (1905-1976) und Paul Lorenzen (1915-1994), die den ersten einschlägigen Standardtext dazu, die *Logische Propädeutik*, herausgegeben haben (Kamlah & Lorenzen, 1967). Später folgten weitere Schriften, von Kamlah (1975), „Von der Sprache zur Vernunft“

und von Lorenzen (2000), „Lehrbuch der konstruktiven Wissenschaftstheorie", um nur zwei wichtige Werke zu nennen. Kuno Lorenz, Jürgen Mittelstraß, Peter Janich, Oswald Schwemmer und Christian Thiel gehörten zur ersten Schülergeneration von Lorenzen und Kamlah. In den 1970er Jahren bilden Jürgen Mittelstraß und Peter Janich mit Carl Friedrich Gethmann und Friedrich Kambartel die sogenannte Konstanzer Schule. In den 1980ern begründet Janich die Marburger Schule und benennt den Begriff Erlanger Konstruktivismus wegen der geografischen Veränderung und der stärkeren Betonung auf Methodik in „Methodischer Konstruktivismus" um. Heute besteht das Programm der Marburger Schule in der Weiterentwicklung des Methodischen Konstruktivismus zum „Methodischen Kulturalismus". Gleichzeitig wird mit der Umbenennung die Bindung von Wissenschaft an den (kulturellen) Kontext betont, der die Richtung und Methodik einer Wissenschaft deutlich beeinflusst.

Die Wissenschaftstheorie der Erlanger Schule gilt als konstruktivistisch. Die Wissenschaft selbst wird als zweckgerichtetes Handeln verstanden, Gegenstände der Wissenschaft werden als Konstruktionen im Sinne von Produkten zweckgerichteten Handelns erkannt. Der Konstruktivismus orientiert sich nicht stillschweigend an Prämissen und Axiomen, sondern ist gebunden an den jeweiligen Kontext und die Alltagspraxis (Lebenswelt, Anwendungsbereich) der am Handeln Beteiligten. Aus konstruktivistischer Sicht ist es die Lebenswelt des Alltäglichen, für die Anwendungssystementwicklung die Lebenswelt der jeweiligen Organisation bzw. des jeweiligen Anwendungsbereiches, aus der/dem heraus Argumentationsanfänge im Konsens und nach pragmatischer Ausrichtung schrittweise, zirkelfrei und alles explizit machend entwickelt und begründet werden. Im Rahmen der konstruktivistischen *Methodologie* werden zum Aufbau von Anwendungssystemen Begriffe dialogisch eingeführt (rekonstruiert), geprüft und schließlich als eindeutig nachvollziehbare Fachbegriffe etabliert. In Anlehnung an Petersen (1997) können drei wesentliche Merkmale des methodischen Verfahrens des konstruktivistischen Begriffsaufbaus genannt werden:

- Zweckbindung der Entwicklungstätigkeit.
- Erkennen als Handeln im Kontext.
- die dialogische Methode der Erkenntnisgewinnung.

Lorenzen geht weiter und integriert die geklärten Fachbegriffe in eine sogenannte Normsprache, er nennt eine solche Sprache auch *Orthosprache*. Sie kann nach den Regeln der Aussagenlogik und Prädikatenlogik verwendet werden (Kamlah & Lorenzen, 1967; Lorenzen, 2000). Dass die Sprachwissenschaft als Grundlagendisziplin aller Wissenschaften erkannt wurde, ist auf den sogenannten Linguistic Turn zurückzufüh-

ren[4]. Für den vorliegenden Kontext steht dabei nicht die Konstruktion einer Idealsprache zur Diskussion, sondern vielmehr die deskriptive Analyse der jeweiligen Gebrauchssprache eines Anwendungsbereiches. Dadurch wird ein Pragmatismus geschaffen, der in seine Handlungstheorie zentral die Sprachhandlung einbezieht. Eine Analyse wissenschaftlicher Verfahren beinhaltet demnach auch die Untersuchung ihrer spezifischen Zweckorientierung.

1.5 Lesehinweise

Dieser Abschnitt verfolgt den Zweck, die Gebrauchstauglichkeit der vorliegenden Arbeit für den Leser zu verbessern. Er beinhaltet Hinweise für den Leser, um ihm einen effektiven, effizienten und zufriedenstellenden Leseerfolg zu ermöglichen.

Der Text ist durchwegs in Schriftgröße 12 formatiert. Die Wahl dieser Schriftgröße erfolgt nicht zufällig. Untersuchungen in der Psychologie ergaben, dass beim Lesen eines Textes mit Schriftgröße 12 mit gesunden Augen der Inhalt unmittelbar dem „*Verstehen*“ zugeführt wird, d.h. eine weitere Auseinandersetzung auf der Gestaltungsebene (Schrift ist zu klein und schlecht lesbar usw.) unterbleibt. Die unmittelbare Auseinandersetzung mit dem Inhalt wird begünstigt.

Querverweise auf Grafiken oder Texte erfolgen zur einfacheren und schnelleren Auffindbarkeit für den Leser jeweils mit Angabe der relevanten Nummerierung und der Seitenzahl. Den Grafiken unterliegt kein Farbkonzept. Zur besseren Unterscheidung verschiedener Elemente einer Grafik werden drei unterschiedliche Grautöne verwendet, die ihrerseits keine zusätzliche Bedeutung haben. Sollte ein besonderes Verständnis für Notationen erforderlich sein oder eine Darstellung auf bestimmten Konventionen beruhen, so ist dies jeweils dem begleitenden Text zu entnehmen.

Dem Glossar wird eine besondere Rolle zuteil. Die Fundierung der Arbeit im Konstruktivismus bedingt eine zirkelfreie, nachvollziehbare und alles explizit machende Vorgehensweise. Um insbesondere Zirkelfreiheit und Nachvollziehbarkeit im Sinne des konstruktivistischen Vorgehens zu unterstützen, werden *Begriffe* im Glossar definiert und erläutert. Dies schließt eine teilweise redundante Erläuterung von Begriffen innerhalb des Textes nicht aus. Begriffe, die sich im Glossar wiederfinden, sind im Text kursiv gedruckt.

4 Eine ausführliche Herleitung und Erörterung dazu findet sich bei Heinemann (2006, S. 41-43), welche wiederum Bezug nimmt auf Graeser (2002, S. 30) sowie Bergmann, Moore, Russell, Wittgenstein, Carnap und andere.

Ein Zweck der vorliegenden Arbeit ist die Dokumentation von Forschungs- und Entwicklungsergebnissen. Erscheint das Werk als Buch oder steht es online zur Verfügung, dient es zur Verbreitung der enthaltenen Erkenntnisse - soweit die Autorensicht.

Aus „Anwendersicht“, also aus Sicht der Leser, stellt sich das naturgemäß anders dar. Will der Leser Kenntnis von den Forschungs- und Entwicklungsergebnissen bekommen, so liest er das Buch, idealerweise als Ganzes. Das Buch wird aber von einer anderen Klientel als Nachschlagewerk benutzt werden. Haben Sie schon einmal versucht, in einem Buch ohne Index, ohne Abbildungs- und Tabellenverzeichnis und ohne Namensverzeichnis zu einem Thema oder Stichwort oder zu einem bestimmten Quellenverweis etwas mit vertretbarem Zeitaufwand nachzuschlagen?

Um dafür tauglich zu sein, bedarf es der Ergänzung und Verankerung von unterschiedlichen Strukturierungen. Alle nachfolgend genannten Verzeichnisse und Ähnliches dienen diesem Zweck und sollten dem Benutzer die Handhabung als Nachschlagewerk erleichtern bzw. überhaupt erst ermöglichen. Einträge in den Verzeichnissen sind im Text referenziert.

- Inhaltsverzeichnis
- Glossar
- Abbildungsverzeichnis und Tabellenverzeichnis
- Abkürzungsverzeichnis
- Literaturverzeichnis und Index

Für den Leser sollten die Lesehinweise die Gebrauchstauglichkeit verbessern. Der Leser kann sich dieser Art besser auf die vermittelten Inhalte konzentrieren, die Akzeptanz der Lektüre steigt.

2 Grundlagen

> Wir wollen in unserem Wissen vom Gebrauch der Sprache eine Ordnung herstellen: eine Ordnung zu einem bestimmten Zweck; eine von vielen möglichen Ordnungen; nicht die Ordnung.
>
> (Ludwig Wittgenstein)

2.1 Grundsätze der sprachbasierten Informatik

2.1.1 Ausprägung einer konstruktivistischen Methodologie

Die Sprachbasierung in der Informatik wurde von Wedekind, Ortner & Inhetveen (2004a-2004e u. 2005) in einem Plädoyer für Informatik als Grundbildung manifestiert und lässt sich u.a. zurückführen auf Werke von Wittgenstein (1984), Lorenzen (2000), Kamlah & Lorenzen (1967), Kamlah (1975), Kambartel (1981), Frege (2002) und Carnap (1976, 1998). Die methodisch systematische Begründung der Sprachbasierung durch Kamlah & Lorenzen (1967) stellt einen mit rationalen Mitteln beherrschbaren Weg zur fehlerfreien Wissensrekonstruktion und Wissensbegründung dar. Diese Auffassung mündete im Erlanger Konstruktivismus. Die konstruktivistische *Methodologie* der Erlanger Schule, insbesondere wesentliche Aspekte des Verständnisses sowie die zugehörigen theoretischen Grundlagen daraus, bilden als konstruktiver Ansatz im Sinne eines „logischen Konstruktivismus" das wissenschaftstheoretische Fundament für die sprachbasierte Informatik.

Die konstruktive Wissenschaftstheorie kann daher als pädagogisch-didaktische Anleitung für Entwicklungsarbeiten und in Teilen auch für das Anwendungsmanagement verstanden werden und ist als solche bereits bei ihrer Grundlegung in der Logischen Propädeutik von Kamlah & Lorenzen (1967) verstanden worden. Alle Elemente und Regeln der Wissenschaftssprache sollen voraussetzungsfrei, zirkelfrei und nachvollziehbar eingeführt werden (Mittelstraß, 2004a, S. 551). Diese Vorgehensweise bezeichnet Heinemann (2006, S. 9 u. S. 40) als Grundpostulat konstruktivistischer Rekonstruktionsprozesse. Nach Lonthoff (2007, S. 7) ist zusätzlich darauf zu achten, dass Axiomen- und Prototypenfreiheit eingehalten wird.

Ein wesentliches Element des konstruktiven Ansatzes der sprachbasierten Informatik ist die dialogische Erkenntnisgewinnung. Der Erkenntnisprozess beginnt mit der Aufdeckung von falschen Urteilen und Scheinklarheiten durch das systematische In-

Zweifel-Ziehen von bestehenden Gewissheiten. Er dauert an bis kein plausibler Einwand gegen bestehende *Hypothesen* mehr möglich erscheint. Der gemeinsame, wechselseitig kritische Denkprozess wird von gleichberechtigten Gesprächsteilnehmern kontrolliert und überwindet die Begrenztheit der individuellen Perspektiven. Er hilft jene Barrieren für die „Wahrheitsfindung" zu überwinden, die einerseits im subjektiv für verlässlich gehaltenen Wissen bestehen, und andererseits in der Abwehr liegen, die eigene Auffassung der Kritik zu stellen. Das Merkmal der Konsensorientierung sichert daher u.a. eine symmetrische Kommunikationsstruktur, denn es soll sich nicht eine der bereits bestehenden Meinungen gegen andere durchsetzen, sondern eine gemeinsame Auffassung erarbeitet werden, mit der jeder Teilnehmer Einsichten verbinden kann, die ihm allein nicht zugänglich gewesen wären.

Damit die dialogische Erkenntnisgewinnung gelingen kann, müssen Dinge gekennzeichnet werden und Erfahrungen benannt werden. Beides sind Handlungen mit Absicht. Sprache ist hierbei sowohl Mittel als auch Zweck. Kennzeichnen passiert in Kooperation, ist Teil der Praxis und stellt soziale Bedeutung her (Konsensorientierung). Durch Kennzeichnen wird Wissen konstruiert oder rekonstruiert und in Form von Sprache repräsentiert. Sprache hat damit soziale Bedeutung. Das Verstehen der gemeinsam erarbeiteten Sprache und auch jedes andere *Sprachspiel* im Sinne von Wittgenstein und Searle (vgl. Abschnitt 5.1, S. 187) bedeuten, dass Menschen praktische Regeln bewältigen können, ohne dass sie diese selbst aufgestellt haben. Die Regeln bestehen aus Techniken, Abläufen und Konventionen, die ein nicht loslösbarer Teil der vorgegebenen Praxis sind (Cockburn, 2003, S. 313-314, mit Bezug auf Ehn, 1988).

In vielen Disziplinen ist die systematische Auseinandersetzung mit Sprache zunehmend relevant und wieder aktuell. Boas (1911a, S. 15) sagte, dass die Sprache eine der wichtigsten Manifestationen des geistigen Lebens überhaupt ist. In jeder Sprache findet man eine ihr eigene Klassifikation der „Welt". Diese Kategorisierungen (Grammatical Categories) können versierten Forschern Einblick in die Kultur der betrachteten Welt geben. Die Sprache wurde damit zu einem der lehrreichsten und aufschlussreichsten Untersuchungsfelder bei den Erhebungen Boas (1911a, S. 70). Er bezog dies auf die Untersuchung der Entstehung von fundamentalen ethnischen Ideen. Im Kontext von Organisationen kann dieses Potenzial von Bedeutung sein für die Betrachtung der Sprachkultur allgemein bzw. einer Organisationskultur im Besonderen. In der (Wirtschafts)Informatik findet die konstruktiv kritische Auseinandersetzung mit Sprache u.a. in der Anwendungsentwicklung statt, z.B. in Form einer systematischen Umsetzung von Erkenntnissen aus der Praxis in Anwendungssystemen und deren Rückführung in die Praxis durch deren Anwendung im Sinne von „aus dem Gebrauch, für den Gebrauch".

Die konstruktive Grundlegung der sprachbasierten Informatik schließt keine Ausprägung der Sprachphilosophie des 20. Jahrhunderts (*Ideal Language Philosophy* und *Ordinary Language Philosophy*) aus (Ernst, 2002, S. 79-87). Für die sprachbasierte Entwicklungsarbeit ist jede der genannten philosophischen Sichten relevant. Innerhalb der sprachbasierten Entwicklungsarbeit gilt es sogar, beide Welten anwendungsorientiert und systematisch zusammenzuführen, ausgehend von der Gebrauchssprache des Anwendungsbereiches über eine sprachliche Normierung hin zu künstlichen Sprachen, die Entwickler verwenden. Die Gebrauchssprache wird nicht als Gesamtsystem im Sinne einer von Carnap geforderten Wissenschaftssprache als „Ideal Language" entwickelt. Auf konstruktivistischer Grundlage erfolgt lediglich eine Erweiterung und teilweise eine Konkretisierung der Gebrauchssprache für einen Anwendungsbereich. Die Ergebnisse einer solchen Konkretisierung werden z.B. in einem Repository hinterlegt und sind dann allgemein verfügbar. Die Forderung nach logischer Vollständigkeit wird für einen Anwendungskontext nicht erhoben.

2.1.2 Sprachkritisches Entwicklungsparadigma

Als grundlegend und in der Anwendungsentwicklung bereits etabliert können Paradigmen wie die Strukturierung (DeMarco, 1978; Balzert, 2000, S. 431-432; Heinrich, Heinzl & Roithmayr, 2004, S. 575; Zuser, Grechenig & Köhle, 2004, S. 58-59) und die Objektorientierung (Zuser et al., 2004, S. 62; Oestereich, 1998, S. 18; Heinrich et al., 2004, S. 468; Balzert, 2000, S. 40) bezeichnet werden. Aber auch paradigmatische Vorgehensweisen, wie z.B. die evolutionäre Entwicklung, die inkrementelle Entwicklung (Scharbert, 2005; Balzert, 2000, S. 120), die iterative Entwicklung (Balzert, 2000; Sommerville, 2001; Dumke, 2003; Scharbert, 2005) oder die agile Entwicklung (Martin, 1991; Cockburn, 2003; Holzinger & Slany, 2006; Pichler, 2008; Agile Alliance, 2008) sind hier einzuordnen. Alle genannten Paradigmen und paradigmatischen Vorgehensweisen stehen orthogonal zum sprachkritischen Entwicklungsparadigma.

Seit 1978 findet diese Form der Wissenserarbeitung auch in der (Wirtschafts)Informatik Anwendung (Ortner, 2005, S. 288). Der Terminus „sprachkritisch" geht auf Ludwig Wittgenstein (1889-1951)[5] zurück. Die Grundthese Wittgensteins besagt, dass philosophische Probleme aus Fehlanwendungen der Sprache resultieren. Von Wilhelm Kamlah (1905-1976) und Paul Lorenzen (1915-1995) wurde diese These sowie ihr Grundterminus in der konstruktiven Wissenschaftstheorie übernommen. Die gesamte weitere Begriffsbasis einer zweckgerichteten Konstruktionslehre galt es wiederum schrittweise, zirkelfrei und alles explizit machend – d.h. nichts anderes als

5 Seine frühen Gedanken sind zusammengefasst im „Traktatus" (Wittgenstein, 1984), das spätphilosophische Werk in „Philosophische Untersuchungen" (Wittgenstein, 2001).

eben „sprachkritisch“ – neu aufzubauen. Damit entspricht das sprachkritische Entwicklungsparadigma einer zweckgerichteten Konstruktionslehre für die Informatik.

Zirkelfreies Vorgehen heißt, bei Erstellung einer Definition darauf zu achten, dass nicht Begriffe zur Erklärung verwendet werden, die selbst noch nicht definiert wurden. Das schrittweise Vorgehen steht in enger Verbindung zur Forderung, alles explizit zu machen. Es dient der Nachvollziehbarkeit und fördert damit Verständlichkeit und Transparenz.

Der Sprache allgemein und der sprachkritischen Systementwicklung im Besonderen lässt sich, angelehnt an die Drei-Schema Architektur für Datenbanken nach ANSI/SPARC, eine Spracharchitektur (Ortner & Wedekind, 2003, S. 42) zugrunde legen (Abb. 1). In der Drei-Schema-Architektur nach ANSI/SPARC entsprechen die Verbindungen zwischen den Ebenen jeweils einer Transformation. Von einer Transformation kann jedoch in der Spracharchitektur nicht gesprochen werden, da Aussagen nicht in Sprache transformiert werden können und Sprache nicht unbedingt in Wissen transformiert werden kann. Die Verbindung zwischen den Ebenen hat den Charakter einer Konstruktion. Aussagen können gleich wie Wissen aus Sprache (re)konstruiert werden bzw. durch Sprache begründet werden.

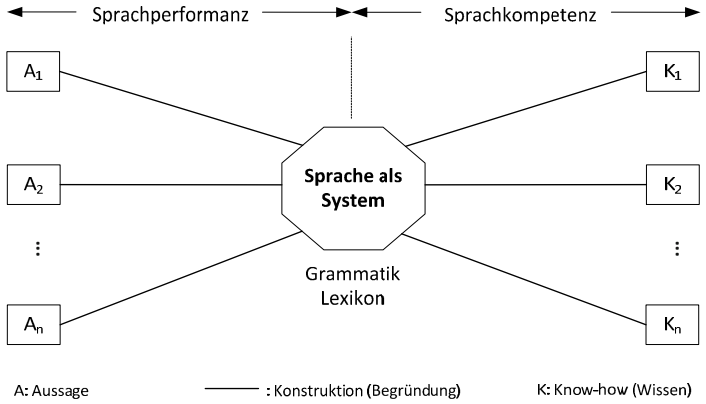

Abb. 1: Spracharchitektur

Sprachkompetenz steht für die Fähigkeit des Sprachbenutzers, den inhaltlichen, ethischen und fachlichen Aufbau der gewählten Sprache zu verstehen. In der Sprachperformanz zeigt sich die Fähigkeit des Sprachbenutzers zum Einsatz seiner Sprachkompetenz beim Handeln, beispielsweise zur rationalen, konsensorientierten Argumentation (Ortner, 2005, S. 168). Das sprachkritische Entwicklungsparadigma bietet eine Plattform, um die Sprachfähigkeit im Anwendungsgebiet zu verbessern.

Die Spracharchitektur gilt jeweils nur für eine Sprache. Die gewählte Sprache stellt dabei das integrative Element dar, welches auf Strukturen (Grammatik, Satzbau) und Begriffe (Lexika, geklärte Fachbegriffe) zurückgreift. Die systematische Handhabung von Begriffen (vgl. Abschnitte 2.2.6, S. 32ff u. 3.2.3, S. 112ff) ist elementar für den gesamten Lebenszyklus von Anwendungssystemen. Die Integration über Sprache erfolgt mittels des materialsprachlichen Ansatzes (= Sprache als System), wie er von Ortner, 1983 grundlegend und 1997 ausführlich, beschrieben wurde. Durch eine gemeinsame Erarbeitung und Einbeziehung von systematischen Sprachprodukten (z.B. Fachsprachen des Anwendungsbereichs) wird die Basis für die angestrebte, nachhaltig wirksame semantische Integration geschaffen.

Strukturell sind dazu die drei Aspekte der Sprachanalyse (= Semiotik) maßgebend. Carnap übernahm diese von Morris (1937 u. 1946) (Krauth, 1997, S. 22-23):

- **Pragmatik** (im engeren Sinne) – Die Pragmatik in der Sprachanalyse beschäftigt sich mit den psychologischen, biologischen und soziologischen Dimensionen des Bezeichnungsvorgangs (Lehmann, 1999). Es geht also in der Pragmatik um die Beziehung zwischen der Sprache und dem Menschen[6] der sie gebraucht.
- **Semantik** – Die Semantik untersucht die Beziehungen zwischen den Sprachzeichen und dem bezeichneten Objekt.[7]
- **Syntax** – Die Syntax befasst sich mit den logischen Beziehungen zwischen den Sprachzeichen untereinander, deren Verknüpfungen und Zusammenhänge.

Carnap verstand Sprachsysteme als *intensionale* und *extensionale* semantische Systeme (Krauth, 1997, S. 47ff). Seine Absicht war es damals keineswegs, den Bereich der Untersuchungen über die Sprachsysteme auf „*ontologisches*“ Gebiet auszudehnen. Aus Sicht der formalen Konstruktion von Sprachsystemen sollte an die Stelle der Frage nach dem ontologischen und erkenntnismäßigen Status von Entitäten die Frage nach der Zulässigkeit bzw. praktischen Verwendbarkeit und Möglichkeit bestimmter sprachlicher Systeme (linguistic frameworks) treten (Krauth, 1997, S. 46). Er zog sich also in dieser Hinsicht aus philosophischen Fragestellungen auf die methodologische

6 Für den vorliegenden Kontext sind dies die Nutzer von Anwendungssystemen in einem Unternehmen bzw. Anwendungsbereich.

7 Nach Carnap ist zu unterscheiden zwischen deskriptiver und reiner Semantik. „Deskriptive Semantik untersucht eine natürliche Sprache bezüglich ihrer semantischen Aspekte, reine Semantik beschäftigt sich mit der Aufstellung von Formationsregeln, Umformungsregeln, Interpretationsregeln und den sich daraus analytisch ergebenden Folgerungen. Alle diese Bestimmungen werden in einer Metasprache niedergelegt und gelten für eine Objektsprache.“ (Krauth, 1997, S. 22-23, mit Verweis auf das umfassende Werk Carnaps).

Ebene zurück. Carnap legte in diesem Zusammenhang dar, dass in konstruierten semantischen Systemen (= künstliche Systeme) u.a. Probleme bezüglich analytischer und synthetischer Sätze verhältnismäßig einfach gelöst werden können. Bei der Verwendung natürlicher Sprachen tauchen aber metaphysische Fragen und Wesensprobleme auf. Deshalb meint er, dass „here the distinction must be based on an empirical investigation of speaking habits…", dass also die Lösung auf pragmatischem Gebiet zu suchen ist (Krauth, 1997, S. 57). Die hier angesprochene methodische Unterstützung auf pragmatischem Gebiet wird mit dem sprachkritischen Entwicklungsparadigma aufgegriffen und in der sprachbasierten Informatik verankert.

Bei der Sprachkritik geht es um den Sprachgebrauch (anderer). Dieser ist Gegenstand der Betrachtung und der Kritik, er wird bewertet und beurteilt. Dies passiert in sogenannten Sprachspielen bzw. Sprechakten (vgl. Abschnitt 5.1, S. 187) auf syntaktischer, semantischer und pragmatischer Ebene. Sprachkritik hat grundsätzlich mit zwei Aktivitäten zu tun, mit der Analyse sprachlicher Äußerungen, Aussagen, Diskurse, Texte und auch der Analyse von Sprachen selbst sowie mit der Beurteilung bzw. Bewertung der analysierten Gegenstände (Wimmer, 2003, S. 417). Sprachkritik ist nicht nur eine Sache von Experten, sondern ist für alle da. Sprachkritik bzw. sprachkritische Aktivitäten entfalten sich für jede Wissenschaft, die sich damit beschäftigt, anders. Jede Disziplin bringt eigene Maßstäbe und Normen legitimierend ins Spiel. Einen einheitswissenschaftlichen Entwurf gibt es demnach nicht.

Will man das sprachkritische Entwicklungsparadigma kurz zusammengefasst wissen, so lassen sich drei Prinzipien anführen (Ortner, 1993 u. 2008a):

1) Ordinary Language is alright (Ludwig Wittgenstein).
2) Languages have already made a great many, sometimes too many, distinctions (Paul Lorenzen).
3) We will be held responsible for what we are saying (Peter Janich).

2.1.3 Ganzheitlicher Ansatz

Ein ganzheitlicher Ansatz befasst sich nicht nur mit Systemen, sondern bezieht immer deren Kontext mit ein. Etwas elementarer ausgedrückt heißt das, in einem ganzheitlichen Ansatz wird ein Gegenstand immer in Verbindung mit seinem Gebrauch gesehen.

Beispiel:

Gegenstand: Computermaus
Kontext: Architekturbüro
Charakter des Gegenstands:Computermaus ist ein Zeichengerät

Gegenstand: Computermaus
Kontext: Internetcafé
Charakter des Gegenstands:Computermaus ist ein Navigationsgerät

Das Beispiel zeigt, dass der gleiche Gegenstand in einem anderen Kontext zu einem anderen Gebrauch eingesetzt werden kann und daher einen anderen Zweck erfüllt. Ein System ist eine Zusammenstellung von Gegenständen, die in einer Beziehung zueinander stehen. Wird ein System in einen Kontext gestellt, erhält es einen bestimmten Charakter bzw. eine bestimmte Ausprägung. Unterschiedliche Kontexte können für ein gleiches System unterschiedliche Anwendungscharakteristika ergeben bzw. Ausprägungen bedingen. Dies gilt auch für Gegenstände, wie das o.a. Beispiel zeigt. Wird beispielsweise eine Maus in einem Architekturbüro zum Zeichnen verwendet, hat sie für den Anwender den Charakter eines Zeichengerätes, d.h. der Nutzer kann mit Hilfe der Maus u.a. Punkte und Linien erzeugen. Dieser Nutzer nimmt die Maus als Zeichengerät wahr. Wird die Maus in einem Internetcafé verwendet, hat sie für den Anwender den Charakter eines Navigationsgerätes, d.h. der Nutzer kann mit Hilfe der Maus u.a. verschiedene Schaltflächen anklicken um Teile der Cyberwelt zu erkunden. Dieser Nutzer nimmt die Maus als Navigationsgerät wahr. Die Verbindung von Gegenstand und Gebrauch bzw. System und Kontext ist also maßgebend für die Ausprägung von Gegenständen bzw. Systemen und deren Wahrnehmung durch einen Nutzer. Diese Zusammenhänge eines ganzheitlichen Ansatzes verdeutlicht Abb. 2 allgemein für System und Kontext sowie Gegenstand und Gebrauch und konkret für das erwähnte Beispiel.

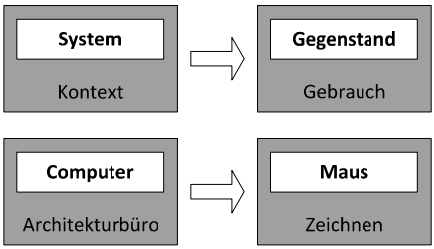

Abb. 2: Zusammenhänge eines ganzheitlichen Ansatzes

Über die Ebene der Pragmatik ist der ganzheitliche Ansatz eng mit dem sprachkritischen Entwicklungsparadigma der sprachbasierten Informatik verknüpft (vgl. Ab-

schnitt 2.1.2, S. 17ff). Für die Umsetzung des ganzheitlichen Ansatzes in der Anwendungsentwicklung heißt dies, dass jede Auseinandersetzung mit Gegenständen einen Bezug zu deren konkreten Gebrauch hat, im Sinne eines gleichzeitig konstruktivistischen Ansatzes heißt es „Aus dem Gebrauch, für den Gebrauch."

2.2 Schemaentwicklung

2.2.1 Einleitung

Der Begriff „Schemaentwicklung" entspringt der transdisziplinären Grundlagenforschung zur sprachbasierten Informatik (Ortner, 2005) und lässt sich trennen in die Begriffe „*Schema*" und „Entwicklung". Schema ist der zentrale Begriff einer Konstruktionslehre, wie sie für Anwendungssysteme in erster Linie von Wedekind und Ortner aus Artefakten und Theorien relevanter Disziplinen und Wissensgebiete (z.B. Prädikatenlogik, „*rationale Grammatik*" als Strukturlehre einer Sprache) rekonstruiert wurde und permanent weiterentwickelt und verfeinert wird. Die Konstruktionslehre für Anwendungssysteme kann in die Teilbereiche „Methodologie", „Anwendungselemente" (im Sinne von Komponenten für den Aufbau und Betrieb von Anwendungssystemen) und „Qualitätsmanagement" (als Begründungslehre) eingeteilt werden (Ortner, 2005, S. 14). Die Schemaentwicklung ist dem Bereich der „*Methodologie*" zuzuordnen.

Die von Ortner skizzierte Schemaentwicklung (Abb. 3; Ortner, 2005, S. 34) stellt einen Rahmen dar, der mit der Beschreibung der Komponenten einer Anwendungsentwicklungsumgebung (Ortner, 2005, S. 203) konkretisiert wird. Handlungsorientiert werden diese Komponenten auch zum Begriff Anwendungsmanagement zusammengefasst. Regelwerke und Standards für die Unterstützung dieser Managementaufgabe finden in Instrumenten wie z.B. einer SOA-*Governance* ihren Ausdruck (z.B. Johannsen & Goeken, 2006; Erl, 2008).

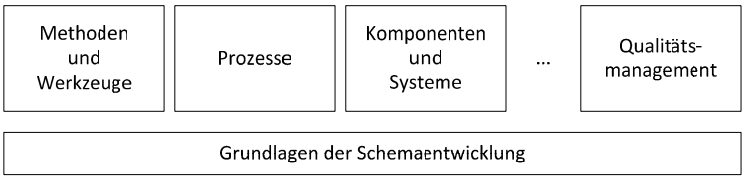

Abb. 3: Rahmen der Schemaentwicklung

Grundsätzlich ist diese Schemaentwicklung allgemein zu sehen, d.h. dass auch Altsysteme, die abgelöst werden, oder Systemteile für Neusysteme als Komponenten ge-

lten können. Abseits der IT kann z.B. das HGB als System (genauer als Wissenssystem) betrachtet und schematisiert werden.

2.2.2 Generischer Sprachansatz für die Konstruktionslehre

Die Sprache bildet die Grundlage der Schemaentwicklung, wobei in der Sprache, zurückgehend auf die indoeuropäischen Sprachen, zwischen

„materiellen Inhalten“ eines Satzes (= Gegenstand der Lexikografie)

und den

„modifizierenden Elementen“ (= Gegenstand der Grammatik)

unterschieden wird (Holzer, 2005, S. 218, mit Verweis auf Boas, 1911a, S. 25f). Dass eine Übertragbarkeit dieser Struktur auf amerikanisch-indianische Sprachen nicht unbedingt gegeben ist, wird für den vorliegenden Kontext als nicht relevant eingestuft. Es kann aber für eine künftige Transformation dieses Verständnisses in den angloamerikanischen Sprachraum durchaus bedeutend sein.

Die umfangreichen ethnischen Sprachforschungen Boas führten zur Erkenntnis, dass zusätzlich zu den beiden grundlegenden Bausteinen einer Sprache (Lexikon und Grammatik) eine umfassende Klassifikation aller Erfahrungen der Sprache zugrunde liegen muss (Boas, 1911a, S. 15f; Holzer, 2005 S. 217). Eine Klassifikation auf grundlegender Ebene berücksichtigt Ortner bei seinem Sprachansatz in Form einer Gegenstandseinteilung. Diese dient u.a. auch zur systematischen Behandlung von Begriffen (vgl. Abschnitt 2.2.6, S. 32ff). In der Logik, den Sprachwissenschaften und der Wissenschaftstheorie fundiert, entwickelt Ortner (1997, S 82-90) auf dieser Basis einen generischen Sprachansatz, der sowohl für die gesamte Konstruktionslehre der sprachbasierten Informatik als auch für ein umfassendes *Enterprise Engineering* Gültigkeit hat (Abb. 4; Ortner, 1997, S. 83). In Erweiterung zu einer reinen Konstruktionslehre eines Wissensgebietes bedeutet dies, dass über diesen Ansatz ein Brückenschlag zu anderen Disziplinen ermöglicht wird. Mit dem Hinweis auf das Enterprise Engineering wird dies exemplarisch für die Einbeziehung der Betriebswirtschaft angedeutet, d.h. der strategische und der organisationale Bereich sowie andere soziotechnische Zusammenhänge in einem Unternehmen und zwischen Unternehmen können systematisch integriert werden. Alle relevanten *Sprachartefakte* lassen sich auf diesen Sprachansatz zurückführen. Der generische Sprachansatz bietet eine universelle Grundlage für die Integration weiterer Disziplinen wie z.B. das Usability Engineering oder das Acceptance Engineering.

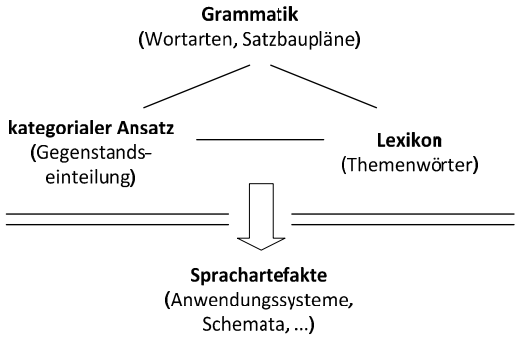

Abb. 4: Generischer Sprachansatz für die Konstruktionslehre

Eine Sprache besteht demnach aus folgenden Bausteinen (Ortner, 2005, S. 28):

- Gegenstandseinteilung z.B. als Grundlage zur Typisierung von Wörtern zur Klärung von *Begriffen* (Ortner, 1997).
- Grammatik z.B. in Form von Satzbauplänen definiert.
- Wortschatz z.B. in Form eines Lexikons organisiert.

Verschiedene Modi von Zusammensetzungen dieser Bausteine, genauer gesagt der Ausprägungen dieser Bausteine, ergeben *Sprachartefakte*. Sie sind als Ergebnisse von Sprachhandlungen zu verstehen, die den Charakter von *Sprachspielen* bzw. Sprechakten haben (vgl. Abschnitt 5.1, S. 187).

2.2.3 Schema, Ausprägung und Sprachhandlungen

Jeder Sprache liegt explizit oder implizit eine Architektur zugrunde. Eine Spracharchitektur (Abb. 1, S. 18) zeigt uns wesentliche, grundsätzliche Zusammenhänge der Anwendung einer Sprache. Eine Sprache mit all ihren Bausteinen bildet das konzeptionelle Schema auf sprachlicher Ebene für jegliche Verbindungen zwischen Gegenständen (Begriffen) und Strukturen (Grammatik). Was für die Sprache selbst gilt, gilt auch für andere Systeme. Die Sprache mit all ihren Bausteinen bildet das konzeptionelle Schema auf sprachlicher Ebene für jegliche Verbindungen zwischen vorhandenen Gegenständen und Strukturen in einem Anwendungsbereich (Wissen, Schemata) und möglichen Sprachartefakten dazu (Aussagen, Ausprägungen), die schließlich durch Sprechakte (Sprachhandlungstypen) zum Ausdruck kommen. Wird der horizontale Zusammenhang von Aussage (Ausprägung), Sprache als System und Wissen (Schema) aus der Spracharchitektur auf unterschiedliche Sprachebenen transferiert, sprechen wir von einer Sprachebenenarchitektur (Heinemann, 2006, S. 112; Heinemann & Ortner, 2004, S. 444). Der horizontale Zusammenhang selbst gilt auf jeder Sprachebene.

Es besteht also jede Sprachebene aus dem Konzeptpaar Schema und Ausprägung, die wiederum über einen Sprachhandlungstyp miteinander verbunden sind. Das charakteristische eines Schemas ist seine universelle Geltung im jeweiligen Bezugsrahmen. Ausprägungen repräsentieren im Gegensatz dazu einen speziellen d.h. einen singulären Aspekt des Schemas (Heinemann, 2006, S. 83). Das Schema ist uns einerseits „Vorlage“ für Äußerungen (Sprachhandlungen), gleichzeitig ist es auch Grundlage dafür, dass Aussagen, hier verstanden als singuläre Aspekte von etwas im Sinne von Information, verstanden werden können (Schema = Wissen; Ausprägung = Information).

Sprechakte sind für die Anwendungssystementwicklung Auskunftsarten im Sinne der Sprechakttheorie von Austin und Searle, d.h. es liegt eine Intention des Sprechers dahinter. Mittels dieser Auskunftsarten werden in der Erhebung sowohl statische als auch dynamische Strukturen (z.B. Aufbau- und Ablauforganisation) rekonstruiert (Jablonski, Böhm & Schultze, 1999, S. 12-16). Im weiteren Verlauf der Entwicklungsarbeit werden diese genauer beschrieben und schließlich im Zuge der Einführung validiert. Sprechakte können also als Beziehungen zwischen Ausprägung und Schema verstanden werden. Demnach sind Sprechakte modellierbar.

> Beispiel:
>
> VERSTEHEN ist eine Beziehung zwischen Ausprägung und Schema
> (in dieser Richtung).
> Deshalb ist VERSTEHEN modellierbar und
> Schemaerwerb fördert deshalb das VERSTEHEN.

Der allgemeinere und auch weiter reichende Begriff zu „Sprechakte“ heißt „Sprachhandlung“. In Anlehnung an Wittgenstein können Sprachhandlungen auch als Sprachspiele bezeichnet werden. Nach dem Konzept der Sprechakte, wie es von Austin und Searle definiert wurde, liegt hinter einem Sprechakt immer die Intention eines Sprechers. Sprachhandlungen hingegen umfassen alle Sprechakte[8] und umfassen zusätzlich alle Handlungen, die als Ergebnis ein *Sprachartefakt* haben, d.h. die Beziehung zwischen Sprache und *Sprachartefakt* ist eine Sprachhandlung (Ortner, 1997, S. 82 mit Verweis auf den Sprachwissenschaftler Bühler, 1978, S. 52):

> Mir dünkt, es sei so etwas wie ein Ariadnefaden, der aus allerhand nur halb begriffenen Verwicklungen herausführt, gefunden, wenn man das Sprechen entschlossen als Handlung (und das ist die volle Praxis im Sinne des Aristoteles) bestimmt.

8 Es ist damit jedes Interagieren von Mensch zu Mensch, von Mensch zu Maschine und umgekehrt und auch von Maschine zu Maschine gemeint.

Sprachhandlungen dienen auch dazu, den vertikalen Zusammenhang zwischen Sprachebenen zu verdeutlichen. Damit wird die Praxisrelevanz der konstruktiven Wissenschaftstheorie (Kamlah & Lorenzen, 1967) deutlich. Der Aufbau einer Sprache in der Praxis kann einerseits unmittelbar aus praktischem Handeln und mit Nennung der relevanten Wörter zu den Handlungen und ihren Gegenständen erfolgen. Dieser Vorgang wird von Kamlah & Lorenzen als Zu- oder Absprechen von Themenwörtern (Prädikatoren) bezeichnet. Andererseits kann Sprache durch Definition neuer Wörter mittels bereits bekannter Wörter erfolgen (Ortner, 1997, S. 82). Beides sind Sprachhandlungen im beschriebenen Sinn. In beiden Fällen handelt es sich um eine Sprachhandlung zur Schemabildung als schema(re)konstruierendes Sprachhandeln mit dem Verständnis von Sprache als System.

Demgegenüber steht die Sprachhandlung, deren Ergebnis (*Sprachartefakt*) eine Ausprägung darstellt. Sie wird von Ortner (2005, S. 276) auch als instanziierendes Sprachhandeln bezeichnet. Die Ausprägungen stellen eine gängige Rede- und Handlungspraxis dar, d.h. auf dieser Ebene finden Aktivitäten und Abläufe in der realen Welt statt, wie z.B. die Kommunikation.

2.2.4 Systematische Verankerung von Sprache im Aktivitätsmodell

Als Grundlage für die Darstellung der systematischen Verankerung von Sprache in Prozessmodellen wird das allgemeine Metamodell für Aktivitäten von Noack & Schienmann (1999, S. 169) herangezogen. Abb. 5 (S. 27) zeigt eine bereits erweiterte Version dieses Modells.

Eine Phase stellt eine sinnvolle Gruppierung von Prozessen dar. Sie ist die höchste dargestellte Aggregationsstufe von Aktivitäten. Durch Gruppierung von Phasen in der Anwendungssystementwicklung können Vorgehensmodelle entstehen, wie z.B. das Multipfad-Vorgehensmodell. Ein Prozess stellt wiederum eine sinnvolle Gruppierung von *Aktivitäten* zu einer planbaren und überschaubaren Einheit dar. Die Gruppierung erfolgt in der Regel nach der zeitlichen Abfolge von Aktivitäten. Sowohl für Phasen als auch für Prozesse gilt, dass sie Resultate erstellen bzw. manipulieren oder diese auch als Input benötigen können. Letztgenannte Beziehungen zwischen den Entitäten sind in Abb. 5 (S. 27) aus Gründen der Übersichtlichkeit nicht eingezeichnet.

Eine *Aktivität* ist die Beschreibung eines oder mehrerer Arbeitsschritte. Es wird beschrieben, was gemacht wird und wie etwas gemacht wird. Da unterschiedlichste Formen von Aktivitäten denkbar sind, gibt es so etwas wie Aktivitätstypen. Aktivitäten haben in der Regel bestimmte Eigenschaften (Attribute) und benötigen einen bestimmten Input, den sie als Arbeitsgrundlage verwenden. Jede *Aktivität* erzeugt ein Resultat, dies kann entweder die Veränderung eines bereits vorhandenen Resultats in

Form einer inhaltlichen Änderung oder Zustandsänderung oder die Erstellung eines neuen Ergebnisses sein. Aktivitäten können in Unteraktivitäten gegliedert werden. In diesem Fall werden erst die jeweiligen Unteraktivitäten detailliert beschrieben. Resultate einer *Aktivität* können in unterschiedlicher Form vorliegen. Als solche Resultatstypen unterscheiden wir beispielsweise die Dokumentation, den Programmcode oder auch ein fertiges System. Resultate können sich aus mehreren Resultaten zusammensetzen und durch Aktivitäten immer wieder verändert werden. Resultate können bestimmte Eigenschaften haben wie z.B. einen Bearbeitungsstatus. Die Relation zwischen Resultat und Aktivität ist im Aktivitätsmetamodell explizit enthalten. Es bestehen aber auch direkte Beziehungen zwischen Resultat und Prozess sowie zwischen Resultat und Phase. Das bedeutet beispielsweise, dass die Resultate einer Phase nicht unbedingt die Aggregation der Resultate der beteiligten Prozesse und Aktivitäten sein müssen.

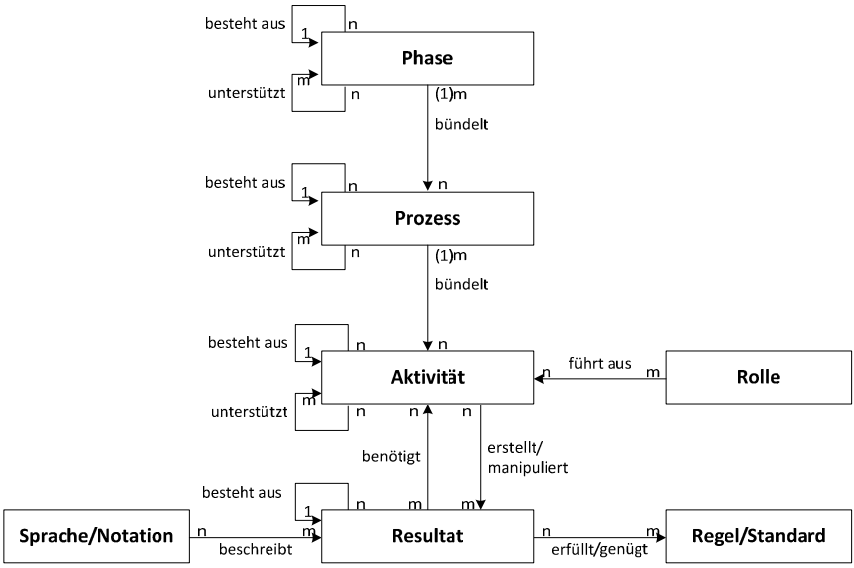

Abb. 5: Aktivitäts-Metamodell

Eine Rolle definiert Fähigkeiten, Kenntnisse und Fertigkeiten von Personen oder Systemen, die zur Ausführung einer Aktivität notwendig sind. Damit wird das Anforderungsprofil von Personen beschrieben und somit die Zuordnung von möglichen ausführenden Personen erleichtert. Über Rollen können Zuständigkeiten und Verantwortlichkeiten, die in Verbindung mit der Aktivität stehen, geklärt werden. Die prozesszentrische Vorgehensweise (vgl. Abschnitt 2.3.2, S. 37ff) greift dieses Faktum

auf und nutzt den Zusammenhang um Rollen in Verbindung mit Aktivitäten zu Stellenbeschreibungen zu akkumulieren. Rollen müssen nicht von Menschen wahrgenommen werden, sondern können auch von Systemen, z.B. Informationssystemen oder Services, wahrgenommen werden.

Zur Ausführung von Aktivitäten werden Techniken, Werkzeuge und Methoden herangezogen. Technologiebezogen handelt es sich dabei um Verfahren bzw. Methoden, die bei Bedarf auch implementiert sein können. Bezogen auf den ausführenden Menschen handelt es sich um Fähigkeiten. Notwendige Fähigkeiten sind in Verbindung mit Rollen aktivitätsbezogen zu definieren. Verfahren und Methoden, unabhängig davon, ob sie implementiert oder manuell auszuführen sind, stellen wiederum Aktivitäten dar. Dieser Zusammenhang wird in der Grafik mit dem rekursiven Bezug „Aktivität unterstützt Aktivität“ deutlich.

Sprachen sind Mittel zur Darstellung von Resultaten. Von der natürlichen Sprache über rationale Sprachen (Wedekind et al., 2004b, S. 265f) und Kunstsprachen, wie sie z.B. mit UML, BPMN aus dem Sprachkanon der OMG (2008a) für die Modellierung zur Verfügung stehen, bis hin zu Programmiersprachen sind hier viele Möglichkeiten gegeben. Im Zuge der Modellierung einer Aktivität ist festzulegen, welche Sprachen zur Darstellung der Resultate zulässig sind.

Regeln und Standards dienen dazu, die Qualität der Resultate zu gewährleisten. Dies kann innerhalb eines Projekts oder auch über mehrere Entwicklungsprojekte hinweg der Fall sein. Beispiele für solche Regeln und Standards sind Programmierrichtlinien (Coding Styleguides), Design Guidelines oder Inhaltsstandards. Eine im Entwicklungsprozess gemeinsam erarbeitete Fachsprache gilt als Inhaltsstandard und dient zur Stabilisierung von Namenskonventionen und anderen Definitionserfolgen im Entwicklungsprozess. Auch branchenspezifische Warensystematiken und Nomenklaturen bilden Inhaltsstandards. Regeln und Standards werden oft bereichsspezifisch festgelegt, z.B. als SOA-Governance, administriert und verfügbar gemacht und zunehmend in unternehmensweite Regelwerke, wie z.B. eine Corporate-*Governance*, integriert. Allgemein anerkannte Regeln und Standards verbunden mit Aktivitäten helfen, viele Diskussionen zu Beginn eines Entwicklungsprojekts zu vermeiden. Die Resultate einer Aktivität sollen die vorgegebenen Regeln und Standards erfüllen. Regeln und Standards können über mehrere Aktivitäten Gültigkeit haben. Die eingeführten Begriffe sind in Abb. 5 (S. 27) zu einem Aktivitäts-Metamodell zusammengefasst.

Im Gegensatz zur Auffassung von Noack & Schienmann wird das Modell (Abb. 5, S. 27) bereits auf der Basisstufe als Metamodell aufgefasst. In Erweiterung zum ursprünglichen Modell wurden die Aggregationsstufen von Aktivitäten hier ausmodelliert. Es repräsentiert als Datenmodell die statische Sicht auf den Zusammenhang zwischen Sprache und Aktivitäten. Eine dynamische Sicht kann sinnvoll ab der Aggre-

gationsstufe „Prozess“ erstellt werden. Eine generische dynamische Sicht kann als sprachlogisches Prozessmodell aufgefasst werden. Das Aktivitäts-Metamodell kann demnach auch als Schnittstelle zwischen dem sprachlogischen Prozessmodell und einem Prozessrelationenmodell dienen. Überlegungen zu einem Prozessrelationenmodell mit Aktivitätsrelationen, Resultatrelationen in Analogie zum Konzept relationaler Datenbanken, werden an dieser Stelle nicht weiter verfolgt, da dies den Rahmen der Arbeit sprengen würde. Eine Betrachtung in einer gesonderten Forschungsarbeit könnte sich jedoch als lohnend erweisen.

2.2.5 Sprachlogisches Prozessmodell

Ausgehend vom Konzeptpaar „Schema und Ausprägung“ lässt sich ein sprachlogisches Kommunikationsmodell (Ortner, 2005, S. 156-158; Heinemann, 2006, S. 90-94) als Basis zur Modellierung von Kommunikationsgeschehnissen ableiten. Verständnis schaffende Kommunikation ist ein Vorgang, der auf Ausprägungsebene passiert. In diesem Vorgang wird Information, verstanden als Ausprägung von Wissen, von einem Kommunikationspartner zu einem anderen Kommunikationspartner gesendet. Das erwartete *Verstehen* der Information wird durch die Kenntnis gemeinsam genutzter Schemata möglich. Auf Seiten des Senders ist das Schema Mittel des Äußerns (als Sprachhandlung) und auf Seiten des Empfängers ist das Schema Mittel des Verstehens (als Sprachhandlung). Aus Sicht der Psychologie handelt es sich bei diesen Schemata um mentale Modelle (Abb. 21, S. 81).

Das sprachlogische Kommunikationsmodell stellt eine Fundierung für Sprachhandlungen auf Ausprägungsebene dar. Kommunikation ist ein gerichteter Ablauf eines Geschehens in Form der Sprachhandlungen „senden“ und „empfangen“. Wenn eine *Aktivität* als gerichteter Ablauf eines Geschehens definiert ist und ein Prozess als Aggregation von mindestens zwei Aktivitäten, dann ist eine Kommunikation eine Ausprägung auf Metaebene im Sinne eines Prozesstyps. Weitere Prozesstypen sind z.B. *Workflows* und Methoden.

Nach Lehmann (1999, S. 254) können Ausprägungen eines Aktivitäts-Meta-Schemas in Geschehnistypen (z.B. Zustand, Prozess) eingeteilt werden. Diesen Typen ordnet er wiederum Verben (Zustandsverben, Aktivitätsverben, usw.) zu, was auf Ausprägungsebene zu konkreten Sprechakten führt. Dem Aktivitäts-Metaschema übergeordnet ist die Ebene des Geschehnis-Schemas, abgeleitet aus der Gegenstandseinteilung nach Ortner (1997) und Lehmann (1999, S. 251). Für jeden Geschehnistyp kann ein Aktivitäts-Metaschema modelliert werden. Für den vorliegenden Kontext ist das Metaschema für den Geschehnistyp „Prozess“ relevant, wir bezeichnen es als Prozess-Metaschema. Die beschriebenen Zusammenhänge sind in Abb. 6 (S. 30) in Analogie

zur Sprachebenenarchitektur von Heinemann & Ortner (2004, S. 444) bzw. Heinemann (2006, S. 112) konkretisiert.

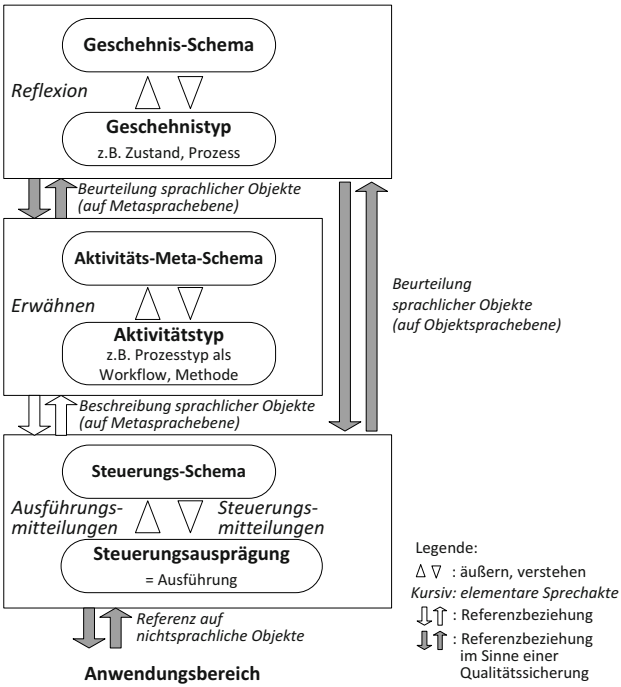

Abb. 6: Sprachebenenarchitektur für Geschehnisse

Der Bezug zu nichtsprachlichen Objekten wird auf der untersten Ebene[9] hergestellt. Nichtsprachliche Objekte sind Gegenstände der realen Welt, beispielweise Waren in einem Großhandelsunternehmen als ein möglicher Anwendungsbereich. Die Steuerungsausprägungen sind eine konkrete sprachliche Beschreibung des Anwendungsbereichs im dort üblichen Sprachgebrauch. Die konkreten Objekte und Zusammenhänge der realen Welt werden sprachlich dargestellt. Die allgemeine Darstellung einer solchen Ausprägung ist ein Steuerungs-Schema. Auf der nächst höheren Ebene, der Metaebene, werden die sprachlichen Objekte der jeweils tiefer liegenden Ebene beschrieben. Dies erfolgt in den möglichen Ausprägungen und wiederum auch als Schema (Metaschema). Auf der Ebene über der Metaebene, der Meta-Meta-Ebene, gilt es schließlich die darunter liegenden Ebenen zu reflektieren und zu beurteilen.

9 Nach OMG (2008a) wird diese unterste Ebene als M1, die Metaebene als M2 und die Meta-Meta-Ebene als M3 bezeichnet.

Aus der Sprachebenenarchitektur für Geschehnisse lässt sich der Zusammenhang zu einem sprachlogischen Prozessmodell ableiten. Das sprachlogische Prozessmodell (Abb. 7) ermöglicht die systematische Beschreibung von Prozessen als Steuerungsschemata mit ihren jeweiligen Ausprägungen, z.B. Beschreibung von Workflow-Schema und Workflow-Instanzen. Ob die Ausführenden durch Mensch oder Technik repräsentiert werden, ist auf dieser Ebene unerheblich, beides ist möglich.

Dieses sprachlogische Prozessmodell steht uns nun als Basis zur Rekonstruktion von konkreten (Arbeits-)Prozessen zur Verfügung. Im Augenblick der Durchführung eines (Arbeits-)Prozesses ist eine temporäre Normierung im Sinne von Schema und Ausprägung erforderlich. Das heißt, die beteiligten Ausführenden halten sich für eine gewisse Zeit an die für die jeweilige *Aktivität* bzw. den Prozess vorgesehenen Konventionen in Form von Regeln, Standards und Bedingungen, um zu einer Übereinstimmung auf Resultatebene im Sinne eines Resultats des Aktivitäts-Metamodells von Abb. 5 (S. 27) zu gelangen. Resultat(e) der Durchführungen von Ausführendem „1“ können vom Ausführenden „2“ und jene von Ausführendem „2“ vom Ausführenden „n“ als Input benötigt werden.

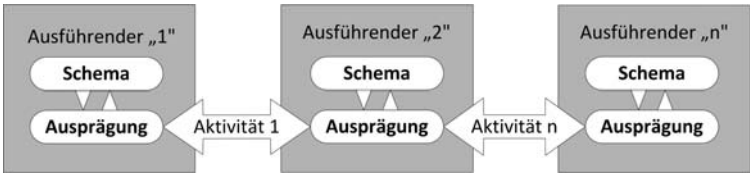

Abb. 7: Sprachlogisches Prozessmodell

Einflüsse von kommunikativen und anderen Bedingungen auf einen Prozess werden im sprachlogischen Prozessmodell nicht unmittelbar transparent. Wenngleich die Auseinandersetzung mit Einflüssen dieser Art überwiegend den Sprachwissenschaften und den Verhaltenswissenschaften zuzurechnen ist, ist diese auch für die Beeinflussung von Akzeptanz und Usability von Relevanz. Hierzu können wir auf die statische Sicht auf Prozesse zurückgreifen. Grundsätzliche Schwierigkeiten des Verständnisses dieser Zusammenhänge können aus der Doppel- oder Mehrfachbedeutung des Begriffs „Prozess“ (= *Homonym*) in der Praxis resultieren. Für den vorliegenden Kontext ist der Begriff „Prozess“ in Verbindung mit dem Aktivitätsmetamodell definiert (vgl. Abschnitt 2.2.4, S. 26ff bzw. Abb. 5, S. 27).

Die gezeigte Schemaentwicklung ist grundlegend für die sprachbasierte Informatik und umfasst damit das Prozess- und das Methodenengineering ebenso wie die Anwendungsentwicklung als Anwendungsbereich für Vorgehensmodelle und Entwicklungsmethoden. Sowohl Methoden als auch Vorgehensmodelle stellen Schemata

bzw. Metaschemata dar, die im Falle des Einsatzes in einem konkreten Projekt instanziiert werden müssen. Jedes Vorgehen lässt sich systematisch auf diese Schemaentwicklung zurückführen. Innerhalb dieser systematischen Schemaentwicklung erfolgt eine besondere Handhabung von Begriffen. Der Rahmen dazu wird nachfolgend gezeigt, eine anwendungsorientierte Konkretisierung erfolgt in Abschnitt 3.2.3 (S. 114ff).

2.2.6 Systematische Handhabung von Begriffen

Menschen äußern sich u.a. in natürlicher Sprache, konkret in Umgangssprache, Hochsprache oder Fachsprache, aber mitunter auch in Wissenschaftssprache. Sowohl sprachliche als auch nichtsprachliche Äußerungen von Menschen lassen sich in Form von Aussagen dokumentieren. Aussagen sind Sätze, die in sogenannten Aussagensammlungen zusammengefasst und bei Bedarf zweckorientiert gruppiert werden. Die Grundbausteine von Aussagen sind Begriffe. Ein Begriff steht auf abstrakter Ebene und bezieht sich auf einen Gegenstand in der Welt. Ein Gegenstand kann entweder ein Ding oder ein Geschehnis sein, welches durch Eigenschaften näher charakterisiert werden kann (Lehmann, 1999, S. 138). Unabhängig davon, ob der Gegenstand abstrakt oder konkret ist, hat er eine Benennung. Ein Begriff wird durch mindestens eine Benennung repräsentiert. Eine Benennung ist beispielsweise ein Wort (Lorenzen, 1985). Ortner (1994b u. 2005) spricht in diesem Zusammenhang von einem Begriffswort bzw. einem Prädikator. Nachdem die Repräsentation von Begriffen klar ist, gilt es nun, eine Systematik für die Erschließung der *Bedeutung* eines Begriffs aufzuzeigen.

Die *Bedeutung* eines Begriffs (= Semantik) wird durch seine *Intension* (= Inhalt eines Begriffs) definiert und durch seine *Extension* (= Umfang eines Begriffs) weiter konkretisiert. Die *Intension* ist die Gesamtheit der Eigenschaften bzw. Merkmale und deren Beziehung zueinander, die einen *Begriff* bzw. einen Gegenstand ausmachen (Baum, 1978). Zwei Begriffe sind intensional identisch, wenn sie inhaltlich das Gleiche bedeuten, z.B. Geige und Violine. Die Intension eines Prädikators ist also die Eigenschaft (oder Relation), die er ausdrückt. Seine *Extension* ist die Klasse derjenigen Objekte (oder geordneten n-tupel), auf die er angewandt wird. Die *Extension* eines Begriffs ist demnach das, was als „in ihm enthalten" gedacht wird, also die Gesamtheit der Arten des Begriffs bzw. die Klasse der unter einen Begriff fallenden Gegenstände (Baum, 1978; Lehmann, 1999). Die *Extension* wird auch als der Umfang des Begriffs bezeichnet. Diese Systematik lässt sich in einem Begriffsmodell zusammenfassen (Abb. 8).

Die Gesamtheit der *Intension* eines Begriffs ergibt das Schema eines Begriffs. Dieses umfasst seine Merkmale (Intension nach innen) und seine Beziehungen zu anderen Begriffen (Intension nach außen). Die Gesamtheit der *Extension* eines Begriffs um-

fasst alle möglichen Ausprägungen des beschriebenen Schemas und die Menge der Beschreibungen der Gegenstände, die unter den Begriff fallen. Für die Anwendungsentwicklung sind die Ausprägungen jeweils auf den Anwendungsbereich beschränkt. Das Begriffsmodell geht zurück auf Frege und wird u.a. von Ortner (1994b, S. 576 u. 2005, S. 239-240), Schienmann (1997, S. 143) und Lehmann (1999, S. 136-146) näher beschrieben.

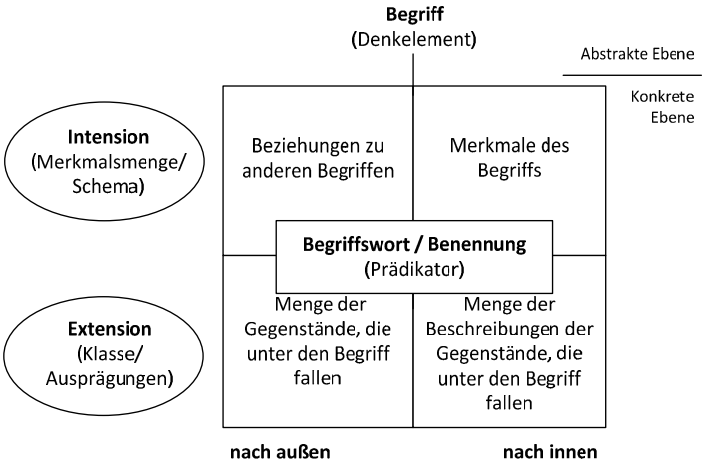

Abb. 8: Begriffsmodell

Ist ein *Begriff* nicht eindeutig definiert, bedarf es einer systematischen Klärung. Das Begriffsmodell dient zur Unterstützung dieser Klärung. Das sprachkritische Entwicklungsparadigma bedingt zirkelfreies Vorgehen, d.h. es ist darauf zu achten, dass Erklärungen zu Begriffen ausschließlich mit bereits definierten Begriffen erfolgen. Um der Forderung „der schrittweisen Erarbeitung“ und derjenigen „alles explizit zu machen“ (Heinemann, 2006, S. 40), nachzukommen, bedarf es einer systematischen Auseinandersetzung mit Begriffen, die in der natürlichsprachlichen Rekonstruktionsarbeit in Form von Wörtern als Benennungen von Gegenständen auftauchen. Für dieses konstruktivistische Vorgehen steht die Sprache und mit ihr die verwendeten Wörter (Themenwörter, Prädikatoren) im Mittelpunkt. Ortner (1997) gibt einen Überblick über Gegenstandseinteilung (Abb. 9, S. 34) und Wortarten als Grundlage zur Typisierung von Wörtern und damit zur systematischen Unterstützung der *Begriffsklärung*. In der dargestellten Gegenstandseinteilung wird betont, dass sie für alle sprachlichen Repräsentationen von Anwendungsbereichen Gültigkeit hat, also z.B. auch für diagrammsprachliche Darstellungen.

Die Repräsentation von Wissen erfolgt durch Aussagen, die im Rekonstruktionsprozess in Schemata überführt werden (Ortner, 2005). Aus der Erhebungsarbeit in diesem Prozess ergibt sich eine Aussagensammlung. Aussagen sind Sätze, die Beziehungen und Gegenstände im Anwendungsbereich beschreiben. Die Aussagen selbst bestehen aus Begriffen und Beziehungen zwischen Begriffen. Sind die Aussagen allgemeiner Natur, handelt es sich um Schemata. Es ist unschwer zu erkennen, dass eine Schematisierung von Vorgängen im Anwendungsbereich einen sehr konsequenten und stringenten Umgang mit Begriffen erfordert. Sind Begriffe im jeweiligen Kontext nicht klar zugeordnet und definiert, ist eine *Begriffsklärung* erforderlich. Dazu bedarf es einerseits der Abgrenzung des Begriffs hinsichtlich Inhalt und Umfang auf Basis des Begriffsmodells, und andererseits der Erkennung und Beseitigung von Begriffsdefekten. Mögliche *Begriffsdefekte* und deren Behebung werden in Abschnitt 3.2.3 (S. 114ff) besprochen.

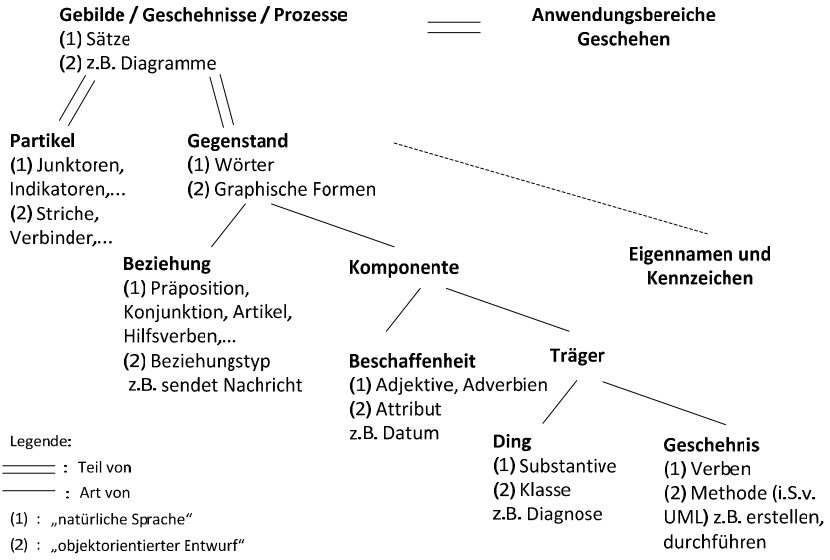

Abb. 9: Gegenstandseinteilung und Wortarten

2.3 Architektur, Vorgehensmodell und Methoden der sprachbasierten Informatik

2.3.1 Ebenen der Informationsverarbeitung

Informationstechnologie im engeren Sinn umfasst die Ebenen Hardware, Kommunikationssysteme und Basissoftware, welche auch einen Teil einer Software-Entwicklungsumgebung (Software Engineering Environment) ausmachen, wie sie von der IEEE (1990, S. 67) definiert wurde. Mit der Entwicklung und Etablierung neuer Technologien, welche stark an Technologieträger gebunden sind, finden wir mit diesen Ebenen nicht mehr das Auslangen. Unter Technologieträgern können Sachen wie z.B. Kleidung, Gegenstände des alltäglichen Gebrauchs, Gebäude, aber auch Lebewesen wie z.B. Rennpferde oder sogar Menschen verstanden werden. Um dieser Entwicklung Rechnung zu tragen werden dem Ebenenmodell der Informationsverarbeitung nach Ortner (2003b) Technologieträger als Basis hinzugefügt und wir sprechen dann von Informationstechnologie im weiteren Sinn (Abb. 10, 36).

Die Einbeziehung der Technologieträger in das Ebenenmodell stellt auch eine Erweiterung der Auffassung einer Software-Entwicklungsumgebung im Sinne von IEEE (1990, S. 67) dar, da die Technologieträger nur zum Teil als typische Elemente einer solchen Umgebung gesehen werden können. Spätestens mit der Einbeziehung von Lebewesen als Technologieträger wird der von der IEEE-Definition gesetzte technische Rahmen weit überstiegen. Für den vorliegenden Kontext erfolgt also eine Erweiterung des Verständnisses von der Software-Entwicklung hin zur Anwendungsentwicklung um zwei Ebenen, die erste Ebene der Erweiterung ist mit der Einbeziehung von Technologieträgern beschrieben.

Auf Basis des nun geklärten Verständnisses von Informationstechnologie kommt in einer nächsten Stufe der Betrachtung Anwendungssoftware ins Spiel. Diese umfasst Datenressourcen einerseits, und andererseits die Funktionen, für die sie gebaut ist (z.B. Abbildung von einfachen Services). Beide Bereiche, Informationstechnologie im engeren Sinn und Anwendungssoftware, werden zusammengefasst unter dem Begriff Informationssysteme. Als Beispiele seien hier Lagerhaltungssysteme, Buchhaltungsprogramme, Zeiterfassungssysteme usw. genannt. Die Planung und Entwicklung von Informationssystemen im beschriebenen Sinn ist Kerngebiet des Software-Engineering (z.B. Balzert, 2000; Heinrich et al., 2004; IEEE, 1990; Sommerville, 2001; Zuser et al., 2004).

Die bisher besprochenen Komponenten des Ebenenmodells der Informationsverarbeitung (Abb. 10) sind für einen benutzerzentrierten Entwicklungsprozess keineswegs ausreichend. Der Mensch wurde zwar auf Ebene der Technologieträger einbezogen,

diese Integration des Menschen hat jedoch nichts mit seiner Rolle in der Anwenderorganisation, seiner relevanten Arbeitsumgebung und seiner möglichen Beteiligung am Entwicklungsprozess zu tun. Eine Anwenderorganisation kann ein ganzes Unternehmen[10] sein oder auch nur Teile davon umfassen. Konkret ist damit das ablauf- und aufbauorganisatorische Umfeld angesprochen, in welchem ein zu entwickelndes Anwendungssystem später eingesetzt wird. Erst die Berücksichtigung des Benutzers selbst und den relevanten Gegebenheiten in der Benutzerorganisation ermöglicht eine adäquate Benutzerorientierung im Entwicklungsprozess.

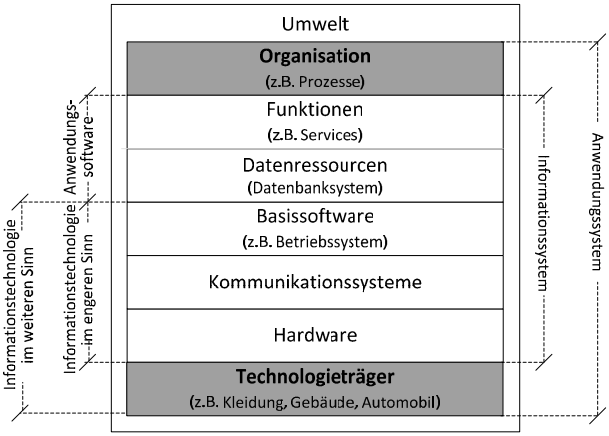

Abb. 10: Erweitertes Ebenenmodell der Informationsverarbeitung

Ein Anwendungssystem zu modellieren bedeutet auch immer, die Anwenderorganisation bzw. einen Teilbereich davon zu modellieren. Für das Ebenenmodell der Informationsverarbeitung heißt das, dass die Organisation (Benutzer und Mitarbeiter sind Bestandteil der Organisation) als zusätzliche Ebene aufzunehmen ist. Im Organisationssystem des Unternehmens ist festgelegt, wie Mensch, Technik und Organisation zusammenwirken. Diese Darstellung erfolgt dual, einerseits durch weitgehend statische Strukturen der Aufbauorganisation, andererseits durch dynamische Strukturen der Ablauforganisation. Die Grundlagen dazu wurden bereits von Nordsieck in den 1950er und 1960er Jahren beschrieben (Nordsieck, 1955 und 1962) und später aktualisiert (Kosiol, 1976; Grochla, 1982).

Mit dem erweiterten Ebenenmodell der Informationsverarbeitung ist eine ganzheitliche Architektur sowohl für den Aufbau von Anwendungssystemen als auch für die

10 Es sind hier auch immer Verwaltungsorganisation und Non-Profit-Organisationen mit eingeschlossen.

nutzerorientierte Entwicklungsarbeit eingeführt. Eine Entwicklungsarbeit die alle Ebenen umfasst, d.h. es werden Technologieträger, der organisationale Kontext des Anwendungssystems und auch relevante Einflüsse der Umwelt berücksichtigt, nennen wir Anwendungssystementwicklung, kurz Anwendungsentwicklung. Die Ergebnisse dieser Arbeit bezeichnen wir als Anwendungssysteme. In diesem Sinne wird der Begriff „Anwendungssystem", „Anwendungssystementwicklung" (application engineering) und „Anwendungsentwicklung" in den weiteren Ausführungen verwendet.

Der Vollständigkeit halber sei hier darauf hingewiesen, dass in der Literatur unterschiedliche Auffassungen zur Abgrenzung der Begriffe „Anwendungssystem" versus „Informationssystem" bestehen (z.B. Ortner, 2003b; Hansen & Neumann, 2005, S. 84; Stahlknecht & Hasenkamp, 2005, S. 204; Ferstl & Sinz, 2006, S. 9-10; vom Brocke, 2006b, S. 11-12, aufbauend auf Teubner, 1999, S. 26; Lonthoff, 2007, S. 13-14). Diese Unterschiede resultieren einerseits aus einer mehrdeutigen Verwendung des Begriffs Information, andererseits aus der unterschiedlichen Anordnung der Ebenen eines Informationssystems. Die vorliegenden Ausführungen basieren darauf, dass der Begriff „Anwendungssystem" der umfassendere von beiden ist und auf dem erweiterten Ebenenmodell der Informationsverarbeitung nach Ortner (2003b) aufbaut.

2.3.2 ProCEM® - Ein Rahmen zur ganzheitlichen Entwicklungsarbeit

Im Sinne einer Konstruktionslehre für Anwendungssysteme, wie sie mit der sprachbasierten Informatik u.a. von Ortner (2005) beschrieben wird, geht es bei der Anwendungsentwicklung nicht lediglich um eine methodische Unterstützung der Entwicklungsarbeit, sondern es wird der Anspruch auf Ganzheitlichkeit gestellt (vgl. Abschnitt 2.1.3, S. 20f).

Eine Erweiterung dieser Sichtweise ergibt sich aus dem Verlauf der Anwendungsentwicklung aus Sicht der Semiotik. Der Verlauf beginnt bezeichnenderweise bei der Pragmatik, also im Anwendungsbereich eines Systems:

- **Pragmatik** – Gebrauch der sprachlichen Ausdrücke im Anwendungsbereich – Was leistet ein sprachlicher Ausdruck, d.h. welche Operationen oder Handlungen werden dadurch ausgelöst? Welche Wirkung haben sprachliche Ausdrücke auf den Interpretanten (Anwender oder Computer).
- **Semantik** – Extensionale oder intensionale *Bedeutung* sprachlicher Ausdrücke. – Worauf nimmt ein sprachlicher Ausdruck Bezug bzw. welcher Inhalt bzw. Sinn wird mit dem Ausdruck transportiert oder vermittelt?

- **Syntax** – Computer funktionieren auf Basis von Syntax – Welche Form haben sprachliche Ausdrücke? Ist ein Ausdruck gültig aufgrund der Form allein?

Einen Rahmen für eine ganzheitliche Entwicklungsarbeit bietet ProCEM® (Process-Centric Enterprise Modeling & Management; Ortner, 2008b; Abb. 11). Diese *Methodologie* unterstützt prozesszentrisch den Lebenszyklus von Anwendungssystemen, d.h. die Entwicklung geht von der Prozessrekonstruktion aus und strebt die Serviceermittlung an. Die Prozesszentrik umschließt als Denkweise die Modellierung, Implementierung, Stabilisierung und den Betrieb eines Systems bis hin zu dessen Re-Engineering.

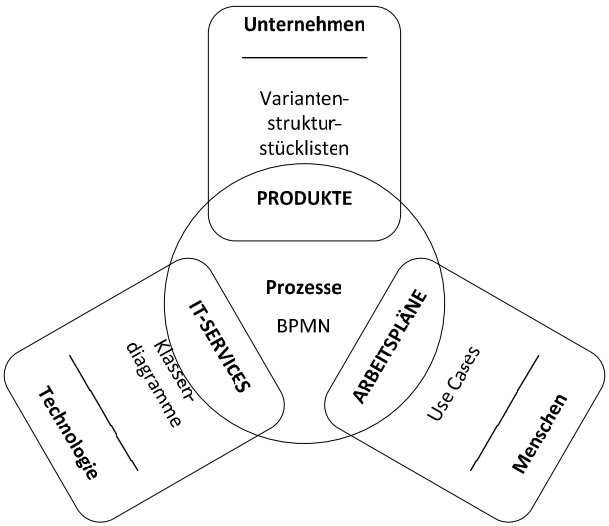

Abb. 11: ProCEM® - Methodologie für ganzheitliche, prozesszentrische Modellierung und Management von Anwendungssystemen

Sie garantiert damit die kontinuierliche Verbesserung der Prozesse und des Technologieeinsatzes eines Unternehmens. ProCEM® bildet auch den Rahmen für verteilte Entwicklungsarbeit und bietet gleichzeitig Richtlinien für die organisationszentrisch-sprachkritische Entwicklung und das Management von Anwendungssystemen. Die Integration des Multipfad-Vorgehensmodells (Abschnitt 2.3.3, S. 41ff) in ProCEM® ermöglicht zusätzlich eine systematische Berücksichtigung von *Nutzungsanforderungen* bereits ab einer frühen Entwicklungsphase. Die Grundlegung von ProCEM® in der sprachbasierten Informatik begünstigt die systematische Integration im Raum zwi-

schen Mensch, Organisation und Technik. Im Zentrum dieser Integration stehen die Prozesse.

Das idealtypische Vorgehen innerhalb von ProCEM® lässt sich aus Sicht der Anwenderorganisation mit der organisationzentrischen Vorgehensweise beschreiben (Ortner & Eller, 2008). Die organisationszentrische Vorgehensweise umfasst neben den umfangreichen vorbereitenden Entwicklungstätigkeiten zusätzlich organisations- und projektspezifische Vorgänge wie z.B. eine Verbesserung und Optimierung von Arbeitsabläufen sowie eine Wandlung in Richtung Serviceorientierung. Es handelt sich dabei um Aktivitäten, die, wenn sie in das Entwicklungsprojekt integriert werden, zu zusätzlichen Synergien aus Sicht der Anwenderorganisation führen können. Zusätzlich ist in dieser Vorgehensweise vorgesehen, die Rekonstruktionsergebnisse bzw. Modellierungsergebnisse dual zu verwenden. In der folgenden näheren Beschreibung dieser Vorgehensweise wird diese Ergebnisverwendung deutlich.

Die Analysearbeiten vor bzw. zu Beginn eines Entwicklungsprojekts bilden den ersten Schwerpunkt im organisationszentrischen Vorgehen. Dabei geht es u.a. um die Umsetzung des Prinzips „application follows processes“ mit einer Orchestrierung von Menschen, Abläufen und Systemen im Unternehmen innerhalb des Rahmens von PROCEM®. Ausgangspunkt für diese Orchestrierung ist die statische Aufbauorganisation einer Organisation. Die Erhebung der Aufbauorganisation zielt bereits auf die Nutzung des künftigen Anwendungssystems ab, z.B. indem sämtliche relevante Stakeholder und deren Rollen in Bezug auf das Anwendungssystem identifiziert werden. Kernpunkt und Zentrum der Analysearbeiten ist jedoch die Ablauforganisation und damit verbunden die Prozessrekonstruktion. *Rekonstruktion* ist hier zu verstehen als eine „Nach-Konstruktion“ von bereits bestehenden Arbeitsabläufen bei deren gleichzeitiger Verbesserung. Mit „Verbesserung“ sind Harmonisierungen von Abläufen über mehrere Organisationseinheiten ebenso gemeint wie Optimierungen innerhalb von Arbeitsabläufen.

In der organisationszentrischen Vorgehensweise werden alle Optimierungspotenziale identifiziert und erfasst, also auch jene, die keinen Systemzusammenhang erwarten lassen. Das Vorgehen basiert auf dem Konzept der Rekonstruktion von Wissen (Ortner, 2005; Ghani, Koll, Kunz, Sahing & Yalcin, 2007). Aus Sicht des Menschen gibt es drei Möglichkeiten einer Optimierung der Arbeitsabläufe:

- Reduktion der Arbeitsbelastung durch Automatisierung.
- Unterstützung des Menschen bei der Verrichtung der Arbeit z.B. durch interaktive Systeme.
- Verbesserung der Qualifikation der Mitarbeiter.

Die Rekonstruktionsarbeit führt in der Regel zu Ergebnistypen wie Prozessmodellen, Partizipationsmodellen, Modellen der Arbeitsschritte sowie Modellen von Daten und Datenfluss. Die Grenzen zwischen diesen Ergebnistypen sind nicht immer scharf zu ziehen. Die Darstellung dieser Ergebnisse erfolgt vorerst primär in Form von Aussagen. Es können aber auch Diagrammsprachen verwendet werden, um Sachverhalte für Anwender zu visualisieren. Sprachen aus dem Sprachkanon der OMG (2008a), z.B. die BPM-Notation (Business Process Modeling Notation) oder Use Cases aus der UML-Familie (Unified Modeling Language), bieten sich dafür an. Für Kommunikationszwecke im Projekt reicht es z.B. aus, Partizipationsmodelle in Gestalt von Use-Cases und Arbeitsabläufe mit Hilfe der BPM-Notation darzustellen. Dabei wird speziell für diesen Zweck die Anzahl der verfügbaren Sprachelemente reduziert. Mit vergleichsweise wenigen Sprachelementen können Arbeitsabläufe visualisiert werden, die in dieser Form auch für Modellierungslaien gut verständlich sind. Die Kommunikation mit den Benutzern wird auf Basis solcher Diagramme nicht gefährdet. Zusätzlich besteht in der organisationszentrischen Vorgehensweise die Anforderung, weitgehend modulare Modellierungsergebnisse zu erzielen.

Die modellierten Abläufe umfassen den gesamten Arbeitsbereich des jeweiligen Benutzers, d.h. es sind manuelle Arbeitsschritte gleichermaßen enthalten wie solche Arbeitsschritte, die später mit Software als Services realisiert werden sollten. Damit liegt eine modulare Beschreibung von Tätigkeiten, die im Unternehmen zu verrichten sind, vor. Neben der Spezifizierung von Services können diese Beschreibungen wiederum zu flexiblen Stellenbeschreibungen zusammengefügt werden. Somit ist eine Rückkoppelung zur Aufbauorganisation gegeben. Damit ist eine wesentliche Grundlage für eine prozessorientierte und elastische Aufbauorganisation geschaffen. Abb. 12 (S. 41) fasst die organisationszentrische Vorgehensweise grafisch zusammen (Ortner & Eller, 2008).

Mit ProCEM® (Abb. 11, S. 38) kommt für Entwicklungsvorhaben eine Methodologie zum Einsatz, die bereits in verschiedenen Projekten in Wirtschaft und Verwaltung den Weg einer aus sich selbst heraus dynamisch und kraftvoll agierenden Unternehmung, im Sinne von Hamel & Välikangas (2003) erfolgreich gegangen ist. Dies geschieht jeweils durch das Auflösen festgefahrener und sowohl Effizienz als auch Effektivität hemmender Situationen, sowie durch die Ausbalancierung der drei unternehmerischen Säulen Mensch, Organisation und Technik. Eine spezielle Charakteristik von ProCEM® ist die Tatsache, dass damit nicht nur „bessere Mittel für beabsichtigte Zwecke“ implementiert werden, sondern dass anschließend das dergestalt erneuerte Unternehmen aus sich selbst heraus in der Lage ist, auf dem nunmehr eingeschlagenen Erfolgskurs zu bleiben. ProCEM® als Methodologie leistet also zweierlei: Sie (er)schafft neue Mittel und führt gleichzeitig in Organisationen zu einem neuen Bewusstsein von Aktionsgemeinschaften wie z.B. Projektteams (Ortner, 2008b). Zur

methodischen Unterstützung der Entwicklungsarbeit im Rahmen von ProCEM® steht in der sprachbasierten Informatik u.a. das Multipfad-Vorgehensmodell zur Verfügung.

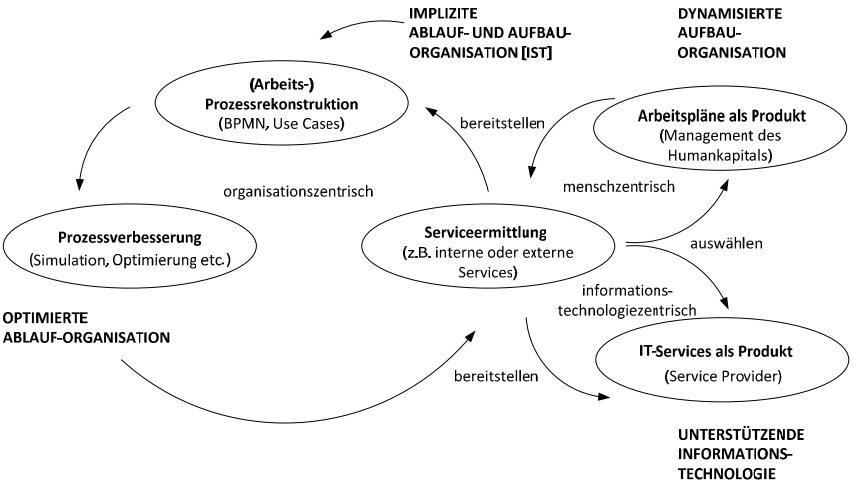

Abb. 12: Organisationszentrische Entwicklung von Anwendungssystemen

2.3.3 Multipfad-Vorgehensmodell

2.3.3.1 Einleitung und Übersicht

Das Multipfad-Vorgehensmodell (Ortner, 2005, S 44-182) ist ein Netz aus Aktivitäts- und Resultatstypen zur ganzheitlichen Entwicklung von Anwendungssystemen. Das Vorgehen gliedert sich in sieben Phasen bzw. Vorgangstypen (V1 – V7). Die konkrete Entwicklungsarbeit im Rahmen dieses Modells kann unterschiedlichen Pfaden folgen – deshalb auch der Name „Multipfad-Vorgehensmodell". Die Hauptpfade sind in Abb. 13 skizziert. Abhängig von der Problemstellung, vom Anwendungskontext und von den Anforderungen der Auftraggeberorganisation (= Anwender der Softwarelösung) und den Möglichkeiten der Auftragnehmerorganisation (= Hersteller der Softwarelösung) sind zusätzliche Pfade gangbar. Auch eine Kombination von Standardlösung, Komponentenlösung und Individuallösung zu einer unternehmensspezifischen Gesamtlösung ist denkbar und wird methodisch unterstützt.

Ausgangspunkt der Entwicklungsarbeit ist in der Regel eine vorliegende Mangelsituation oder ein Konflikt im Anwendungsbereich. Den dargestellten Hauptpfaden folgend, werden im Rahmen von Voruntersuchungen (V1) interne und externe Anforderungen erhoben und in einem Pflichtenheft dokumentiert. Eine Bedingungsmatrix

dient in dieser Phase zum Ordnen der relevanten Themen aus den unterschiedlichen betroffenen Disziplinen (*Transdisziplinarität*). Ist ein Outsourcing der Entwicklungsarbeit bzw. die Anschaffung einer Standardsoftware vorgesehen, folgt hier bereits die Stabilisierung (V6).

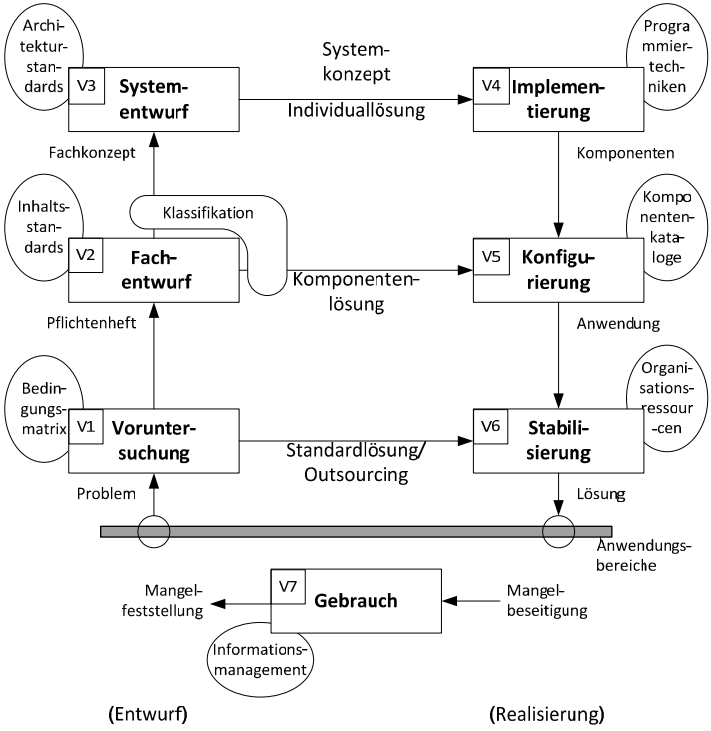

Abb. 13: Multipfad-Vorgehensmodell

Kommt ein Outsourcing der Problemlösung bzw. ein Einsatz einer Standardsoftware nicht in Frage, setzt sich der Entwicklungsprozess im Fachentwurf (V2) fort. Im *Vorgangstyp* Fachentwurf wird eine fachliche Lösung zur spezifischen Aufgabenstellung „rekonstruktiv" ermittelt. Als Mittel zur Integration werden im Fachentwurf „Inhaltsstandards" eingesetzt. Inhaltsstandards haben integrativen Charakter wie z.B. die rekonstruierte Fachsprache des Anwendungsbereiches, Wissensartefakte, Thesauri und Fachterminologien. Sie können allein oder in Kombination zum Einsatz kommen. Es liegt nahe, dass die im Anwendungsbereich vorherrschenden Fachbegriffe auch Basis für die Software sind.

Fragestellungen hinsichtlich der Entscheidung für ein bestimmtes Lösungsparadigma (z.B. Datenbank-Anwendung, Workflow-Management-System, Expertensystem) sind eindeutig nach dem Fachentwurf zu behandeln, entweder im Vorgangstyp Systementwurf (V3) oder im Vorgangstyp der Konfigurierung (V5) von Komponentenlösungen. Wenn bereits beim Fachentwurf technologiebedingte Anforderungen eingebracht werden, können diese die Erarbeitung optimaler fachlicher Lösungen behindern (Ortner, 1993, S. 11). Diesem Abstraktionsprinzip folgen wir in der vorliegenden Arbeit, somit bleiben technologische Aspekte aus der Betrachtung ausgeklammert.

Die Resultate aller Aktivitäten der Phase „Fachentwurf" bilden das Fachkonzept. In Form einer Sammlung normierter Aussagen stellt es nach einer Klassifizierung der Aussagen die Basis für den Systementwurf (V3) bzw. für die Konfigurierung (V5) einer Komponentenlösung dar. Im Falle einer Eigenentwicklung folgt nach der Systemspezifikation (V3) die Implementierung (V4) d.h. die eigentliche Programmierung des Systems. Teilweise ist es bereits möglich, ein System aus Komponenten bzw. Services zu orchestrieren. Die Orchestrierung von Services findet im Vorgangstyp Konfigurierung (V5) statt. Bevor nun das Anwendungssystem an die Anwenderorganisation „ausgeliefert" wird, gilt es letzte Einstellungen und Anpassungen für die Anwenderorganisation vorzunehmen. Der zugehörige Vorgangstyp ist die Stabilisierung (V6). Aus Sicht der Technik wird ein Anwendungssystem parametrisiert und bei Bedarf noch angepasst. Aus Sicht des Menschen wird in dieser Phase von Stützung der Anwender gesprochen. Darunter werden alle Maßnahmen subsumiert, die zur Vorbereitung der Anwender auf die Anwendung des Systems erforderlich sind, wie z.B. Schulungen, Durchführung von Benutzertest und Probebetrieb, Bereitstellung von Benutzerhandbüchern.

Die *Vorgangstypen* sind nicht unbedingt sequentiell abzuarbeiten. Es sind Iterationen genauso möglich wie ein Vorausdenken, Rücksprünge oder auch ein gänzliches Überspringen einzelner Vorgangstypen. Aufgaben wie Projektmanagement, Qualitätsmanagement usw. werden in der Phasenbeschreibung lediglich am Rande berührt. Sie verstehen sich jedoch als begleitend über den gesamten Entwicklungsprozess. Werden die Sprünge zwischen einzelnen Vorgangstypen und Aufgaben nachgezeichnet, entsteht eine Vernetzung mit System, wie dies auch bei einer Darstellung der Spielzüge eines Schachspiels der Fall ist. Diese „Schachbrettmetapher" soll Systematik und Freiheitsgrade der möglichen Pfade im Multipfad-Vorgehensmodell zu veranschaulichen.

Einzelne Phasen des Multipfad-Vorgehensmodells werden in Anlehnung an Ortner in den folgenden Abschnitten näher beschrieben. Eine ausführliche Beschreibung findet sich bei Ortner (2005, S 44-182). Nachfolgend wird insbesondere auf die Phasen der Voruntersuchung, des Fachentwurfs und der Stabilisierung mit den jeweiligen Über-

gängen eingegangen. Dies sind jene Phasen, in denen die Gebrauchstauglichkeit von Anwendungssystemen effektiv und nachhaltig beeinflusst werden kann. Die Beeinflussung von *Akzeptanz* ist hier ebenfalls gegeben, jedoch kann diese auch phasenunabhängig erfolgen. In den Phasen Voruntersuchung und Fachentwurf sprechen wir aus methodischer Sicht auch von sprachkritischer Entwicklungsarbeit, wenngleich sich die sprachkritische Vorgehensweise (im Sinne eines *Paradigmas*) über die gesamte Entwicklungsarbeit bemerkbar macht. Der Fokus der sprachkritischen Systementwicklung liegt in der Informationserhebung und Anforderungsentwicklung sowie in der Transformation dieser Erkenntnisse in Systemanforderungen. Die Abgrenzung zwischen diesen Phasen ist aus methodischer Sicht nicht immer eindeutig möglich. Methoden werden in Abschnitt 2.3.4 (S. 52ff) besprochen und nach Möglichkeit den Phasen zugeordnet.

2.3.3.2 Voruntersuchung

Die Beschreibung des Modells beginnt bei der Voruntersuchung auf Seiten der Herstellerorganisation. Das heißt, eine Problembeschreibung von Seiten der Anwenderorganisation liegt bereits vor. Ein wesentlicher Teil der Anforderungsermittlung und auch der eigentliche Beginn eines Projektes zur Entwicklung eines neuen Anwendungssystems oder zur Modifikation und Verbesserung eines schon vorhandenen Anwendungssystems (Re-Engineering) sind bereits vorweggenommen. Die systematische Erfassung und Einbringung der Mangelsituation als Problembeschreibung z.B. in Form einer *Anforderungsspezifikation* von Seiten der Anwenderorganisation ist in der Modellbeschreibung unterrepräsentiert. Das heißt, dass weder die systematische Erarbeitung der Problembeschreibung noch die Möglichkeiten der systematischen Integration des Usability Engineering im Sinne einer Rekonstruktion des Anwenderwissens zur Benutzung einer Systemlösung beschrieben sind. Diesem Defizit wird mit der Erweiterung des Multipfad-Vorgehensmodells im Zuge Integration auf konzeptioneller Ebene begegnet (vgl. Abb. 35, S. 168).

Der Vorgangstyp Voruntersuchung (V1) geht also von einem bestimmten Problem aus, welches durch die Entwicklung und die Einführung des neuen oder modifizierten Anwendungssystems gelöst werden soll. Die Problembeschreibung ist durch die Anwenderorganisation erfolgt. Trotzdem ist es aus Sicht der Herstellerorganisation notwendig, dass Ziele und Zwecke, und die damit verbundenen Voraussetzungen und Anforderungen an das zu entwickelnde System, sowohl aus fachlicher wie auch aus grober technischer Sicht zu er- oder überarbeiten und bei Bedarf klarzustellen sind. Meist erfolgt diese Darstellung unter dem Titel „Rahmenbedingungen“. Die Verantwortung für die Voruntersuchungen liegt bei der Herstellerorganisation.

Die Herausforderung bei den Voruntersuchungen liegt in der Interdisziplinarität. Es ist eine enge und aktive Zusammenarbeit zwischen Anwenderorganisation und Herstel-

lerorganisation erforderlich. Verfügt das Anwenderunternehmen über eine eigene Entwicklungsabteilung, sind unter Anwenderorganisation die Fachabteilungen zu verstehen, für die das System entwickelt wird, und als Herstellerorganisation ist in diesem Fall die Entwicklungsabteilung zu betrachten.

Diese Phase ist daher geprägt von einer intensiven Kommunikation zwischen Fachleuten aus den unterschiedlichsten Fachbereichen (Interdisziplinarität) und Anwendern mit unterschiedlichem Bildungsniveau und unterschiedlichem Wissen über Aufgaben, Strukturen und Abläufe im Anwendungsbereich. Dabei ist sowohl die Granularität dieses Wissens als auch das Wissen über die, dem Wissen zugrunde liegende, Historie von Bedeutung. Hinzu kommt, dass Mitarbeiter aus der Führungsebene mit entsprechend bereichsübergreifendem Wissen (Orientierungswissen) und Tiefenwissen (Verfügungswissen) ebenso in diese Phase integriert sein sollten, jedoch nicht dominierend. Entwicklungsexperten (Informatiker, Programmierer, Oberflächendesigner usw.) bringen mit ihrem technischen Know-how über Konzeption, Implementierung von Systemen und technischen Infrastrukturen zusätzliche Komplexität in diese Kommunikation. Ihre Denk- und Handlungsweise ist in der Regel sehr technisch bzw. technologisch ausgerichtet, was sich nicht zuletzt über den Sprachgebrauch äußert. Gehören die Gesprächspartner unterschiedlichen Unternehmen an, kommen neben den Herausforderungen hinsichtlich der Kommunikation oft auch noch andere Restriktionen hinzu. Diese können beispielsweise bedingt sein durch unterschiedliche zeitliche Verfügbarkeit, unterschiedlich gelagerte wirtschaftliche Interessen hinsichtlich des Projektes, bereits vorgefasste Vorstellungen zur Systemlösung oder durch vereinbarte Kostenbeschränkungen. Die Komplexität der zu bewältigenden Aufgaben erhöht sich.

Besonders in dieser Analysephase, aber auch in der Konzeptionsphase (Fachentwurf) ist eine enge Zusammenarbeit zwischen den beteiligten Organisationen gefordert, damit ein gebrauchstaugliches System entstehen kann, das den Anforderungen der Nutzer bestmöglich gerecht wird. Missverständnisse und Kommunikationsprobleme zwischen Systemanwendern und Systemherstellern beruhen nämlich nicht nur auf der Tatsache, dass hier unterschiedliche Disziplinen mit verschiedenen Sichtweisen aufeinander treffen. Die Kooperation wird zusätzlich dadurch erschwert, dass verschiedene Interessensgruppen zusammenarbeiten. Systemhersteller verfolgen in der Regel an erster Stelle das Ziel der Gewinnmaximierung, während ein Auftraggeber die Kosten gering halten möchte. Es herrscht immer ein Interessenkonflikt zwischen Auftragnehmer (Systemhersteller) und Auftraggeber (Systemanwender). Hinzu kommt eine asymmetrische Informationsverteilung. Solche und andere Zielkonflikte machen es unerlässlich bereits in dieser Phase eventuell notwendige Abstimmungen und Austauschbeziehungen (Trade-off) im Hinblick auf das Anwendungssystem und die Systemnutzung transparent zu machen.

Sowohl für die Kommunikation als auch für die systematische Erfassung der Ergebnisse der Arbeiten im Rahmen der Voruntersuchung stehen ausreichend Methoden zur Auswahl. Die Herausforderung liegt dabei eher in der Methodenvielfalt in Verbindung mit der Anwendungskompetenz der Beteiligten. Auch zeitliche Restriktionen treten in der Praxis immer wieder in den Vordergrund. Eine nähere Betrachtung von ausgewählten und einsetzbaren Methoden aus Sicht der sprachbasierten Informatik findet sich in Abschnitt 2.3.4 (S. 52ff).

Die Phase der Voruntersuchung entspricht einem wesentlichen Teil des Requirements Engineering. Sämtliche Ressourcen, die für das System erforderlich sind, unterliegen der Analyse. Die Ergebnisse werden in der Regel in einem Pflichtenheft dokumentiert. Die Inhalte des Pflichtenhefts sind wesentlich für den Fachentwurf und kommen auch in der Stabilisierungsphase wieder zum Tragen.

Wird ein System neu entwickelt, weiterentwickelt oder ist eine Orchestrierung von Komponenten und Services beabsichtigt, folgt den Voruntersuchungen der Vorgangstyp Fachentwurf. Soll eine Auslagerung der Entwicklungsarbeiten erfolgen, kann aus Sicht der Anwenderorganisation bereits hier in die Phase der Stabilisierung verzweigt werden. Dies ist auch dann der Fall, wenn die Einführung einer Standardsoftware vorgesehen ist. Für beide Fälle ist ein Teil des Fachentwurfs, im speziellen die Rekonstruktion des Anwenderwissens für die Benutzung der Software-Lösung, in die Stabilisierung zu verlegen.

2.3.3.3 Fachentwurf

Im Vorgangstyp Fachentwurf (V2) wird ein fachliches Lösungskonzept für das zu entwickelnde Anwendungssystem erstellt. Dazu wird das Fachwissen eines Anwendungsgebietes in Zusammenarbeit mit den Anwendern methodisch rekonstruiert (Ortner & Söllner, 1989). Die Rekonstruktion beginnt mit der Wissensbeschaffung über Sachverhalte und Abläufe der Unternehmung bereits in der Voruntersuchung und wird im Fachentwurf weitergeführt. Die sprachkritische Rekonstruktion erstreckt sich somit über beide Vorgangstypen.

Das notwendige Wissen wird vor allem in kommunikativen Prozessen zwischen Experten und Anwendern gesammelt. Zur Sammlung der relevanten Informationen werden Erhebungstechniken eingesetzt (vgl. Abschnitt 2.3.4, S. 52ff). Die Kommunikation zwischen Anwendern und Entwicklern erfolgt in einer allgemeinverständlichen, natürlichen Sprache. Dies ist grundlegend, um ein gegenseitiges Verständnis aufzubauen. Als Zwischenergebnis liegen Prosaaussagen vor. Diese Aussagen werden in einem weiteren Schritt normiert. Die Resultate aller Aktivitäten der Phase „Fachentwurf“ bilden das Fachkonzept, zusammengestellt in einer Sammlung relevanter Aussagen in einer normierten Sprache auf Basis geklärter Fachbegriffe. Das Fachkon-

zept ist eine vollständige Beschreibung des Systems und dient als Grundlage für den Systementwurf.

Die gesammelten und erarbeiteten Prosaaussagen sind oft missverständlich, widersprüchlich und unvollständig. Grund dafür sind Begriffe der natürlichen Sprache, die teilweise missverständlich und widersprüchlich verwendet werden (Ortner & Söllner, 1989). Das liegt einerseits an der Nicht-Eindeutigkeit der natürlichen Sprache an sich, und andererseits an spezifischen Defekten von Fachbegriffen. Die Klärung und eindeutige Definition solcher Begriffe wird durch die sprachliche Rekonstruktion im engeren Sinn erreicht.

Dazu werden diese Begriffe systematisch geklärt (vgl. Abschnitte 2.2.6, S. 32ff und 3.2.3, S. 114ff), Inhaltsstandards herangezogen, aber auch geschaffen, und die Ergebnisse in Lexika (Glossar, Repository, Fachwörterbuch) hinterlegt („semantische Normierung"). Ein weiterer Schritt der Rekonstruktionsarbeit liegt in der strukturellen Normierung der vorliegenden Aussagen aus grammatikalischer Sicht. Den Aussagen wird dabei eine einfache, klare Grammatik zugrundegelegt. Es entsteht eine reglementierte Sprache, auch Normsprache genannt. Dieses Vorgehen entspricht einer semantischen Integration. Diese gelingt nachhaltig nur dann, wenn die Benutzer explizit und systematisch in diesen Prozess einbezogen werden. Damit steht die sprachliche bzw. fachliche Effizienz im Fachentwurf im Vordergrund.

Die Erstellung des Fachentwurfs ist „problemorientiert". Damit ist gemeint, dass eine fachliche Anwendungslösung vom Standpunkt der Anwender und des Anwendungsgebiets neutral gegenüber einer spezifischen technischen Lösungsarchitektur (objektorientiert, datenorientiert, prozessorientiert) erstellt wird und somit in Bezug auf die Anwendungssysteme technologieneutral ist. Diese Besonderheit wird auch unter dem Begriff „*methodenneutral*" zusammengefasst. Das frühe Einbringen von technologiebedingten Anforderungen in der Konzeptionsphase eines Systems kann nach Ortner (1997) die Erarbeitung optimaler fachlicher Lösungen behindern.

Für die Praxis bedeutet „methodenneutral", es kommt keine *Methode* (z.B. eine Diagrammsprache) bevorzugt zum Einsatz. Entwicklungsrelevante Sachverhalte des Anwendungsbereiches werden mittels einer reglementierten Sprache klar und eindeutig beschrieben. Die reglementierte Sprache (= Konstruktionssprache) (Ortner, 1997, S. 86-87) zeichnet sich aus durch einfache Satzbauweise, hohe Affinität zur natürlichen Sprache und Verwendung von geklärten Fachbegriffen (Ortner, 2003a, S. 20).

Aus dem Blickwinkel der Informationstechnologie (Basis- und Anwendungssoftware und die ihnen zugrundeliegenden Sprachmodelle) kann *Methodenneutralität* auch als Technologieunabhängigkeit bezeichnet werden. Der Fachentwurf ist „neutral" (aber nicht unabhängig) gegenüber der Frage, ob z.B. eine Workflow-Management-

Anwendung oder eine Datenbank-Management-Anwendung entwickelt wird. „Neutral“ ist demnach bereits eine Abschwächung gegenüber „unabhängig“. Gäbe es ein durchgängiges Sprachmodell, genauer ein sprachkritisches Prozessmodell - sowohl für den Anwender (Organisationslehre) als auch für die Technologie (SOA) - dann stellte sich die Frage, ob eine *Methodenneutralität* im Fachentwurf überhaupt noch erforderlich ist. Die Praxis zeigt, dass wir über dieses (sprachkritische) Prozessmodell bis dato (noch) nicht verfügen. Erste Schritte eines solchen Modells sind in Abb. 7 (S. 31) dargestellt, sie reichen aber noch nicht aus. Weder IT-Industrie noch die Anwender in ihrer Aufbau- und Ablauforganisation haben den Ansatz der Methodenneutralität umgesetzt.

In einem nächsten Schritt erfolgt eine Klassifizierung des Aussagenbestandes. Zum einen passiert dies zur Überprüfung auf Vollständigkeit des Aussagenbestandes, und zum anderen gilt es, die Aussagen zu ordnen. In diesem Übergang von methodenneutralem zu methodenspezifischem Vorgehen wird berücksichtigt, welche Lösungsarchitektur entsprechend dem verwendeten Anwendungssystemtyp angestrebt wird, und es werden dementsprechende spezielle Spezifikationssprachen eingesetzt (Lehmann, 1999).

In der Literatur ist der Übergang zwischen Methodenneutralität und Methodenspezifität und deren Zuordnung zu den Phasen des Multipfad-Vorgehensmodells (Fachentwurf und Systementwurf) nicht eindeutig geklärt. Es gibt Quellen, die schon den Fachentwurf zweiteilen, nämlich in einen methodenneutralen und einen darauf folgenden methodenspezifischen Teil (Schienmann, 1997; Lehmann & Ortner, 1996 u. 1999; Lehmann 1999). Andere sprechen wiederum davon, dass der Fachentwurf rein methodenneutral ist und sie somit den methodenspezifischen Teil zum Systementwurf zählen (Ortner, 2005). Wir gehen davon aus, dass im Übergang zwischen Fachentwurf und Systementwurf die methodenneutral erstellte Aussagensammlung unter Berücksichtigung der relevanten Lösungsarchitektur klassifiziert und methodenspezifisch transformiert wird. Darüber hinaus erfolgt die Modellierung im Fachentwurf einem *normsprachlichen* Entwurfsansatz. Dieser ist, neben der Technologieneutralität, gekennzeichnet durch *Methodenneutralität*, *Normsprachlichkeit* und einem materialen Charakter (*Materialsprachlichkeit*) (Jablonski et al., 1999).

Nach Klassifizierung der Aussagen dient das Fachkonzept gleichermaßen als Basis für den Systementwurf (V3) wie für die Konfigurierung (V5) einer Komponentenlösung. Die technische Sicht erfolgt erst im Systementwurf bzw. wird im zweiten Teil des Fachentwurfs erstmals betrachtet.

Das Konzept der sprachkritischen Systementwicklung, wodurch sich das Multipfad-Vorgehensmodell auszeichnet, ist schwerpunktmäßig im Fachentwurf verankert. Die sprachkritische Rekonstruktion des fachlichen Softwareentwurfs reduziert sich kei-

nesfalls auf die Datenmodellierung. Auch Organisations-, Funktions-, Prozess- und Steuerungssicht werden nach dieser Methode erarbeitet und dargestellt.

2.3.3.4 Vom Systementwurf zur Konfigurierung

Im Systementwurf wird das Fachkonzept modifiziert zur weiteren, vor allem zur technischen Konkretisierung im Hinblick auf die Implementierung. Der Systementwurf ist ein Konzept für einen bestimmten Anwendungssystemtyp. In diesem Konzept wird die Systemarchitektur beschrieben. Darauf aufbauend erfolgt die Ausgestaltung von statischen und dynamischen Strukturen, z.B. in Ausprägung von Benutzeroberflächen, die (technische) Spezifikation von Datentypen, Prozesstypen, *Workflows*, Schnittstellen usw. Dies geschieht durch Spezifikationssprachen, wie sie z.B. als Modellierungssprachen in der UML-Familie verfügbar sind. Des Weiteren erfolgt hier die konkrete Modularisierung mit Definition der Schnittstellen und Festlegung von relevanten Testfällen. Der Systementwurf ist Grundlage für die Implementierung.

Im Vorgangstyp Implementierung werden alle Systemteile, die im Systemkonzept spezifiziert wurden, programmiert, sofern diese nicht als Services oder Komponenten verfügbar sind und dieser Art integriert werden sollten. Hier steht die technische Effizienz im Vordergrund. Zu den Teilaufgaben dieser Phase gehören die Programmierung, die Umsetzung des Konzepts für die Benutzeroberfläche nach den Vorgaben des Systementwurfs und die physische Datenorganisation. Die Erstellung einer vollständigen Dokumentation sowie das Testen des Programms und die begleitende Qualitätssicherung sind in dieser Phase zusätzlich wichtig. Inwieweit eine automatische Umsetzung von in Modellierungssprachen (z.B. BPML) spezifizierten Abläufen in Code erfolgen kann und soll, wird in diesem Rahmen nicht weiter betrachtet. Auch im Zuge der Implementierung sind Iterationen vorgesehen, insbesondere in Verbindung mit dem Prototyping und durchzuführenden Tests.

Die Phase der Konfigurierung folgt der Implementierung. Für den Fall, dass serviceorientiert bzw. komponentenorientiert, aber auch modular entwickelt wird, verzweigt der Entwickler bereits nach dem Fachentwurf in die Konfigurierung. Abhängig von der vorgesehen Architektur werden die Komponenten, Services oder Module in der Phase der Konfigurierung orchestriert. Diese Phase gewinnt mit der zunehmenden Service-Orientierung von Unternehmen (Gartner, 2005) und dem damit verbundenen Potenzial immer mehr an Bedeutung. Services sind eine besondere Form von Komponenten. Sie bieten in sich abgeschlossene Dienste an, die zu einem Prozess oder einem Anwendungssystem zusammengesetzt werden können. Serviceorientierte Architekturen (SOA) (z.B. Starke & Tilkov, 2007; IBM, 2008) bieten Infrastrukturen an, mit denen vorhandene Komponenten der Informationstechnologie so koordiniert werden können, dass ihre Leistungen als Dienste (Services) zur Verfügung stehen. Gleiche Dienste können in unterschiedlichem Kontext eingesetzt werden.

Es ist durchaus vorstellbar, dass Konzerne diese Technologie unternehmensintern nutzen und somit eine weltweite Standardisierung von Arbeit (entweder als Service oder als Produkt) erreichen können. Der Ansatz folgt nach Erl (2008) grundsätzlich acht Prinzipien. Dazu gehören u.a. die Wiederverwendbarkeit von Systemteilen und die Flexibilität von ganzen Anwendungssystemen. Prozesse können durch den Austausch oder das Hinzufügen von Services relativ schnell und einfach an neue Anforderungen angepasst werden. Die Auswahl und zweckorientierte Zusammensetzung der Komponenten erfolgt auf Basis der klassifizierten Aussagen aus dem Fachentwurf. Wie Komponenten bzw. Services organisiert (orchestriert) werden, kann nicht restlos im Fachentwurf bzw. im Systementwurf geklärt werden.

Eine große Herausforderung ist die systematische Beschreibung von Komponenten und Services für den Markt und deren Verfügbarkeit. Für die systematische Beschreibung kann auf das sogenannte Stücklistenprinzip zurückgegriffen werden. Für alle Einzelteile ist festzulegen (Spezifikation) wie Input und Output konkret auszusehen haben. Das Zusammenfügen von Services kommt einer „Montage“ (assembling) gleich. Ein Service ist demnach über eine Stückliste vollständig beschrieben.

Verfügbare Komponenten werden bereits auf speziellen Online-Marktplätzen angeboten. Für Services gibt es noch keine Marktplätze. Zurzeit etablieren Softwareanbieter proprietäre Services als Geschäftsmodell. Die Potenziale und Auswirkungen, die frei verfügbare Services, sowohl für Hersteller- als auch für Anwenderorganisationen mit sich bringen, sind (noch) nicht zur Gänze erforscht.

2.3.3.5 Stabilisierung und Gebrauch

Das Ergebnis der Konfigurierung ist ein fertiges Anwendungssystem. Dieses wird im Rahmen der Stabilisierung der Anwenderorganisation näher gebracht. Die Übergabe des Anwendungssystems in die Anwenderorganisation ist sehr gut vorzubereiten. Der Fokus liegt nun allerdings nicht mehr auf dem entwicklungsrelevanten Fachwissen, sondern auf den rekonstruierten aufbau- und ablauforganisatorischen Strukturen und dem Wissen der Anwender, die das System benutzen werden. Alle Ressourcenkategorien aus der Voruntersuchung, wie z.B. „menschliches Wissen“, „Organisationsstrukturen“ (z.B. Prozesskomponenten) und weitere „physische Dinge“ (z.B. Hardware, Mitarbeiterqualifikation, Betriebsstoffe, Maschinen, Arbeitsmittel) kommen nun (wieder) ins Spiel. Deren systematische Anordnung in der Bedingungsmatrix hilft, die Validierung des fertigen Systems zu überblicken. Sie bietet aber auch Struktur für die Einführung des Systems und die Steuerung von weiterem Anpassungsbedarf (*Customizing*). Die Schulungsbedürfnisse von Anwendern lassen sich mit ihrer Hilfe leichter feststellen und zuordnen. Hier rücken die ergonomische *Effizienz*, die *Effektivität* des Gesamtsystems sowie die Zufriedenstellung der Anwender und damit auch eine ausreichende Könnensunterstützung in den Vordergrund.

Im Rahmen der Stabilisierung ist (wieder) der gesamte Anwendungsbereich mit sämtlichen Faktoren (z.B. Betriebsmittel, Material), die zur Aufgabenerfüllung genutzt werden, Betrachtungsgegenstand. Neben dem fertigen Softwaresystem aus der Konfigurierung sind die Ergebnisse des Fachentwurfs notwendiger Bestandteil für den Aufbau des gebrauchstauglichen Systems. Für Standardsoftware ist dies das Pflichtenheft. Im Zuge der Stabilisierung ist es auch möglich, dass wiederholt die Aussagensammlung und die Aussagenklassifikation punktuell erweitert werden müssen. Es handelt sich um Iterationen, mit denen die „Schachbrettmetapher“ (vgl. Abschnitt 2.3.3.1, S. 41ff) nunmehr Gestalt bekommt. Nach Ortner (2005, S. 134) ist ein mehrmaliger Wechsel zwischen Aufgaben der Stabilisierung und notwendiger Wissensbeschaffung (Wissensrekonstruktion) als Aufgabe von Voruntersuchung und Fachentwurf eher die Regel als die Ausnahme.

Als Teilaufgaben der Stabilisierung gelten die Organisationsmodellierung, die *Simulation* von Prozessen zu deren Optimierung sowie die Benutzungsplanung und -steuerung. Die Organisationsmodellierung und die *Simulation* stehen im Kern der Einführung von Standardsoftware (Lehmann, 1999, S. 268-272; Ortner, 2005, S. 137-138). Mittels *Customizing* kann die Anwendung noch in einem bestimmten Umfang an das Anwenderunternehmen angepasst werden. Zu den Tätigkeitsbereichen der Stabilisierung gehören zudem das Testen am Betriebsort, die Identifikation und Planung der zu erbringenden Dienstleistungen im Rahmen der Einführung, wie z.B. Systemschulungen genereller Natur und auch das Einüben des Gebrauchs der neuen bzw. veränderten Anwendungslösung und die Einführung des Systems. Letztere sind für eine erfolgreiche Nutzung des Systems ausschlaggebend.

Wesentlich in diesem Vorgangstyp ist der Übergang der Aufgaben und der Verantwortung von der Herstellerseite zur Anwenderseite. Damit sind besondere Herausforderungen verbunden. Aspekte des Usability Engineering können hier stark zum Tragen kommen, sei es bei notwendigen Tests und deren Bewertung oder bei der endgültigen Abnahme und Validierung des Systems. Im Gegensatz zu den Anfangsphasen der Entwicklung ist in dieser Phase das Thema der Überprüfung der Gebrauchstauglichkeit z.B. mit Usability Tests bereits etabliert. Wie breit und wie tief diese Überprüfungen in der Praxis angelegt sind, kann jedoch nicht ohne weiteres festgestellt werden. Ein möglicher Maßstab wäre eine Überprüfung nach DIN EN ISO 13407 (1999). Einer solchen Überprüfung kann sich jede Organisation unterziehen. In der Regel wird dies allerdings nur dann passieren, wenn eine derartige Überprüfung in den Vertragsunterlagen auch von Seiten der Anwender verlangt wird. Mit Tests kann jedoch in der Regel nicht die Interaktion im realen Umfeld nachvollzogen werden.

Über den Vorgangstyp Gebrauch schließt sich der Lebenszyklus eines Anwendungssystems. Dieser Vorgangstyp (Ortner, 2005, S. 151-182) ist in das Informationsmanagement der Organisation eingebettet. Ortner räumt ein, dass es in bestimmten Projektkonstellationen sinnvoll ist, einen Vorgangstyp Einführung zwischen den Vorgangstypen Stabilisierung und Gebrauch einzuschieben. Diesem Hinweis wird in der Erweiterung des Multipfad-Vorgehensmodells gefolgt (vgl. Abschnitt 3.5.3, S. 160). In diesem Rahmen sind dann auch spezifische Herausforderungen der Einführung, wie z.B. Umgang mit Altsystemen, Parallelbetrieb oder Ablösung des Systems zu einem bestimmten Stichtag, Benutzungsplanung etc. aufzugreifen und interdisziplinär zu diskutieren. Im Mittelpunkt stehen hier aber wieder die Anwender.

In der Regel werden viele Probleme oder Mängel erst beim Gebrauch des Anwendungssystems bemerkt, da hier die *Interaktion* zwischen Anwender und System in allen Prozessen, in realem Kontext und unter den gewohnten Arbeitsbedingungen stattfindet. Der systematischen Betrachtung dieser Interaktion kommt dabei insofern eine wichtige Rolle zu, als hier (noch) nicht realisierte Optimierungspotenziale in den Arbeitsabläufen identifiziert werden können, die als Anregung für künftige Systemverbesserungen, folgende Releases oder auch die Entwicklung zusätzlicher Module und Services dienen.

Weiters fallen die Pflege und Wartung des Systems, sowie die Aufrechterhaltung des Betriebs an sich unter diesen Vorgangstyp. Ob hier ein vollständiges oder teilweises Outsourcing in Frage kommt und welches die relevanten Aspekte zur Beurteilung einer solchen Frage sind, wird hier nicht diskutiert. Für die Erörterung dieser Fragestellungen sei auf einschlägige Fachliteratur verwiesen (z.B. Balzert, 2000; Sommerville, 2001; April & Abran, 2008). Auch für die Erörterung des mit dem Gebrauch eines Systems verbundenen Informationsmanagement sei auf die einschlägige Fachliteratur verwiesen (z.B. Heinrich, 2002; Scheer, 2002; Ferstl & Sinz, 2006). Perspektiven dazu, insbesondere im Zusammenhang mit der sprachkritischen Entwicklung und die Organisationslehre betreffend, erörtert Ortner (2005, S.153-181). Beachtenswerte Aspekte der Mensch-Computer-Interaktion in diesem Zusammenhang finden sich in Abschnitt 3.4.4 (S. 154ff).

2.3.4 Methoden im Multipfad-Vorgehensmodell

2.3.4.1 Taxonomie und Aufbau von Methoden

Nach der Definition im Duden 5 (2007, S. 497) ist unter einer *Methode* ein „auf einem Regelsystem aufbauendes Verfahren, das zur Erlangung von (wissenschaftlichen) Erkenntnissen oder praktischen Ergebnissen dient“, zu verstehen. Heinrich et al. (2004, S. 427) definieren aus wissenschaftstheoretischer Sicht eine Methode als eine

auf einem System von Regeln aufbauende, intersubjektiv nachvollziehbare Handlungsvorschrift zur Problemlösung. Der Wortstamm „Methode“ kommt aus dem Griechischen und kann als „Der Weg zu etwas hin“ oder nach Ableitung aus dem Lateinischen auch „ein nach Mittel und Zweck planmäßiges (= methodisches) Verfahren, das zu technischer Fertigkeit bei der Lösung theoretischer und praktischer Aufgaben führt“, verstanden werden (Mittelstraß, 2004a, S. 876). Demnach ist eine Methode Mittel zum Zweck. Die Entwicklung oder Zusammenführung von Methoden stellt natürlich die Methode als Zweck in den Mittelpunkt. Hierzu sind Mittel einzusetzen, eben wieder Methoden.

Der gemeinsame Nenner der Definitionen von Methoden ist das Vorgehen und „etwas“ in Verbindung damit. Ortner (2005, S. 230) bringt in der sprachbasierten Informatik Sprachen bzw. Sprachartefakte als „etwas“ ins Spiel. Sprachen bzw. Sprachartefakte zusammen mit Vorgehensweisen für den Spracheinsatz (z.B. die Beschreibung eines Handlungs- und Operationszusammenhangs zur Ermittlung einer Kennzahl wie dem „Return on Investment“) bilden Methoden im weiteren Sinn. Es wird damit die Metaebene des Methodenverständnisses integriert, die Heinrich et al. (2004, S. 429) als Methodenmodell beschreiben. Eine Methode kann demnach universell als eine Sprache in Verbindung mit einer Vorgehensweise für den Spracheinsatz definiert werden (Abb. 14). Diese Auffassung bildet das hier verwendete, grundlegende Methodenverständnis. Für eine Beschreibung aus statischer Sicht können wir auf das Aktivitäts-Metamodell zurückgreifen. Methoden sind demnach Bündel von Aktivitäts- und Resultatstypen. Ergebnisse einer Methodenanwendung werden mittels einer Sprache als Resultatstypen (z.B. Use Cases, BPMN-Diagramme, Aussagensammlung) dargestellt und unterliegen vorgegebenen Regeln und Standards. Die Vorgehensweise selbst wird in Form von Aktivitätstypen zum Ausdruck gebracht.

Durch die Explikation der Sprache als Bestandteil einer Methode in der sprachbasierten Informatik ist es ein Leichtes, mehrere Methoden zu Methodenbündeln und diese im Sinne einer Konstruktionslehre als *Methodologie* in ein übergeordnetes System einzubinden.

Das systematische Zusammenfügen bzw. Aneinanderreihen von verschiedenen Methoden kann als Prozessmodell bzw. Vorgehensmodell verstanden werden. Der Methodeneinsatz gliedert sich in der Regel jedoch nicht nach Entwicklungsphasen, sondern ist geprägt durch die Herausforderungen im Anwendungsbereich. Dadurch können sich (Verständnis-)Lücken zwischen Vorgehensmodell und Methodeneinsatz ergeben. Eine anwendungsorientierte Beschreibung von Methoden soll helfen, solche Lücken zu überbrücken. Dies ist nur dann möglich, wenn nicht systematische Fehler der Grund für diese Lücken sind. In einem solchen Fall könnte die Forderung der Abbildung eines systematischen Übergangs von der Anforderungs- und Aufgabenanaly-

se zum Systementwurf im Vorgehensmodell nicht erfüllt werden. Für die sprachbasierte Informatik gibt es diese Lücke nicht, sie ist eben durch die systematische und grundlegende Einbindung von Sprache nicht möglich.

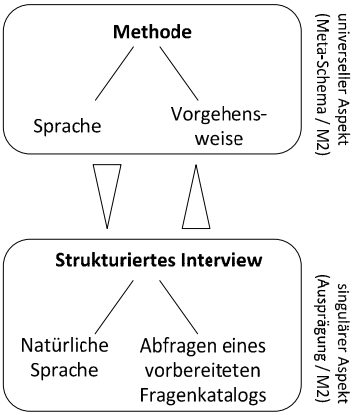

Abb. 14: Universeller und singulärer Aspekt einer Methode auf Metaebene

2.3.4.2 Methodenkategorien in der sprachbasierten Entwicklungsarbeit

Zur Unterstützung der Erhebungen im Rahmen der sprachkritischen Entwicklungsarbeit werden unterschiedliche Methodenkategorien eingesetzt. Eine Klassifizierung von beispielhaften Erhebungsmethoden in primär dialogische, hermeneutische und praktische Methoden und deren Zusammenhang mit der Art der Wissenserhebung (empraktisch und epipraktisch) zeigt Abb. 15 (S. 55; Ortner, 2005, S. 62). Die skizzierten Methodenklassen sind oft nicht klar abzugrenzen und überschneiden einander. Deshalb ist eine geeignete Kombination der Methodenkategorien für die Unterstützung der Erhebungsarbeiten sinnvoll und notwendig um den Anwendungsbereich zu verstehen, darüber relevante Aussagen zu sammeln, Begriffe eindeutig klären zu können und die Abläufe zu gestalten. Eine konkrete Auswahl von Methoden erscheint erst im Kontext eines Projekts sinnvoll.

Nach den Erhebungen bedarf es einer sukzessiven Bearbeitung und Verfeinerung der Erhebungsergebnisse, damit diese im Rahmen der Implementierung in eine maschinenverstehbare Sprache gebracht werden können. Mit dieser Verfeinerung der Ergebnisse wird die Unternehmensorganisation mittels geeigneter Modellierungsmethoden in Modelle gegossen. Durch den ganzheitlichen Ansatz der sprachbasierten Informatik sind die Methoden nicht begrenzt auf die Modellierung von Systemen, sondern es sind Methoden der Unternehmensmodellierung relevant, die mit jenen der Anwendungssystementwicklung korrespondieren. Für eine grundsätzliche Typi-

sierung und ausführliche Charakterisierung von Erhebungsmethoden, z.B. als dialogische, hermeneutische und praktische Erhebungsmethoden, sei auf die einschlägige Literatur verwiesen (z.B. Wedekind, 1976; Schienmann, 1997 u. 2002; Lehmann, 1999; Wandmacher, 2002; Ortner, 2005).

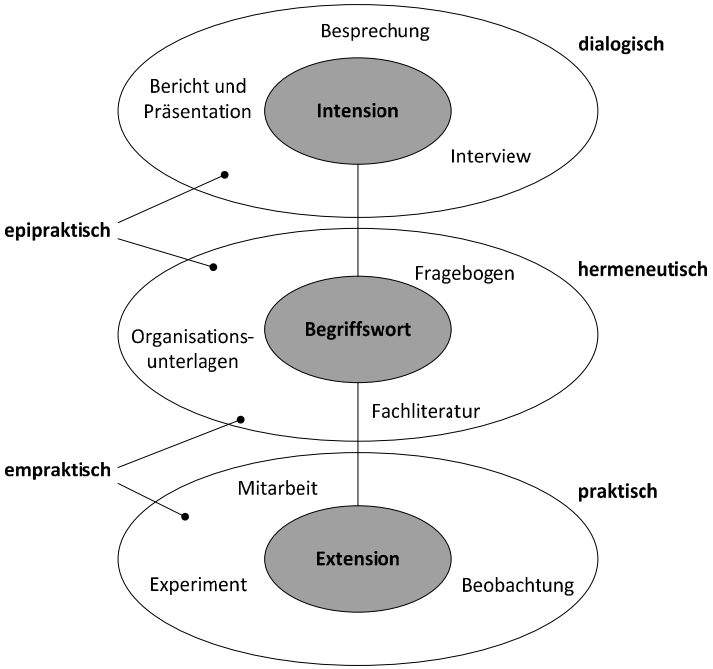

Abb. 15: Erhebungsmethoden zur Rekonstruktion von Wissen

Für diese Unternehmensmodellierung steht uns mittlerweile ein umfassendes Instrumentarium zur Verfügung (Abb. 16, S. 56). Es handelt sich um Sprachen und Vorgehensweisen, die von der OMG in Verbindung mit der UML erarbeitet wurden. Das *Verfügungswissen* ist das Wissen über Mittel, das *Orientierungswissen* ist das Wissen über Zwecke. Beide Wissenskategorien werden unter dem (expliziten) Weltwissen subsumiert, das in seiner Relevanz für ein Unternehmen beispielsweise in Form einer Unternehmens-Enzyklopädie effizient verwaltet werden könnte.

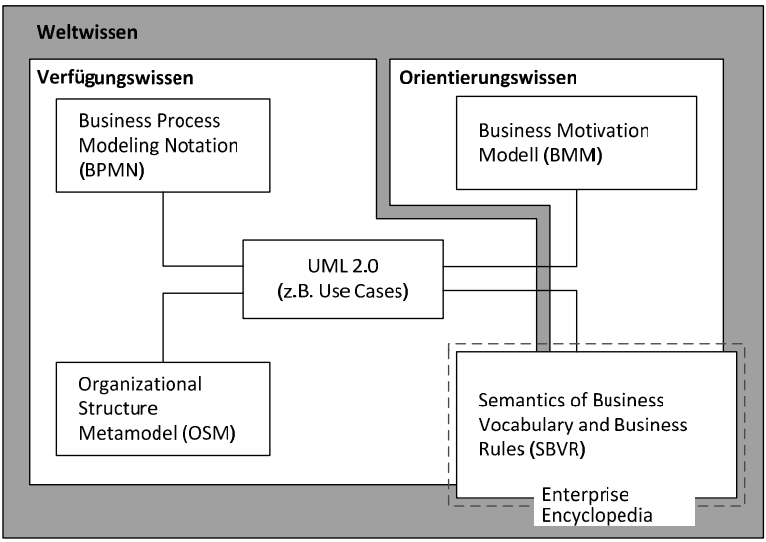

Abb. 16: Instrumentarium der Unternehmensmodellierung

Aus dem Sprachumfang der UML bilden z.B. Use Case Diagramme ein geeignetes Instrument für die statische Unternehmensmodellierung. Andere Teilsprachen der UML befinden sich dagegen überwiegend für den objektorientierten Softwareentwurf und inzwischen auch für die serviceorientierte Softwareentwicklung im Einsatz (Ortner, Eller & Elzenheimer, 2008). In Unternehmensmodellen werden aktuelles Verfügungswissen und Orientierungswissen, repräsentiert in den Aussagen einer Aussagensammlung, vereint.

Die wissensbasierte Zuordnung von Modellen erleichtert den Übergang zu Methoden zur Stützung der Nutzer im Gebrauch eines Anwendungssystems. Da es bei der Stützung der Nutzer um Bildung geht, ist die breite Palette von einsetzbaren Lehr- und Lernmethoden in diesem Kontext zu nennen. Diese Methoden haben den Zweck, die in den vorangehenden Entwicklungsphasen erarbeiteten und schließlich implementierten Modelle dem Benutzer näher zu bringen. Zusätzlich zu Lehr- und Lernmethoden im engeren Sinn, können Methoden und Systematiken des Wissensmanagements Anwendung finden (u.a. Foster, 1982; Nonaka & Takeuchi, 1995, S. 73; Fink, 2000, S. 24-25 u. S. 71; Zelger, 2000; Zegler & Buber, 2000; Ortner, 2005, S. 166f bzw. 187ff; Heinemann, 2006, S. 134-137; Zelger, Raich & Schober, 2008).

2.3.4.3 Methodenübersicht nach Phasen

Im sprachhandlungsorientierten Ansatz des Multipfad-Vorgehensmodells ist eine Methode definiert als eine Vorgehensweise in Verbindung mit einer Sprache. Alle verwendeten Sprachen sind durch eine Grammatik (Syntax) und eine *Terminologie* (Semantik) definiert. Jede Phase im Multipfad-Vorgehensmodell ist gekennzeichnet durch die empfohlenen, hinterlegten Vorgehensweisen und die damit verbundenen Sprachen. Durch die Ausdrucksmächtigkeit der jeweils eingesetzten Sprachen werden die Ergebnisse der Phase in gewisser Weise vordefiniert. In Tab. 3 (S. 58) wurde eine Zuordnung ausgewählter Methoden zu Vorgangstypen bzw. Phasen vorgenommen.

Die Zuordnung ist teilweise redundant, da Methoden nahezu immer phasenübergreifend eingesetzt werden. In der Methodenanwendung können immer wieder Iterationen und Rückschritte vorkommen, aber auch das Überspringen von Phasen ist möglich (Schachbrettmetapher). Auch die Methoden selbst sind oft nicht klar abzugrenzen und können einander sogar ganz oder zumindest teilweise umschließen. So umschießt z.B. die sprachkritische Rekonstruktion die *Begriffsklärung* und die Sammlung von Aussagen und auch Erhebungsmethoden wie z.B. Interviews oder die teilnehmende Beobachtung. Methoden können also ihrerseits wieder Methodenbündel sein, im Fachjargon der Informatik würde man von Methodenkomponenten sprechen. Die Zuordnung erhebt auch keinen Anspruch auf Vollständigkeit.

Die Phasen der Voranalyse und des Fachentwurfs bilden im Multipfad-Vorgehensmodell die sogenannten „frühen“ Entwicklungsphasen. In diesen Phasen sind die Einflussmöglichkeiten für die Systemgestaltung am besten gegeben. Im Hinblick auf die systematische Integration des Usability Engineering auf Methodenebene liegt der Schwerpunkt auf diesen frühen Phasen der Entwicklungsarbeit. Es kommen überwiegend Erhebungsmethoden, Modellierungsmethoden und Dokumentationsmethoden zum Einsatz. Zielsetzung des Methodeneinsatzes ist es, die relevanten Sachverhalte im Anwendungsbereich möglich präzise zu erfassen und (sprachlich) darzustellen. Die Herausforderung dabei liegt in der Gratwanderung zwischen erforderlicher *Abstraktion* und notwendiger Detailtreue. Ein gezielter Einsatz von Methoden des Wissensmanagements bereits in diesen frühen Phasen kann proaktiv helfen, eine Verbesserung der Stützung der Nutzer zu gewährleisten.

Vorgangstyp/Phase	Methoden
Voruntersuchung	• Interview (z.B. Wedekind, 1976; Schienmann, 2002) • Dokumentenanalyse (z.B. Lehner et al., 1991; Lehmann, 1999) • Sprachkritische Rekonstruktion (z.B. Ortner, 1993 u. 2005; Schienmann, 1997; Lehmann, 1999) • Aussagensammlung (z.B. Apel, 1976; Ortner, 2005) • Begriffsklärung (z.B. Lorenzen, 1985; Ortner, 1997 u. 2005; Schienmann, 1997; Lehmann 1999) • Bedingungsmatrix (z.B. Franke, 1975; Wedekind & Ortner, 1980; Ortner, 2005)
Fachentwurf	• Teilnehmende Beobachtung (z.B. Lüders, 2006) • Dokumentenanalyse (s.o.) • Sprachkritische Rekonstruktion (s.o.) • Aussagensammlung (s.o.) • Begriffsklärung (s.o.) • Repository (z.B. Ortner, 2005) • Normierung und Klassifizierung von Aussagen (z.B. Ortner, 1997; Schienmann, 2002)
Systementwurf	• Klassifizierung von Aussagen (z.B. Schienmann, 2002; Ortner, 2008a) • UML-Sprachfamilie (z.B. Fowler & Scott, 2000; Rupp, Queins & Zengler, 2007; OMG, 2008a) • BPMN (z.B. OMG, 2008b) • Diagrammübersetzung (z.B. in BPEL)
Implementierung	• Programmiertechniken und -sprachen, Programmdokumentation
Konfigurierung	• Katalogauswahl • Orchestrierung
Stabilisierung	• Customizing • Simulation (z.B. Wedekind, Görz, Kötter & Inhetveen, 1998) • Schulungsmethoden
Gebrauch	• Methoden des Informationsmanagements und Wissensmanagements

Tab. 3: Methoden im Multipfad-Vorgehensmodell

2.4 Usability und Akzeptanz in der Anwendungsentwicklung

2.4.1 Einordnung und Begriffsverständnis

Human Computer Interaction (HCI) entwickelte sich aus den Disziplinen der Ergonomie (als Teilgebiet der Informatik) und der *kognitiven Psychologie*. HCI geht weit über das Usability Engineering hinaus und umfasst den Menschen, Hardware, Software und zusätzlich Kontextfaktoren wie Aufgaben und Organisationsstrukturen sowie die Interaktion der Faktoren. Die von der ACM SIGCHI (Hewett et al., 2008) veröffentlichte Arbeitsdefinition ist entsprechend breit angelegt:

> Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

Fragen der Human Computer Interaction hängen sehr eng mit anderen arbeits- und organisationspsychologischen Themen zusammen. Bearbeitete Themenfelder wie z.B. Avatare, *anthropomorphe Agenten*, adaptive Systeme, ubiquitäre Computerleistungen oder Spracheingaben und –ausgaben sind meist technisch orientiert. Andere Themen sind wiederum auf bestimmte Anwendungen ausgerichtet wie z.B. CSCW oder E-Learning. Ein weiteres Teilgebiet der HCI befasst sich mit Vorgehen und Methoden bei der Entwicklung von interaktiven Systemen – dieses Teilgebiet wird Usability Engineering genannt. Das Usability Engineering beschäftigt sich mit Themen wie z.B. Contextual Design, Rapid Prototyping, Usability Testing (Wandke, 2007, S. 203).

Im deutschsprachigen Raum sieht sich das Usability Engineering am ehesten in den Arbeitswissenschaften verwurzelt. Es geht dabei traditionell um die Bedienung von Maschinen und um die Gestaltung von Arbeitsplätzen (Hassenzahl & Hofvenschiöld, 2003, S. 135). In der IT-Branche werden darunter nach wie vor primär jene Aktivitäten und damit verbundene Werkzeuge subsumiert, die im Software-Entwicklungsprozess dazu dienen, die Usability, d.h. die Gebrauchstauglichkeit von Software-Produkten, zu verbessern sowie deren Reifegrad zu messen. Usability Engineering soll als Konzept verstanden werden, die Entwicklung interaktiver Systeme mit angemessenen Methoden und Vorgehensweisen zu unterstützen um Anwendbarkeit und Akzeptanz im Nutzungskontext der Systeme zu gewährleisten. Die Aktivitäten sollen dabei nicht auf die Entwicklung von Benutzeroberflächen (= User Interface Engineering) und auch nicht auf Test- und Messmethoden eingeschränkt bleiben. In diesem Sinne gilt es nun, den Begriff „Usability“ zu klären.

Das Verständnis von Usability in der Literatur ist durchaus nicht einheitlich (vgl. z.B. Nielsen, 1993; Mayhew, 1999a; Balzert, 2000; Manhartsberger & Musil, 2001; Rosson & Carroll, 2002; DIN EN ISO 9241-11, 2006; DATech, 2008). Übersetzt mit Gebrauchstauglichkeit ist ein unmittelbarer Bezug zu allen Objekten (Produkten oder Leistungen) gegeben, die für eine Benutzung durch den Menschen vorgesehen sind. Eine klare und handhabbare Definition für die Anwendungsentwicklung ist nicht zu finden. Dies rührt u.a. daher, dass sich das Konzept der Usability nicht auf die Anwendungssystementwicklung beschränkt.

Die International Organization for Standardization (ISO) definiert Usability als ein Maß dafür, wie gut ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend (aus arbeitspsychologischer und arbeitsphysiologischer Sicht) zu erreichen (DIN EN ISO 9241-11, 2006). Zahlreiche Autoren der einschlägigen deutschspra-

chigen Literatur haben diese Sichtweise übernommen (z.B. Heinsen & Vogt, 2003; Heinrich et al., 2004). Es gibt aber auch kritische Stimmen zu dieser Definition (z.B. Manhartsberger & Musil, 2001; Hassenzahl, Beu & Burmester, 2001). In der einschlägigen englischsprachigen Literatur (z.B. Rosson & Carrol, 2002; Shneiderman & Plaisant, 2005; Nielsen, 1993 u. 2001) scheint dieses Konzept nicht annähernd den gleichen Stellenwert zu besitzen. Ein Bezug zur Akzeptanzforschung z.B. von Davis (1989) und Davis & Venkatesh (2000) wird von der ISO nicht hergestellt.

Eine Erweiterbarkeit der Eigenschaften (effektiv, effizient, zufriedenstellend) ist nach dem Konzept der ISO nicht vorgesehen. Diese drei Eigenschaften der Zielerreichung werden auch als Maßstab herangezogen, wenn versucht wird, den Reifegrad der Gebrauchstauglichkeit einer Software festzustellen (DATech, 2008). Diese damit zu Tage tretende vordergründige Test- und Messorientierung des ISO-Konzeptes ist nicht bzw. nur implizit dazu geeignet, die Beeinflussung von Usability im Entwicklungsprozess zu systematisieren.

Dass die klassische Begriffsauffassung von Usability (nach DIN EN ISO 9241-110, 2006) zu kurz greift und es einer Ausweitung des traditionellen Begriffs „Usability“ (Gebrauchstauglichkeit) bedarf, scheint erkannt zu sein (Hassenzahl & Hofvenschiöld, 2003, S. 135). Es existieren Ansätze zur Erweiterung bzw. Ergänzung des Begriffs „Usability“ (z.B. von Hassenzahl, Burmester & Beu, 2001; Djajadinigrat, Overbeeke & Wensveen, 2000; Gaver & Martin, 2000; Monk & Frohlich, 1999), doch es gibt keine gemeinsame Position. Das Spektrum an möglichen Interaktionsformen, das für die Zukunft aufgespannt wird, reicht von Spaß und Natürlichkeit der Interaktion mit akteur- und erzählerbasierten Präsentationen über *Multimodalität* bis hin zu Formen des Interagierens, bei dem sich das Publikum aktiv beteiligen kann und somit zum Mitgestalter des Systems wird. Mit dem ubiquitären Computing kommt es zudem zu einer Emotionalisierung der Mensch-Computer-Schnittstelle, was eine neue Herausforderung für das Usability Engineering der Zukunft darstellt.

Weiter gefasst ist das Konzept von Nielsen. Er stellt Usability als Teil der Systemakzeptanz (system acceptability) dar und differenziert diese in eine „social acceptability“ und in eine „practical acceptability“. In der Kategorie der Akzeptanzsicherung aus praktischer Sicht (practical acceptability = Akzeptanz den Systemzweck betreffend) beschreibt er Usability durch Systemeigenschaften wie easy to learn, efficent to use, easy to remember, few errors und subjectively pleasing (Abb. 17; Nielsen, 1993, S. 25). Nielsen vermittelt klar, dass Usability ein wichtiger Teil der Systemakzeptanz ist, nimmt allerdings keinen Bezug zur Akzeptanzforschung von Davis (1989) bzw. Davis & Venkatesh (2000). Wirtschaftliche Fragen (cost), technische Fragen (compatibility), Fragen der *Reliabilität* und andere Fragen stehen für Nielsen klar außerhalb des Konzepts der Usability, werden jedoch in das Konzept der *Akzeptanz* den System-

zweck betreffend eingeschlossen. Auf den Teil der „social acceptability“ geht Nielsen nicht näher ein.

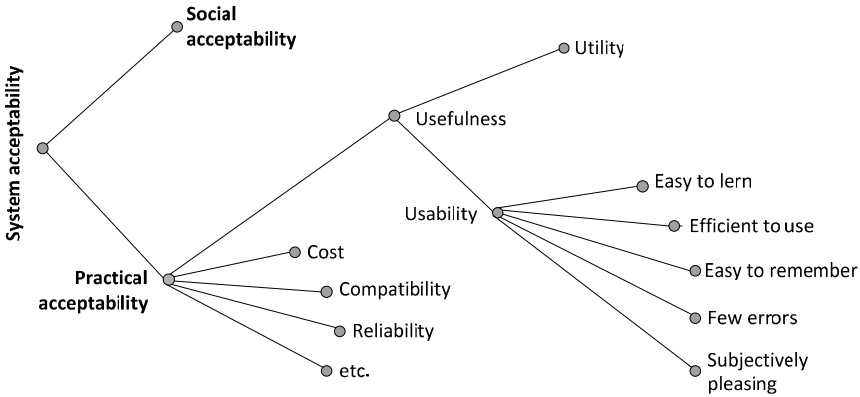

Abb. 17: Usability-Begriffsmodell von Nielsen

Wenn davon ausgegangen wird, dass es bei allen Aktivitäten zur Verbesserung von Usability letztlich immer auch darum geht, die Akzeptanz vorübergehend oder nachhaltig zu verbessern, so kann das Konzept von Nielsen als grundlegend bezeichnet werden. Wir orientieren uns im Folgenden am Konzept von Nielsen, um Usability in Relation zu Akzeptanz zu positionieren. Dies erscheint umso attraktiver, als durch die Einordnung des Konzeptes der Usability unter die Akzeptanz ein Raum eröffnet wird, der eine verhaltensorientierte Betrachtung von Systemeigenschaften ebenso zulässt wie eine nutzungsorientierte Betrachtung, wie sie das ISO-Konzept vorsieht. Die verhaltensorientierte Betrachtung öffnet uns in der Anwendungsentwicklung den Raum zu einer problemgeleiteten, ingenieurmäßigen Herangehensweise an Faktoren zur Beeinflussung von *Akzeptanz* im Sinne eines *Acceptance Engineering* (vgl. Abschnitt 2.4.6, S. 85ff). Ein Acceptance Engineering ist demnach als ein Konzept zu verstehen, welches das Usability Engineering umschließt.

Wird Usability als Teilaspekt von Akzeptanz verstanden, heißt das jedoch nicht, dass eine gebrauchstaugliche Software automatisch zu deren Akzeptanz führt (z.B. geringe Akzeptanz von Betriebssystem-Alternativen zu MS Windows). Usability ist demnach in der Anwendungsentwicklung für Akzeptanz notwendig, aber nicht hinreichend. Usability ist die von den Benutzern und anderen Stakeholdern des Systems mit der Nutzung generell und im Rahmen der Interaktion subjektiv erlebte Qualität mit Qualitätseigenschaften, wie sie z.B. Nielsen aufzählt.[11] Es wurde deutlich gemacht, dass das

11 Für die vorliegende Problemstellung kann die systematische Verbindung von Usability Engineering zum Qualitätsmanagement in den Hintergrund gerückt werden.

Konzept erweiterbar sein muss, sodass z.B. auch Systemeigenschaften, die die Emotionen des Benutzers berühren oder das Verstehen des Systems fördern, in diesem Konzept Raum finden können. Die Erweiterbarkeit des Konzepts soll auch dazu führen, dass künftig Aspekte der Usability im Hinblick auf die Erreichung organisationaler Akzeptanz systematisch adressiert werden können. Die Verhaltensorientierung muss im Raum zwischen Mensch, Organisation und Technik systematisch adressierbar sein.

2.4.2 Dimensionen von Usability

Wenn Usability ein skalierbares Konzept sein soll, dann müssen neben „joy of use“ auch weitere Eigenschaften, wie z.B. das Verstehen dessen, was in den Systemen abgebildet ist und was dort passiert - im Sinne von „easy to understand“ (Inhalte als Wissen = Schemata) - oder Eigenschaften in Verbindung mit der Wahrnehmung von Systeminhalten - im Sinne von „pleasant to perceive“ - explizit verankert werden können. Eine zusätzliche Strukturierung von Usability in die Dimensionen „physiologisch“, „epistemologisch“ und „psychologisch“ soll dies ermöglichen (Abb. 18, S. 63; Eller, 2008a). Jede Dimension wird aus Sicht des Menschen und der Technik betrachtet. Ist die organisationale Akzeptanz Gegenstand der Betrachtung, kommt auch noch diese Perspektive hinzu.

Die physiologische Dimension betrifft in erster Linie die *Benutzungsschnittstelle* und die Gestaltung von Objekten mittels Style Guides, Farben, akustischen Merkmalen, Positionen, Layout von Objekten u.a.m. Aus Sicht der Technik geht es in der physiologischen Dimension um Themen wie Suchmaschinen, interaktive Prozesse, Sprachen aus Entwicklersicht und technische Fragen z.B. zu barrierefreiem Design. Aus Sicht des Menschen ist es die Wahrnehmung eines Anwendungssystems via *Benutzungsschnittstelle* mit allen Sinnen, vor allem mit dem Sehsinn und mit dem Hörsinn, die in dieser Dimension ins Zentrum der Betrachtung rückt. Die physiologische Dimension von Systemeigenschaften lässt sich durch die menschlichen Sinne, aber auch durch Technologien und Technik (z.B. Sensoren), erschließen. Die Gebrauchstauglichkeit eines Systems aus diesem Blickwinkel gesehen, wird unter dem Begriff „Sensual Usability“ subsumiert. In Verbindung mit der systematischen Auseinandersetzung mit Web-Usability wurde vor allem die physiologische Dimension (z.B. Nielsen, 1999; Manhartsberger & Musil, 2001; Norman, 2002; u.v.a.m.) von Usability adressiert. Sichtweisen des Usability Engineering, die sich auf die Gestaltung von Benutzeroberflächen beschränken, betreffen überwiegend die physiologische Dimension.

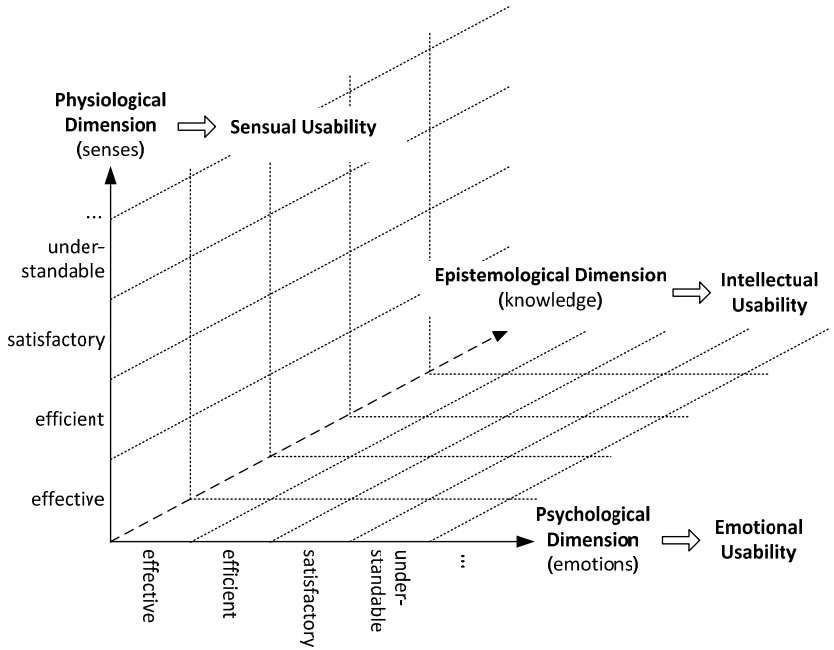

Abb. 18: Dimensionen von Usability

Die psychologische Dimension befasst sich mit Fragen der Gefühlswelt der Benutzer. Es geht dabei um ausgelöste Gefühle bei der Benutzung des Systems (z.B. Freude durch Erfolgserlebnisse oder Ärger durch Misserfolg) sowie um latente oder induzierte Stimmungen wie z.B. gute Laune oder Depression (Steitz, 2008, S. 7). Es sind dabei sowohl Einflüsse der Systemnutzung auf die Gefühlswelt des Benutzers als auch Einflüsse von Gefühlen des Benutzers auf die Systemnutzung von Bedeutung. Diese Aspekte sind jedoch wenig erforscht. Nielsen (1993, S. 143) und Norman (2002, S. 34) stellen in diesem Zusammenhang fest, dass Benutzer dazu neigen, sich selbst die Schuld zu geben, wenn es bei der Benutzung eines Systems zu Fehlersituationen kommt. Dass ein Problem oft auch von Fehlern bei der Entwicklung herrührt, wird von den Benutzern nicht gesehen. Fehlersituationen können aber durchaus Frustration beim Benutzer hervorrufen und zur Ablehnung eines Systems führen (Winand, 2000). Auch das Auftreten von Affekten z.B. in Form von verbaler oder physischer Gewalt gegen den Computer kann als Folge von Fehlersituationen auftreten (Sarodnick & Brau, 2006, S. 15). Eine Reihe von Fehlersituationen beim Gebrauch eines Anwendungssystems, auch im Sinne von mangelhafter Gebrauchstauglichkeit, kann also zu Voreingenommenheit des Benutzers und damit zu einem nachhaltigen Akzeptanz-

verlust führen. Wie umgekehrt die vorhandene Stimmung des Benutzers die Systemnutzung beeinflusst, ist ungleich weniger untersucht.

Die psychologische Dimension von Usability ist u.a. über die Repräsentationsformen impliziten Wissens erschließbar. Dies erfordert Kenntnis und Einsatz von Methoden aus Psychologie und Kommunikationswissenschaften. Alle *Aspekte* dieser Dimension werden unter dem Begriff „Emotional Usability" subsumiert. Die Emotional Usability ist für alle möglichen Unterstützungsgrade des Anwenders relevant, also auch für die Arbeitsabnahmedurch die IT. Der Benutzer soll weder über- noch unterfordert werden. Die Funktions- und Wissensverteilung soll angemessen sein. „Angemessen" bedeutet in diesem Zusammenhang, dass die relativen Fähigkeiten und Grenzen des Menschen im Vergleich zur Technik berücksichtigt werden, wie z.B. hinsichtlich Zuverlässigkeit, Schnelligkeit, Genauigkeit des Benutzers (Sarodnick & Brau, 2006, S. 41). Maßgebende Erkenntnisse zur Beeinflussung und Steuerung der Emotional Usability kommen aus der Arbeitspsychologie und aus der Akzeptanzforschung.

Über die epistemologische Dimension wird das Wissen adressiert, das in den Systemen gespeichert ist. Es geht aber auch um das Aufbereiten und Verstehen dieses Wissens. Fragen wie z.B. ‚Sind die Inhalte aktuell?', ‚Sind die Inhalte gut organisiert?', ‚Sind sie von veraltetem Wissen bereinigt?' oder ‚Ist der Wissensbestand skalierbar?', sind aus Sicht der Technik innerhalb dieser Dimension maßgebend. Aus Sicht des Menschen geht es aus diesem Blickwinkel u.a. um das Wissen jedes Benutzers in Bezug auf die konkrete Systemnutzung. Dies umfasst sowohl Orientierungs- als auch Verrichtungswissen. Fragen zum erforderlichen Qualifikationsniveau kommen hier ebenso vor wie die Frage nach der erforderlichen Hilfestellung für den Benutzer durch das System während der Systemnutzung. Bildung an sich ist also ein zentrales Thema. Die hier diskutierten Aspekte werden unter dem Begriff „Intellectual Usability" subsumiert.

Der epistemologischen Dimension ist Sprache immanent, da Wissen über Sprachen bzw. Sprachartefakte repräsentiert wird. Über diese Dimension ist die unmittelbare Verbindung zur sprachbasierten Informatik gegeben. Es sind nämlich nicht nur Wissensrepräsentationen, sondern auch Informationstransformationen sowie Wissens- und Informationstransfers unter dem Aspekt der Gebrauchstauglichkeit zu sehen.

Beispielsweise kommen im Rahmen der *Begriffsklärung* Fragen der intellectual Usability zum Tragen. Hier gilt es, Verstehbarkeit (lokal, inhaltlich-kognitiv, innen), Anwendbarkeit (innen, global) und auch Zugänglichkeit (in die Tiefe, außen, technisch materiell) zu gewährleisten (vgl. Abb. 22, S. 83; Sauer, 1995; Bartsch, 2001, S. 18-24). Diese Eigenschaften kommen aus der Disziplin der Textgestaltung und Textverarbeitung und können auch auf die Verarbeitung von Zeichen (Semiotik) umgelegt werden. Die Lesbarkeit einer Benutzeroberfläche, als lokale Wahrnehmungs- und Verarbei-

tungseinheit aus technisch materieller Sicht nach außen, ist der physiologischen Dimension zuzuordnen. Geht es um das um sinnentnehmende Lesen und Sehen einschließlich der Zuordnung zu übergreifenden Handlungszielen für den Anwender, steht die epistemologische Dimension im Vordergrund.

Die Dimensionen bilden eine Metaebene für Nutzungseigenschaften und stehen orthogonal zu einer aufgabenorientierten bzw. prozessorientierten Sicht auf die Gebrauchstauglichkeit eines Anwendungssystems. Jede Dimension wird sowohl aus Sicht des Menschen als auch aus Sicht der Technik betrachtet. Die Betrachtung erfolgt im Rahmen der organisationalen *Erfordernisse*. Systemattribute und Systemcharakteristika, aber auch Anwenderattribute und Anwendercharakteristika sind in allen Dimensionen zu thematisieren und deren mögliche Beeinflussung zu diskutieren. Gleiche Systemattribute können in den verschiedenen Dimensionen unterschiedliche Bedeutung haben. Beispielsweise wird „joy of use“ schwerpunktmäßig in der psychologischen Dimension aus Sicht des Menschen zu behandeln sein, ohne aber physiologische und epistemologische Aspekte außer Acht zu lassen. Für jede Dimension können zusätzliche Eigenschaften festgesetzt werden, unabhängig davon, aus welchem Blickwinkel (Mensch, Organisation, Technik) und in welcher Dimension Usability gerade betrachtet wird. Über die Metaebene der Usability-Dimensionen können also jederzeit Attribute wie „joy of use“(= Gefühle), „pleasant to perceive“(= Sinne) und „easy to understand“(= Wissen) in den Rahmen der Entwicklungsarbeit eingeordnet, gewichtet und systematisch erschlossen und auch beeinflusst werden.

Die in der einschlägigen Literatur diskutierten Fragen zu Usability und Akzeptanzforschung finden schwerpunktmäßig in der physiologischen, aber zunehmend auch in der psychologischen Dimension statt, wie z.B. bei Nielsen (1993), Mayhew (1999), Davis & Venkatesh (2000) oder Rosson & Carroll (2002). Wo aber bleibt die Behandlung der epistemologischen Dimension sowohl aus Sicht des Menschen als auch aus Sicht der Technik? Entweder findet die Diskussion über die Gebrauchsfähigkeit des repräsentierten Wissens in der einschlägigen Literatur nicht statt, oder sie wurde auf Nebenschauplätze verschoben und findet dort fragmentiert oder ansatzweise statt, wie beispielsweise im Ontology-Engineering oder in der Semantic-Web Community? Eine Methode, die geeignet erscheint, die Gebrauchstauglichkeit von Wissen systematisch zu berücksichtigen, insbesondere diese zu beeinflussen, ist die sprachkritische Rekonstruktion.

2.4.3 Gestaltungsrahmen für den Usability Engineering Prozess

2.4.3.1 Einleitung und Übersicht

Aus der Vielzahl von Prozessmodellen des Usability Engineering (z.B. Martin, 1991; Beyer & Holtzblatt, 1998; Daimler Benz, 1998; Mayhew, 1999a; Constantine & Lockwood, 1999; Manhartsberger & Musil, 2001; Rosson & Carroll, 2002; Cockburn, 2003; Daimler Chrysler, 2003; Gundelsweiler et al., 2004; Holzinger & Slany, 2006; Sarodnick & Brau, 2006; Agile Alliance, 2008; DSDM, 2008) haben wir für die systematische Integration des Usability Engineering den Gestaltungsrahmen für Usability Engineering Prozesse der DATech ausgewählt. Er wird selbst nicht als Prozessmodell deklariert (DATech, 2008, S. 11). Der Gestaltungsrahmen wurde auf Grundlage von DIN EN ISO 13407 (1999) erstellt. Innerhalb dieses Rahmens ist bereits eine Integration des Usability Engineering in die Anwendungssystementwicklung angedacht und ansatzweise besprochen (DATech, 2008). Der Prozess beschränkt sich nicht auf Teilbereiche der Anwendungssystementwicklung (wie z.B. das User-Interface-Engineering), sondern versteht sich als Konzept der Entwicklung interaktiver Systeme mit besonderem Augenmerk auf die Gestaltung ergonomisch angemessener Arbeitsweisen von Benutzern im Nutzungskontext innerhalb dieser Systeme. Das Konzept ist als Lebenszykluskonzept ausgelegt und unterscheidet in diesem Zyklus die Sichtweisen von Auftraggeber und Auftragnehmer. Mit Hilfe dieses Rahmens soll durch ein Zusammenwirken von Anwender und Hersteller die Entwicklung und zyklische Verbesserung von Systemen ermöglicht werden.

Die für die Entwicklungsarbeit generell geforderte konstruktive und ingenieurmäßige Vorgehensweise gab den endgültigen Ausschlag für die Auswahl des DATech Gestaltungsrahmens zur Integration des Usability Engineering in die Anwendungssystementwicklung. Die Philosophie der Verwendung von Standards kommt aus dem Maschinenbau. Aufgreifen und Einsatz von bereits vorhandenen Standards können der Anwendungssystementwicklung dabei helfen, von der Werkstattfertigung zur Industrialisierung zu reifen.

Diese Grundlegungen sind auch dafür verantwortlich, dass der Gestaltungsrahmen mit Phasen und Aktivitäten eines Systementwicklungsprozesses korrespondiert. Aus Sicht des Usability Engineering wurden im Gestaltungsrahmen der DATech auch Eigenschaften der Prozessmodelle von Mayhew (1999a) und Rosson & Carroll (2002) berücksichtigt. Die Darstellung des Gestaltungsrahmens gliedert sich in Phasen und zugehörige Prozessschritte sowie Querschnittaktivitäten (Abb. 19, S. 67; DATech, 2008, S. 15).

Trotz aller gegebenen Parallelen weist der Gestaltungsrahmen der DATech immer noch ausreichend Spielraum für eine konkrete Ausgestaltung der Entwicklungsarbeit

auf. Die DATech will die angegebenen Vorgangstypen nicht zwingend als Phasen im Sinne eines Vorgehensmodells verstanden wissen. Sie sieht darin eine thematische Ordnung. Diese muss nicht streng an einer gedachten Zeitlinie entlang (z.B. Lebenszyklus) geordnet sein (DATech, 2008, S. 11). Das Konzept ist also anpassbar an spezifische Anforderungen bei der Entwicklung unterschiedlichster Anwendungssystemtypen und folgt dem Lebenszykluskonzept. Erst die Betrachtung und die Wahrnehmung von Einflüssen über den gesamten Lebenszyklus eines Anwendungssystems ermöglicht es, die Potenziale des Usability Engineering voll auszuschöpfen.

Auftraggeberprojekt		Auftragnehmerprojekt			Auftraggeber-projekt
Planung	**Kontext-analyse**	**Projekt-vorbereitung**	**Design und Validierung**	**Implementierung und Test**	**Nutzung und Pflege**
• Impuls für Anwendungs-entwicklung • Projektvor-bereitung • Sensibilisierung für Usability (Gebrauchstaug-lichkeit)	• Kosten-/Nutzen-analyse • Ist-Analyse • Analyse des Nutzungskontextes -Benutzer -Arbeitsaufgaben -Arbeitsumgebung • Spezifikation von Nutzungsanforde-rungen • Lead-User-Interview • Ausschreibung mit Usability als geforderter Produktqualität	• Marketing-unterstützung • Angebot und Vertrag • Planung von Usability-Aktivitäten • Projektspezifische Festlegungen • Rollenverteilung	• Analyse des Nutzungskontextes -Benutzer -Arbeitsaufgaben -Arbeitsumgebung • Aufgabendesign • Interaktionsdesign • Usability-Prototyping -Nutzungskontext -Entwurf -Evaluierung • Oberfächendesign (Informations-darstellung) • Claims Analysis • Validierung der Systeman-forderungen • Benutzerdokumen-tation	• Design-Regelwerke • Unterstützung des Designs und Begleitung der Implementierung • Entwicklungs-begleitende Usability-Tests • Abnahmetest	• Einführungsprojekt • Konfiguration des Produkts • Benutzerschulung • Feedback von Benutzern auswerten • Pflegeprozess • Vorbereitung des nächsten Releases
Benutzerbeteiligung					
Qualitätsmanagement					

Abb. 19: Gestaltungsrahmen für den Usability Engineering Prozess nach DATech

Dies steht im Gegensatz zu Modellen der Softwareentwicklung, die nur einen Teil des Produktlebenszyklus abbilden, nämlich den Entwicklungsprozess im engeren Sinn vom Entwurf über die Implementierung bis zur Übergabe an den Kunden. Diese verkürzte Sichtweise wird spätestens mit der Integration des Usability Engineering hinfällig. In den herausragenden Konzepten in dieser Disziplin (z.B. Mayhew, 1999a; Rosson & Carroll, 2002; Gundelsweiler et al., 2004; DATech, 2008) ist die Lebenszyklusorientierung verankert.

Die erweiterte Betrachtung des Prozesses im Sinne des gesamten Lebenszyklus eines Anwendungssystems, sowohl aus Sicht des Auftraggebers (Anwenderunternehmen) als auch aus Sicht des Auftragnehmers (Herstellerunternehmen), soll andeuten, welche Organisation jeweils die Hauptverantwortung für die darunter liegenden Phasen inne hat. Sie unterstreicht gleichzeitig die Wichtigkeit der Berücksichtigung der ver-

schiedenen Sichten und Intentionen der beteiligten Organisationen. Diese Sicht erfolgt im DATech Gestaltungsrahmen in Anlehnung an das V-Modell XT (Rausch & Broy, 2006), wobei dort diese Sichtweisen wohl skizziert, jedoch methodenspezifisch nicht näher ausmodelliert sind.

Der Gestaltungsrahmen sieht für den Usability Engineering Prozess folgende Phasen bzw. Projektbausteine vor:

- **Planungsphase** – In der Planungsphase wird im Unternehmen für das Thema „Usability" sensibilisiert. Auf Basis der vorliegenden Impulse für die Anwendungssystementwicklung werden erste Projektvorbereitungen getroffen.
- **Kontextanalyse** – Im Rahmen der Kontextanalyse wird auf Basis der Ist-Analyse der Nutzungskontext des Systems (Arbeitsaufgaben, Arbeitsumgebung, Benutzer) analysiert und die *Nutzungsanforderungen* spezifiziert. Eine Kosten-Nutzen-Analyse sowie erste Lead-User-Tests erfolgen bereits vor der Ausschreibung.
- **Projektvorbereitung** – Die Projektvorbereitung umfasst die Vertragsgestaltung mit dem Hersteller, die Verteilung der Rollen im Projekt (auch in der Zusammenarbeit zwischen Auftraggeber- und Auftragnehmerorganisation) sowie sonstige projektspezifische Festlegungen, insbesondere die Planung von notwendigen Usability-Aktivitäten. Inwieweit Marketingaktivitäten zu setzen sind, ist abhängig vom Typ des zu entwickelnden Systems (z.B. für off-the-shelf Produkte).
- **Design- und Validierungsphase** – Die Design- und Validierungsphase umfasst eine nochmalige Auseinandersetzung mit dem Nutzungskontext, diesmal aus Sicht des Auftragnehmers. Mit einem prototyping-orientierten Vorgehen werden die Systemanforderungen des Auftraggebers validiert und vom Aufgabendesign über das Interaktionsdesign bis hin zum Oberflächendesign (im Sinne einer *Claims Analysis* immer mit Fokus auf den Benutzer) schrittweise erstellt und begleitend evaluiert.
- **Implementierungs- und Testphase** – Die Implementierungs- und Testphase sieht die Unterstützung und Begleitung der Implementierung vor. Die Phase ist flankiert mit entwicklungsbegleitenden Usability-Tests bis hin zu einem Abnahmetest. Zugrundeliegende Regelwerke sind bei Bedarf fortzuschreiben.
- **Nutzungs- und Pflegephase** – die Nutzungs- und Pflegephase beginnt bereits bei der Einführung des fertigen Anwendungssystems im Anwenderunternehmen. Damit sind in dieser Phase auch verstärkt Benutzerschulungen

und die Auswertungen des Benutzerfeedback vorgesehen. Die Erkenntnisse fließen in die Vorbereitung des nächsten Release ein. Der Pflegeprozess erstreckt sich über den gesamten Nutzungszeitraum des Systems.

Zwischen den Phasen gibt es enge Rückkopplungen. Dabei werden in fast allen Schritten des Usability-Engineering Evaluationsmaßnahmen durchgeführt. Je nach Ergebnis ist ein Rücksprung in eine frühere Phase möglich (Sarodnick & Brau, 2006).

Als Querschnittsaufgaben sieht das Modell die Benutzerbeteiligung und das Qualitätsmanagement vor. Eine Besonderheit dieses Prozessmodells ist die Einbeziehung von eindeutigen Rollen der Beteiligten. Das heißt in allen Phasen des Modells sind die Rollen und die entsprechenden Aufgaben der Beteiligten eindeutig und ihren jeweiligen fachlichen Kompetenzen entsprechend verteilt und beschrieben.

Nachfolgend werden die Bausteine des DATech-Prozessmodells beschrieben. Jede Phase wird überblicksmäßig angesprochen. Der Schwerpunkt der Beschreibung liegt jedoch in den beiden Anfangsphasen (Auftraggeber-Projekt) und im Übergang zum Auftragnehmer-Projekt, da insbesondere in diesen Anfangsphasen der Entwicklung das Usability Engineering von großer Bedeutung ist (DATech, 2008). Eine frühzeitige Berücksichtigung von Aspekten der Gebrauchstauglichkeit kann in den späteren Phasen, insbesondere während und nach der Einführung, dazu beitragen, erhebliche Kosten zu sparen (Landauer, 1995; Donahue et al., 1999). Eine Beschreibung zu den einzelnen Phasen des Gestaltungsrahmens findet sich in Anhang 5.2 (S. 188ff).

Die Modellphasen bzw. Modellbausteine des von der DATech empfohlenen Gestaltungsrahmens für den Usability Engineering Prozess bilden den gesamten Produktlebenszyklus eines Anwendungssystems ab. Auffallend ist dabei, dass das Projektmanagement in Form von zwei Phasen bzw. Bausteinen in Erscheinung tritt. Es bleibt jedoch unklar, ob diese Phasen dazu dienen, bestimmten Aktivitäten im Rahmen des Usability Engineering besonderes Gewicht zu verleihen (z.B. der Sensibilisierung für Usability). Beispielsweise ist neben der eigentlichen Prozessdurchführung das Maß und die Art und Weise, wie die Anwenderorganisation selbst die Benutzerzentrierung fördert und unterstützt, eine wichtige Facette eines erfolgreichen Usability Engineering. Dies zu betonen, ist Absicht der explizit verankerten Projektmanagementphase zu Beginn der Entwicklung. In keinem der einleitend skizzierten Prozessmodelle erscheint das Projektmanagement dieser Art im Vordergrund. Dem Charakter nach ist das Projektmanagement auch in Verbindung mit dem Usability Engineering als Querschnittfunktion zu betrachten.

Für die Entwicklungsarbeit und auch entwicklungsnahe Themen wie das Usability Engineering stehen eine Reihe von Normen zur Verfügung (z.B. DIN EN ISO-Reihe 9241, DIN EN ISO 13407) und werden laufend weiterentwickelt (z.B. ISO 23973 – wird zukünftig DIN EN ISO 9241-151 heißen oder ISO/TS 16071 – wird zukünftig DIN EN ISO

9241-171 heißen; DATech, 2008). Die Verwendung von Normen verspricht Potenzial für die Verbesserung der ingenieurmäßigen Vorgehensweise und birgt gleichzeitig Integrationspotenzial auf struktureller Ebene, insbesondere im Hinblick auf eine vertragliche Verankerung der Gebrauchstauglichkeit zwischen Auftraggeber und Auftragnehmer.

Im Gestaltungsrahmen für den Usability Engineering Prozess besonders hervorzuheben ist die klare Adressierung der Auftraggeberorganisation und der Herstellerorganisation. Die Entwicklungsarbeiten beginnen und enden auf der Auftraggeberseite. Die Bedeutung der Herstellerseite wird im Modell bereits (visuell) relativiert. Auch wenn die Zuordnung der Verantwortung sich nicht immer an die Grenzen von Phasen hält, so ist doch hervorzuheben, dass im DATech Gestaltungsrahmen die Auftraggeberorganisation deutlich in die Pflicht genommen wird. Dies wird in den Detailausführungen über den gesamten Prozess immer wieder deutlich. Eine klare Positionierung dieser Art kann durchaus als notwendige (wenn auch nicht hinreichende) Voraussetzung für die Verankerung der Benutzerbeteiligung gesehen werden.

Die Anführung der Benutzerbeteiligung als Querschnittfunktion stellt eine Redundanz zu den bei den Modellbausteinen hinterlegten Aktivitäten dar. Die Integration der Benutzer ist eine Forderung, die es permanent über alle Phasen zu realisieren gilt. Es ist anzunehmen, dass durch diese Darstellung die Benutzerbeteiligung über den gesamten Lebenszyklus des Produkts betont werden soll. Eine tatsächliche Beteiligung der Benutzer kann jedoch innerhalb der Anwendung von Methoden erfolgen, damit auf Ebene der Methoden (vgl. Tab. 1, S. 7). Erfolgt diese mit ausreichendem Gewicht, kann auf eine Verankerung der Benutzerbeteiligung als Querschnittfunktion verzichtet werden.

2.4.4 Methodenempfehlungen für den Usability Engineering Prozess

2.4.4.1 Einführung und Methodenübersicht nach Prozessbausteinen

Methoden des Usability Engineering sind ebenso zahlreich vorhanden wie Methoden der Anwendungssystementwicklung. Eine Systematisierung der Methodenbeschreibung speziell für Methoden des Usability Engineering existiert nicht. Auf einen Versuch der allgemeinen Ordnung und Klassifizierung wird hier verzichtet. Für eine Detailbeschreibung von Methoden des Usability Engineering kann der bereits eingeführte und aspektorientierte Ordnungsrahmen herangezogen werden (vgl. Abschnitt 3.5.4.3, S. 177ff). Die Begründungen für eine aspektorientierte Beschreibung sind übertragbar.

Für das Usability Engineering wurden zahlreiche, oft sehr speziell ausgeprägte Methoden entwickelt (z.B. Paper-Mock-up, GOMS-Modell) oder aus anderen Disziplinen

(z.B. der Pädagogik) übernommen und adaptiert. Jede der Methoden hat bestimmte Stärken und Schwächen, und abhängig von der Aufgabenstellung sind auch verschiedene Kombinationen von Methoden sinnvoll (*Triangulation*). Auflistungen mit Kurzbeschreibungen finden sich u.a. in Methodenpools akademischer Einrichtungen (z.B. Universität Köln, 2008; Jacobsen, 2006) und in einschlägiger Literatur (z.B. Mayhew, 1999a; Sarodnick & Brau, 2006; ISO/TR 16982, 2002).

Um die Benutzerfreundlichkeit von Anwendungen zu sichern, stehen dem Usability-Experten nicht nur Usability-Testmethoden zur Verfügung. Es gibt eine Reihe weiterer Methoden (z.B. Kontextanalysen, Claims Analysis, Interviews, teilnehmende Beobachtung, Thinking aloud), die zusätzlich zum Einsatz kommen können (z.B. Jacobsen, 2006). Diese Methoden forcieren bereits in den frühen Entwicklungsphasen eine Beteiligung der Benutzer und damit auch die Beeinflussung der Gebrauchstauglichkeit eines Anwendungssystems. Die Beteiligung der Benutzer lässt sich hinsichtlich Verbesserung der Gebrauchstauglichkeit und Sicherung bzw. Verbesserung der Akzeptanz von Systemen bereits vor ersten Tests nicht abgrenzen. Die genannten Methoden haben ihren Ursprung u.a. in der Systementwicklung und in den Kommunikationswissenschaften (z.B. Kontextanalysen, Prototyping, Interviews). Die Kommunikation zwischen den Beteiligten in einem Entwicklungsprozess bezeichnen Seffah, Desmarais & Metzker (2005) als eine der größten zu bewältigenden Hürden, deren Bewältigung vom Usability Engineering zu unterstützen sei.

Usability-Testmethoden sind im Vergleich von Methoden zur systematischen Beeinflussung von Usability sowohl in der Wissenschaft als auch in der Praxis am weitesten ausgereift und etabliert (z. B. Sarodnick & Brau, 2006). In Institutionen wie CURE, MCI und USECON (Manhartsberger) herrscht die wissenschaftliche Auseinandersetzung mit der Test-Thematik immer noch vor. Die kommerzielle Umsetzung auf breiter Ebene steckt dennoch in den Kinderschuhen, wird aber von den genannten Institutionen in zahlreichen Projekten für KMU (z.B. INTERREG-Projekte der Bodenseeregion) forciert. Testmethoden des Usability Engineering dienen zur Bewertung erreichbarer Sachverhalte und Qualitätskriterien und finden auf Entwicklungsergebnissen Anwendung. Methoden zur Beeinflussung der Gebrauchstauglichkeit, die diese im Rahmen der Entwicklungsarbeit fördern können, stehen jedoch im Mittelpunkt des Integrationsvorhabens.

Für den DATech Gestaltungsrahmen gibt es nur mittelbare Hinterlegungen von einzusetzenden Methoden in Bezug zu den genannten Aufgaben (vgl. Abb. 19, S. 67), wobei anzumerken ist, dass Aufgaben und Methoden oft gleich benannt sind. Der Problematik der Methodenauswahl und Methodenbeschreibung, die sich aus den Ansätzen von DIN EN ISO 9241-110 (2006) und DIN EN ISO 13407 (1999) ergibt, wurde z.B. mit Methodenbeschreibungen im ISO/TR 16982 (2002) begegnet. Sie sollten als

Hilfestellung zu Entscheidungen über den Einsatz von Usability-Methoden und Verfahren für den Projektmanager dienen. Die Ausführungen sind jedoch (noch) nicht so weit ausgereift, dass sie in einen eigenen Standard oder z.B. als Ergänzung zum Gestaltungsrahmen übernommen werden könnten.

Exemplarisch wurde in Tab. 4 (S. 24) eine Zuordnung von Methoden zu einzelnen Phasen bzw. Projektbausteinen des DATech Gestaltungsrahmens vorgenommen. Die Zuordnung bildet keine zeitliche Reihenfolge des Methodeneinsatzes ab. Eine Mehrfachzuordnung soll den phasenübergreifenden Einsatz der Methoden abbilden. Die Auswahl beschränkt sich nicht auf jene Methoden, die bereits von der DATech begleitend zum Gestaltungsrahmen beschrieben bzw. erwähnt wurden bzw. im ISO/TR 16982 (2002) beschrieben sind.

Die Benutzerbeteiligung in den Methoden erfolgt jeweils mit unterschiedlicher Intensität und wird als den Methoden immanent erachtet. Auf die Benutzerbeteiligung ist größter Wert zu legen, da eine unzureichende Beteiligung der Benutzer den höchsten Risikofaktor in Software-Entwicklungsprojekten darstellt (DATech, 2008, S. 12 mit Verweis auf CHAOS Reports der Standish Group 1995, 1999, 2001 u. 2006).

Das Methodenspektrum des Usability Engineering ist sehr breit (Holzinger, 2005). Ausgewählte Methoden für den Einsatz in frühen Entwicklungsphasen (vgl. Tab. 4, S. 73) werden nachfolgend kurz charakterisiert. Wenn möglich werden synonym bezeichnete Methoden erwähnt und es wird auf Methodenvarianten hingewiesen, eine vollständige Klärung der Begriffe zu den einzelnen Methoden findet an dieser Stelle nicht statt.

Prozessbaustein/Phase	Methoden
Planung	Methoden der Kommunikation zur Sensibilisierung für Usability
Kontextanalyse	• Kosten-/Nutzenanalyse (z.B. Sen, 2000; Hirschmeier, 2005) • Lead-User-Interview (z.B. DATech, 2008) • Teilnehmende Beobachtung (z.B. DATech, 2008) • Ethnografische Beobachtung (z.B. Sommerville & Kotonya, 1998) • Dokumentenanalyse (z.B. Lehner et al., 1991; DATech, 2008) • Thinking aloud (z.B. Holzinger, 2004) • Personas (z.B. usability.gov, 2008) • Claims Analysis (z.B. DATech, 2008; Usability first, 2008)
Projektvorbereitung	Keine Methoden des Usability Engineering zugeordnet
Design & Validierung	• Teilnehmende Beobachtung (s.o.) • Aufgabenanalyse (z.B. DATech, 2008) • Ethnografische Beobachtung (s.o.) • Dokumentenanalyse (s.o.) • Szenariotechniken (z.B. Mayhew, 1999a; Rosson & Carroll, 2002) • Use Cases (z.B. Fowler & Scott, 2000; Interface consult, 2008) • Walkthrough-Verfahren (z.B. Nielsen, 1993) • Fokusgruppen (z.B. usability.gov, 2008) • Usability Prototyping (Pomberger et al., 1991; Sarodnick & Brau, 2006, S. 158-159; DATech, 2008, S. 28-30) • Style Guidelines (z.B. Nielsen, 1993; Mayhew, 1999a; Sarodnick & Brau, 2006) • Claims Analysis (z.B. Seffah et al., 2005; DATech, 2008; Usability first, 2008)
Implementierung & Test	• Style Guides (s.o.) • Heuristiken (z.B. Nielsen, 1993) • Expertenleitfäden (z.B. Sarodnick & Brau, 2006; Mayhew, 1999a) • GOMS-Modell (z.B. Sarodnick & Brau, 2006) • Usability Prototyping (s.o.)
Nutzung & Pflege	• Nutzungstest • Benutzerbefragung (z.B. DATech, 2008) • Feedback von Benutzern auswerten • Erfassen von Change Requests

Tab. 4: Methoden im Gestaltungsrahmen für den Usability Engineering Prozess

2.4.4.2 Kontextanalyse

Bei der Aufgabe der Erarbeitung eines Nutzungskonzeptes spielt die Kontextanalyse mit den untergeordneten Methoden wie Interviews, teilnehmende Beobachtung, Szenariotechnik eine wesentliche Rolle. Diese Methoden können durch das Thinking aloud (Holzinger, 2004) und weitere Kommunikationstechniken ergänzt werden. Auch die Erstellung von *Personas* fällt in diesen Rahmen. Die Kontextanalyse kann als Methodenkomposition verstanden werden.

Szenariotechnik

Szenariotechniken werden sowohl von Seiten des Software Engineering (Jarke, 1999; DATech, 2008 - hier als Nutzungsszenarien bezeichnet) als auch im Rahmen der HCI (Rosson & Carroll, 1992; Carroll, 2000) als Methode zur Erfassung von Systemanforderungen forciert. Auch wenn es keine konsistente bzw. eindeutige Definition für ein *Szenario* gibt, eignen sich Szenarios dazu, die objektorientierte Analyse mit dem objektorientierten Design zu verbinden. Die Vorstellungen, wie Szenarien eingesetzt werden sollten, gehen dabei sehr wohl auseinander.

Ein *Szenario* im Allgemeinen ist eine episodische Beschreibung von Aufgaben und Tätigkeiten in ihrem Kontext. Szenarien werden als Hilfsmittel eingesetzt. Durch sie sollen Sachverhalte besser veranschaulicht werden. Ein Kontextszenario (DATech, 2008) beschreibt Arbeitsprozesse (Arbeitstätigkeiten) eines Benutzers, die mit Hilfe des zu entwickelnden Anwendungssystems erledigt werden sollen. Hierbei wird versucht Bedingungen und Einfluss des Nutzungskontexts auf die Ausführung der Tätigkeit zu analysieren. Das Kontextszenario enthält keine Informationen über die tatsächliche *Interaktion* des Anwenders mit dem System.

Ein Use-Szenario (DATech, 2008) ist die Beschreibung von Arbeitstätigkeiten, die ein Benutzer zur Erledigung einer Arbeitsaufgabe in Interaktion mit einem Anwendungssystem erledigt. Ein Szenario dieser Art dient der Ermittlung von Anforderungen an das zu entwickelnde System. Use-Szenarien können beispielsweise auch durch „Screen Shots“ erläutert werden. Während der Entwicklung dienen Use-Szenarien der Beschreibung des Interaktionsentwurfs und somit der Umsetzung von *Nutzungsanforderungen* für jede im Nutzungskontext identifizierte wichtige Arbeitsaufgabe eines Benutzers (= Kernaufgabe). Entwicklungsbegleitend oder zur Systemabnahme dienen Use-Szenarien auch als Testszenario oder als Testprotokoll.

Jarke (1999) setzt Szenarios als Zwischenglied bzw. Zwischenprodukt im Entwicklungsprozess ein. Aus seiner Sicht sind Szenarios Konstrukte, die eine Menge möglicher Aktivitäten oder Zustände beschreiben. Carroll (2000) sieht demgegenüber Szenarien als treibende Elemente über den gesamten Entwicklungsprozess. Rosson & Carroll (1992) schlagen dazu vor, potenzielle Objekte des künftigen Systems aus fertigen Szenarios zu extrahieren und in Form eines Netzwerks von Objekten anzuordnen. In einem nächsten Schritt sollten den Objekten bestimmte Funktionen zugeordnet werden. Diese Vorgehensweise unterstützt eine Beschreibung aller Objekte, die sehr nahe am Benutzer und seiner Sichtweise liegt. Es wird empfohlen die objektorientierte Analyse und das objektorientierte Design mit einem szenariobasierten Ansatz zu verbessern.

Eine Schwäche von Szenarien als integratives Konstrukt ist es, dass sie als informelle Darstellungen üblicherweise in natürlicher Sprache verfasst und oft unzulänglich bzw. mangelhaft sind und für die schwierige Diskussion zwischen Benutzern, Entwicklern, Usability Experten und anderen Beteiligten mit jeweils unterschiedlichen Motiven und Hintergründen nicht ausreichen. Szenario-Darstellungen in natürlicher Sprache entbehren oft der erforderlichen Präzision in der Ausdrucksweise und bringen auch Zwei- oder Mehrdeutigkeit mit sich. Formale Darstellungen (z.B. Aktivitätsdiagramme, Sequenzdiagramme) von Szenarien helfen, das Problem der Zwei- bzw. Mehrdeutigkeit zu lösen und erleichtern eine formale Prüfung von Eigenschaften bzw. der Wirksamkeit von Anforderungen (Validierung). Diese formalen Darstellungen sind auf der Gegenseite aber oft schwer zu verstehen und damit auch schwerer weiterzuentwickeln. Dies gilt sowohl für Neulinge in diesem Betätigungsfeld als auch für Benutzer. Seffah et al. (2005, S. 46) weisen hier auf einen notwendigen Trade-off zwischen der Präzision von formalen Darstellungen und der bequemen Besprechung von Szenarien im Kontext der Ausführung von Aufgaben hin. Designer und Benutzer müssen in der Lage sein, Beschreibungen von Szenarien zu erörtern und weiterzuentwickeln und das über den gesamten Entwicklungszyklus hinweg. Dazu bedarf es eines vielfältigen Medieneinsatzes, um bestehende Optionen zu verschiedensten Zwecken und Zielsetzungen und aus verschiedenen Sichten zu diskutieren oder Ideenreichtum und Einfallskraft der Beteiligten zu stimulieren. Diese Forderung soll für das Integrationskonzept übernommen werden.

Die vollständige Erfassung von Arbeitsabläufen in Form von Szenarien ist oft nicht erforderlich. Szenarien eines konkreten Projekts beschränken sich auf wesentliche Teile von Arbeitsabläufen im Nutzungskontext (Kontextszenario). Aus den erfassten Szenarien sind in einem weiteren Arbeitsschritt *Nutzungsanforderungen* zu entwickeln und zu validieren. Im Leitfaden für Usability (DATech, 2008, S. 131-167) ist die Erstellung und der Einsatz von Szenarien ausführlich erläutert. Die Ausführungen sind auf die Prüfzwecke des Gesamtdokuments ausgerichtet. Eine Übernahme von wichtigen Elementen dieser Ausführungen für eine zweckneutrale Methodenbeschreibung (im Sinne von „nicht auf Prüfzwecke ausgerichtet“) erscheint sinnvoll. Das trifft insbesondere für die dort enthaltenen Erfahrungswerte und nützlichen Anwendungsempfehlungen zu. In der Methodenzusammenführung können diese als wertvolle Anreicherung für eine integrative Methodenbeschreibung dienen.

Szenarien werden von der DATech als Dokumentationsmethode verstanden. Zur Erhebung von Szenarien werden Methoden wie die Aufgabenanalyse, die teilnehmende Beobachtung, die Benutzerbefragung, Dokumentenanalyse und zur Prüfung von Systemen auch die Methode der Inspektion empfohlen. Als weitere mögliche Erhebungsmethoden für Szenarien seien hier noch die Analyse kritischer Vorfälle bzw. die ethnografische Beobachtung genannt (ISO/TR 16982, 2002).

Teilnehmende Beobachtung

In Verbindung mit den frühen Phasen der Entwicklungsarbeit ist hier von der teilnehmenden Beobachtung im Nutzungskontext die Rede. In diesem Kontext und mit steigender Komplexität von Arbeitsabläufen wird empfohlen (DATech, 2008, S. 125) die Beobachtungsergebnisse mittels Szenarios zu dokumentieren. Die sachliche Richtigkeit der erfassten Objekte, Beziehungen und (kritischen) Merkmale der Arbeitstätigkeit muss durch die beobachtete Person bestätigt werden. Die teilnehmende Beobachtung aus Sicht der Entwicklung von Anwendungssystemen wird in Abschnitt 5.4 (S. 201) kurz charakterisiert.

Die teilnehmende Beobachtung kann auch mit Elementen von Interviews angereichert werden und es entsteht ein Wechselspiel zwischen Beobachtung und Befragung. Zusätzlich kann die beobachtete Person dazu animiert werden, während ihrer Tätigkeit laut mitzudenken (Thinking aloud). Dies erhöht die Komplexität der Erhebungsaufgabe für den Entwickler erheblich, erlaubt aber zusätzliche Erkenntnisse über den Arbeitsablauf bei gleichem Zeitaufwand.

Thinking aloud

Das laute Sprechen des Beobachteten während der Beobachtung bzw. während seiner gewohnten Tätigkeit wird als Thinking aloud bezeichnet. Die Methode ist ideal um Benutzer vertieft in den Entwicklungsprozess einzubeziehen. Die Benutzer müssen vorher in die Methode eingeführt werden (Briefing). Die Aufgabenstellung für die Benutzer muss klar sein. Die Durchführung läuft im Rahmen der teilnehmenden Beobachtung. Ein bewusster Abschluss des Methodeneinsatzes mit einer Rückschau sollte nie fehlen (Holzinger, 2004, S. 4-5).

Holzinger hebt die Einfachheit und praktische Verwertbarkeit dieser Methode besonders hervor. Das „Laute Denken“ erlaubt Einsicht in die mentalen Prozesse von Benutzern (Nisbett & Wilson, 1977). Mit dieser Erkenntnis lassen sich von versierten Analytikern interessante Untersuchungsszenarien konstruieren und in empirischen Untersuchungen eventuell zusätzliche Erkenntnisse zur Akzeptanz von Anwendungssystemen finden. Die Methode stammt ursprünglich aus der Problemlöseforschung (Holzinger, 2004 mit Verweis auf Claparède, 1932 und Duncker, 1945). Holzinger schildert die Anwendung des Thinking aloud in Zusammenhang mit Rapid Prototyping-Methoden und Usability-Tests. Er bezeichnet das Thinking aloud dabei als eine Königsmethode im Usability Engineering.

2.4.4.3 Walkthrough-Verfahren und Claims Analysis

In einem ersten Nutzungskonzept bzw. Teilen davon, als Ergebnis der Kontextanalyse, tauchen in der Regel Widersprüche auf. Um diese systematisch auszuräumen kann die Methode der *Claims Analysis* angewendet werden. Der Einsatz erfolgt z.B. im Rahmen von moderierten Workshops und kann auch in Walkthrough-Verfahren eingebettet sein. In diesem Zusammenhang können auch erste (Papier-)Prototypen (Paper Mock-up) auftauchen und dabei helfen, geplante Abläufe der Interaktion zu simulieren. Aus erhobenen Szenarien sind Anforderungen an das zu entwickelnde System zu generieren. Relevante Arbeitsabläufe können anhand der erarbeiteten Szenarien im Rahmen von Walkthrough-Verfahren evaluiert werden.

Zu Walkthrough-Verfahren existieren Ausprägungen wie z.B. *Cognitive Walkthrough* (z.B. Usability first, 2008) oder Pluralistic Walkthrough (z.B. Nielsen, 1993). Im Pluralistic Walkthrough-Verfahren werden vorliegende Abläufe aus mehreren Perspektiven durchgedacht, d.h. in Gedanken simuliert. Entwickler, Usability Engineers, Analytiker und Benutzer durchdenken die dokumentierten Arbeitsabläufe gemeinsam. Es ist dies eine erste Evaluation von Zwischenergebnissen in einem iterativen Gestaltungsprozess. Auf Basis dieser Evaluationsergebnisse erfolgt dann die weitere Gestaltung. Der Pluralistic Walkthrough ist besonders geeignet für den Einsatz in einem frühen Entwicklungsstadium und dient der Verbesserung der benutzerorientierten Gestaltung.

In Workshops früher Entwicklungsphasen, die u.a. der *formativen Evaluation* von Zwischenergebnissen dienen, gilt es auch, sich mit Forderungen an das zu entwickelnde System auseinanderzusetzen. Einen Rahmen für die systematische Auseinandersetzung bietet die Methode der Claims Analysis. Mit der Durchführung einer *Claims Analysis* soll verhindert werden, dass aus Annahmen und Wünschen der Stakeholder beliebige Anforderungen resultieren und dass Designentscheidungen in das Belieben von Designern gestellt werden. Insofern trägt eine *Claims Analysis* maßgeblich zur Validierung von Anforderungen und Lösungsentwürfen bei (DATech, 2008).

Das Wort „*Claim*“ wird im Englischen im Sinne von Forderung, Anspruch und Schaden verwendet. Mit *Claims Analysis* ist ein Prozess gemeint, der alle drei Bedeutungen einschließt. In diesem Prozess werden Forderungen der am Entwicklungsprojekt beteiligten Interessenvertreter (Stakeholder) untereinander abgegrenzt und verglichen, um durch Konsensfindung die Vorteile eines Lösungsvorschlags zu maximieren und die Nachteile (Schäden) zu minimieren. Das Finden von Kompromissen ist charakteristisch für alle Designprozesse („Design is Compromizing“, Winograd, 1997). Auch Rosson & Carroll (1992, S. 77) folgen diesem Verständnis der systematischen Gegenüberstellung von Vor- und Nachteilen von Lösungsansätzen unter der Bezeichnung „Claims Analysis“.

Eine *Claims Analysis* ist demnach als Methodenbündel zu verstehen um Widersprüche in den vorangegangenen Anforderungsfestlegungen und Designs (Kontextszenarien oder Nutzungsszenarien) zu identifizieren. Wie Anforderungsfestlegung und Design gestaltet sein müssen, um eine Claims Analysis zu fördern, wird nicht näher beschrieben. In den Beschreibungen der DATech (2008, S. 30-31) heißt es lediglich, es müssen Bedingungen geschaffen werden, die eine Claims Analysis fördern, damit o.a. Widersprüche aufgedeckt werden.

Für jede Forderung (*Claim*) sollte ein klarer Bezug zu einem Szenario hergestellt werden, um abschätzen zu können, inwieweit die gestellte Forderung eine Förderung bzw. Verbesserung eine Behinderung oder Verschlechterung des Ablaufs darstellt. Die Analyse von *Claims* kann in der Folge dazu führen, dass bereits hergeleitete Nutzungsanforderungen mit Bedingungen versehen werden (Conditional Usage Requirements), unter denen sie in Produktanforderungen umzusetzen sind (DATech, 2008).

Eine Claims Analysis wird meist im Wege eines (oder mehrerer) Workshops durchgeführt. Moderiert wird der Workshop idealerweise von einem Usability-Engineer. Er ist in der Lage, mit der Bewertung von *Nutzungsanforderungen* und Lösungsvorschlägen den Projektleiter zu unterstützen und Entscheidungen zielführend vorzubereiten. Die Methode der Claims Analysis eignet sich also dazu, eine Synthese der Vorstellungen von Entwicklern und Benutzern herbeizuführen. Allein die Gegenüberstellung von Vor- und Nachteilen von Forderungen bzw. Lösungsvorschlägen kann bewirken, dass sich Benutzer darüber klarer werden, was die eigentlichen Anforderungen an eine Lösung sind. Die begleitende Dokumentation solcher Workshops wird empfohlen (DATech, 2008).

2.4.4.4 Prototyping und Usability Prototyping

Das Ergebnis von Kontextanalysen sind *Nutzungsanforderungen*, welche in ihrer Gesamtheit ein Nutzungskonzept bilden. Noch nicht hinreichend geklärte Nutzungsanforderungen können durch die Erstellung von Prototypen besser beurteilt und weiterentwickelt werden (= exploratives Prototyping). Mittels explorativem Prototyping können auch verschiedene Lösungsmöglichkeiten veranschaulicht und anhand des Modells (Prototyp) diskutiert werden. Dieser Art werden Lücken zwischen Nutzungskonzept (Sicht des Benutzers) und Designkonzept (Sicht des Designers) in einem synthetischen Prozess gemeinsam bearbeitet. Zur Bewertung dieser Sichten kann ergänzend die Methode der Claims Analysis herangezogen werden. Vor- und Nachteile werden so lange diskutiert, bis es eine Einigung auf eine angemessene Lösung gibt. Dem Benutzer wird gezeigt, dass es verschiedene Lösungsmöglichkeiten geben kann. Designentscheidungen werden begründet und dokumentiert (DATech, 2008, S. 28). Das Pendant zum explorativen Prototyping ist das evolutionäre Prototyping. Dieses

wird überwiegend zum Oberflächendesign eingesetzt. Ist Einigkeit über bestehende Unklarheiten erzielt worden, kann der Prototyping-Prozess beendet werden.

Ähnlich wie bei der Kontextanalyse bezeichnet das Usability Prototyping ein Bündel von möglichen Vorgehensweisen, die situativ auszuwählen sind und entsprechend adaptiert eingesetzt werden können. Darunter fallen Methoden wie das Erstellen von Mock-Up, Walkthrough-Verfahren (z.B. Cognitive-Walkthrough oder Pluralistic-Walkthrough), verschiedene Nutzungstests, aber auch moderierte Workshops. Die besprochenen Methoden zur Kontextanalyse, das Usability Prototyping und die Claims Analysis, stehen in enger Verbindung zueinander.

Das Prototyping gilt auch als Ansatz zur Gestaltung von Anwendungssystemen mit ausgeprägter Benutzerbeteiligung. Ziel dabei ist es, die Qualitätsverbesserung (Struktur-, Prozess- und Ergebnisqualität) bereits während der Entwicklung in möglichst frühen Phasen vorwegzunehmen. Für diese Vorgehensweise typisch ist das vielfache Durchlaufen von kurzen Planungszyklen (Analyse, Entwurf, Implementierung, Evaluierung). Nach jedem Zyklus wird das Systemverhalten hinsichtlich der definierten Anforderungen (Validierung) und auch in Bezug auf die Akzeptanz geprüft (evaluiert). Es wird dabei unterschieden zwischen evolutionärem Prototyping, experimentellem Prototyping, explorativem Prototyping und schnellem Prototyping (rapid Prototyping). Prototyping ergänzt in der Regel andere Vorgehensweisen (Heinrich et al., 2004, S. 532-533). Die unterschiedlichen Typen von Prototyping lassen sich funktionsorientiert und situativ in Vorgehensmodelle integrieren (Abb. 20, S. 80; Pomberger et al., 1991, S. 45). Intensiv mit prototypingorientierter Software Entwicklung befassten sich in den 1980er und Anfang der 1990er Jahre u.a. Floyd (1984), Jörgensen (1984), Conell & Schafer (1989), Keller (1989), Bischofberger (1990), Pomberger et al. (1991) und Pomberger, Pree & Stritzinger (1992). Diese Entwicklung hing eng zusammen mit der Etablierung der objektorientierten Programmierung zu dieser Zeit.

Prototyping als Methode lässt sich während des gesamten Entwicklungsprozesses mit unterschiedlichem Charakter in der Durchführung einsetzen. Die Nutzer werden mittels Prototyping im Rahmen von Gesprächen sowie Interviews, Gruppensitzungen und Tests konsequent in die Systemerstellung miteinbezogen. Dadurch wird nicht nur das Know-how der Mitarbeiter für die Anwendungssystementwicklung verfügbar, sondern es können umgekehrt auch die Bedürfnisse der Anwender hinsichtlich Information und Qualifizierung ermittelt werden. Das Prototyping kann beispielsweise auch durch Videolernsysteme und andere Medien unterstützt werden. Dies erleichtert es in diesem Zusammenhang zusätzlich Schwachstellen des Arbeitsprozesses zu identifizieren und auszuräumen. Der Arbeitsablauf wird verbessert. Der gesamte Beteiligungsprozess der Mitarbeiter trägt wiederum zu deren fachlicher und metho-

discher Weiterqualifizierung bei. Mit dieser Charakteristik ist Prototyping eine Vorgehensweise, der die Integration des Benutzers immanent ist.

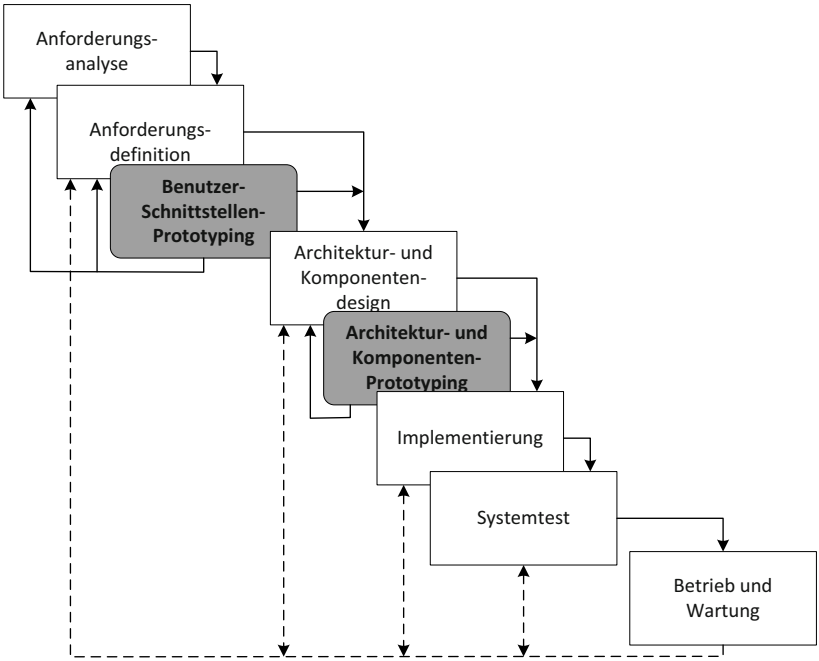

Abb. 20: Positionierung des Prototyping im Entwicklungsprozess

2.4.5 Das Verstehen des Systems durch den Anwender

Ein wesentlicher Zweck des Usability Engineering ist es, die Verstehbarkeit von Systemen für den Anwender aufrechtzuerhalten bzw. permanent zu verbessern. Das Verstehen von Anwendungssystemen durch den Benutzer passiert im Zuge der Interaktion über die Benutzeroberfläche und hängt von dessen Vorwissen ab (Abb. 21, S. 81; Schnotz, 1994, S. 214). Nun gibt es aus dieser Sicht zwei Möglichkeiten der Einflussnahme:

1) Das Vorwissen des Anwender wird verbessert (durch Bildung bzw. Ausbildung, vgl. Wedekind et al., 2004a, S. 172).
2) Der Benutzer wird in der Interaktion zusätzlich darin unterstützt, mentale Modelle zu entwickeln.

Punkt 1 ist relativ klar: Um ein besseres *Verstehen* des implementierten Schemas und seines organisationalen Bezugs (Anwendungssystem im Nutzungskontext) zu erreichen, wird der Benutzer spezifisch geschult. Damit können Schemata (Wissen) auf Seiten des Benutzers verbessert werden. Punkt 2 bedarf einer eingehenden Erläuterung. In Anlehnung an Schnotz (1994, S. 214) wird davon ausgegangen, dass bei der Interaktion mit Anwendungssystemen multiple mentale Repräsentationen gebildet werden:

- Mentale Repräsentation der Benutzeroberfläche.
- *Mentales Modell* des Interaktionsinhaltes.
- *Propositionale* Basis (semantisch-syntaktische Verarbeitung).

Wir konzentrieren uns hier auf die semantisch-syntaktische Verarbeitung. Sie steht aus Sicht der sprachbasierten Informatik im Vordergrund.

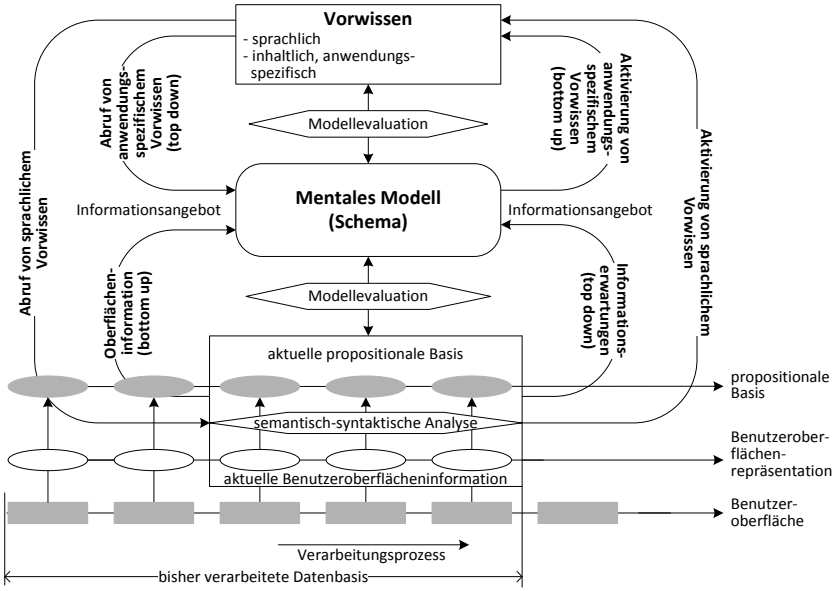

Abb. 21: Schema zum Verstehen als dynamische mentale Modellkonstruktion

Die direkte *Interaktion* mit einem Anwendungssystem erfolgt über die Benutzeroberfläche. Während der Systemnutzung werden die konkreten sprachlichen Darstellungen und die sprachlichen Eigenschaften dieser Darstellungen - Schnotz bezeichnet diese als aktuelle propositionale Basis - über die aktuelle Benutzeroberflächenrepräsentation vom Anwender wahrgenommen. Mittels dieser Benutzeroberflächeninformation werden durch semantisch-syntaktische Verarbeitungsprozesse (z.B. Re-

zeption, Kognition, Applikation – näheres dazu z.B. bei Bartsch, 2001, S. 35, mit Verweis auf de Beaugrande & Dressler, 2001; Zimbardo & Gerrig, 2004, S. 132 u. S. 347) entsprechende *Propositionen* generiert, die dann in der Folge die propositionale Basis bilden. Ziel dieses Prozesses ist der Aufbau eines mentalen Modells auf Basis des dargestellten Inhalts. Diese Schemabildung geht von der propositionalen Basis aus und erfolgt unter Rückgriff auf das Vorwissen. Das Vorwissen in Bezug auf den Anwendungsbereich ist das vorhandene Orientierungs- und Verrichtungswissen des jeweiligen Benutzers. Die jeweiligen Repräsentationen entsprechen den verschiedenen Verarbeitungstiefen beim Menschen.

Während die Benutzeroberflächenrepräsentation auch ohne Verständnis ihres Sinns originalgetreu wiedergegeben werden kann, erfordert die semantisch-syntaktische Verarbeitung mit dem Aufbau einer propositionalen Repräsentation ein bereits etwas tiefer greifendes Verstehen. Der notwendige Verarbeitungsaufwand ist noch relativ gering. Die Herstellung der Zusammenhänge erfolgt durch die Verknüpfung der Propositionen untereinander. Schnotz bezeichnet diesen Vorgang als „lokale Kohärenzbildung“ (Schnotz, 1994, S. 215). Diese Begriffe stammen aus der Psychologie. Ein äquivalenter und bereits eingeführter Begriff aus der sprachbasierten Informatik ist das Verrichtungswissen. Bei der lokalen Kohärenzbildung wird also Verrichtungswissen gebildet.

Die globale Kohärenzbildung ist das Verständnis des Gesamtzusammenhangs und repräsentiert die Bildung von Orientierungswissen. Sie erfordert unter größerem Verarbeitungsaufwand die Konstruktion eines *mentalen Modells*. Die Erledigung von Aufgaben in Form einer eingeübten Abarbeitung kommt einer Erledigung ohne ein *mentales Modell* gleich. Besser zur Problemlösung eignet sich das mentale Modell unter Anwendung von Informationen aus der Benutzeroberfläche. Bildung beruht auf mentalen Modellen. Bei Schulungen wird oft nur eine Abarbeitung eingeübt.

Mit Hilfe mentaler Modelle können ähnlich gelagerte Probleme in gleichem Kontext oder gleich gelagerte Probleme in ähnlichem Kontext bewältigt werden. Bestehende mentale Modelle werden im Zuge der Problemlösung in einem weiteren Rezeptionsprozess elaboriert, d.h. erweitert, verändert und angereichert. Mentale Modelle korrespondieren in der Regel besser mit einem Wirklichkeitsausschnitt als Repräsentationen auf einer Benutzeroberfläche. Bei letzterer handelt es sich um analoge Repräsentationen (Bartsch, 2001, S. 33). Darin werden jeweils nur die situativ relevanten Merkmale repräsentiert.

Die Güte der Darstellungen auf einer Benutzeroberfläche - aus Sicht der Psychologie kann sie in diesem Fall auch als Kohärenz bezeichnet werden - hängt nun davon ab, ob es für mehrere Anwender möglich ist, auf dieser Basis ein einheitliches *mentales Modell* zu bilden. Das ist nicht mit jeder Darstellung auf einer Benutzeroberfläche

möglich. Je größer die Zahl der möglichen mentalen Modelle, die auf Basis eines solchen Bildes geformt werden können, desto schwieriger wird die korrekte Aufgabenbewältigung für den Anwender.

Um das Konzept der mentalen Modelle auf Seiten des Benutzers nun konzeptionell mit der Verstehbarkeit und Anwendbarkeit eines Systems zu verknüpfen, wird das Modell von Sauer verwendet (Abb. 22; Sauer, 1995, S. 162 u. 168). Das Modell orientiert sich ursprünglich an den verschiedenen Ebenen des Textumgestaltens. Die Ebenen entsprechen wichtigen Merkmalen von Texten. Text ist Sprache. Mittels Sprache werden Schemata repräsentiert.

Sauer unterscheidet Gestalt- und Inhaltsseite von sprachlichen Repräsentationen und zwar jeweils global und lokal. Die linke Hälfte des Schemas ist vorgesehen für deren technisch materielle Wahrnehmung. Bei der Lesbarkeit (Readability) geht es um Faktoren, die die visuelle Wahrnehmung auf lokaler Ebene beeinflussen und um deren Einfluss auf die subsemantischen Verarbeitungsprozesse. Beispielsweise können durch mikrotypografische Gestaltung die Verarbeitungsprozesse beim Leser beeinflusst werden. Wenn es um Texte geht sind das beispielsweise Schrifttyp, Schriftgröße, Zeilenabstand, Spaltenbreite, Textverteilung auf der Seite, Anteil von leeren Flächen, Druckbild, Symbole, lokale Piktogramme oder lokale Hervorhebungen als Einflussfaktoren (Sauer, 1995, S. 163). Die technisch materielle Wahrnehmung auf globaler Ebene (also die Tiefe des Textes) bezeichnet Sauer als Brauchbarkeit[12]. Brauchbarkeit bzw. Zugänglichkeit repräsentiert Nützlichkeit im Sinne von „Utility“ nach Nielsen (1993, S. 25f). Die Erschließung von relevanten Systemeigenschaften erfolgt schwerpunktmäßig in der physiologischen Dimension (vgl. Abschnitt 2.4.2, S. 62ff).

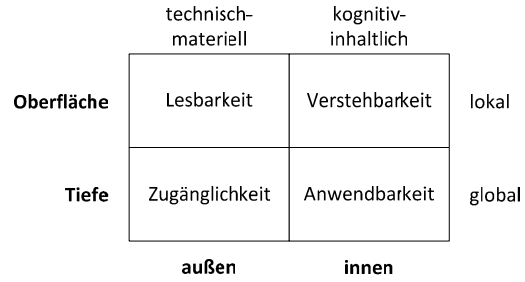

Abb. 22: Vier-Felder-Modell nach Sauer

12 Sauer verwendet statt Brauchbarkeit später den Begriff „Zugänglichkeit“.

Die kognitiv-inhaltliche Verarbeitung der Oberfläche eines Systems auf lokaler Ebene bezeichnet Sauer (1995, S. 162) als Verstehbarkeit (Comprehensability bzw. Understandability), auf globaler Ebene als Anwendbarkeit (Applicability). Nach der erfolgten visuellen Wahrnehmung werden die wahrgenommenen Elemente kognitiv-inhaltlich verarbeitet. Dies entspricht einer semantisch-syntaktischen Analyse, wie sie Schnotz (1994) in seinen Modellen verwendet. Diese Analyse erfolgt in der Regel nicht auf Begriffsebene, wie dies bei der Begriffsklärung der Fall ist, wenngleich Sauer die Begriffsklärung implizit als Optimierungsschritt zu dieser semantisch-syntaktischen Analyse nennt:

- Verdeutlichungen (Wortbildung, Wortwahl, Tempusfolge, Wortfolge).
- Ergänzungen (Explikationen von Zwischenschritten).
- Einfügungen (etwa von Beispielen).
- Auslassungen (von unnötigen Elementen).
- Überprüfungen der Konsistenz des Fachvokabulars.
- Kontrolle der Definitionen.
- Kontrolle der Übergänge von Einheit zu Einheit (Vorverweise und Rückverweise, deiktische Ausdrücke).

All diese Optimierungen orientieren sich am vermutlichen Vorwissen von potenziellen Anwendern und beziehen sich auf die lokale Kohärenzbildung, wie sie mit Verweis auf Schnotz (1994, S. 215) bereits weiter oben eingeführt wurde. Bartsch verweist in diesem Zusammenhang auf die Gricesche Konversationsmaxime der Modalität (Grice, 1993, S. 250), die lautet: „Sei klar!“. Sie umfasst die Untermaximen „vermeide Dunkelheit des Ausdrucks“, „vermeide Mehrdeutigkeit“, „sei kurz (vermeide unnötige Weitschweifigkeit)“ und „der Reihe nach!“.

Aber auch die Maximen der Qualität und Relevanz erfasst Sauer mit dem Ergänzen von Zwischenschritten und Auslassen von unnötigen Elementen. Die Maxime der Qualität kann man als Forderung nach sachlicher Richtigkeit interpretieren, womit sie zum Feld der Anwendbarkeit gehört. Alle genannten Maximen können wiederum auf die konstruktivistische Grundlegung - schrittweise, zirkelfrei und alles explizit machend - zurückgeführt werden.

Durch die Herstellung möglichst guter wechselseitiger Beziehungen zwischen den vier Feldern kann „Nützlichkeit“ erreicht werden. So können z.B. die Informationen im Sinne der Anwendbarkeit zwar vollständig sein, sind sie aber aufgrund mangelnder Zugänglichkeit in einer Anwendungssituation nicht in angemessener Zeit auffindbar, so wird der Zweck der Systemunterstützung nicht erfüllt.

Beispiel:

Suchmöglichkeit ohne eine hinterlegte Indizierung oder alphabetische Ordnung.

Mängel in der Gestaltung der einzelnen Felder können unter zusätzlichem Verarbeitungsaufwand bis zu einem gewissen Grad kompensiert werden. Tendenziell sind Mängel leichter zu kompensieren, wenn sie in einem untergeordneten Feld auftreten z.B. ist ein schlechter Druck erträglicher als das Fehlen einer adäquaten Suchmöglichkeit. Ein Anwender mit großem thematischem Vorwissen in Bezug auf den Anwendungsbereich kann Mängel der inhaltlich-kognitiven Gestaltung kompensieren, ein routinierter Anwender solche der technisch-materiellen Gestaltung. Jede Kompensation ist jedoch nur zu einem gewissen Grad möglich. Dieser Grad ist benutzerindividuell und vorab nicht festlegbar.

Die Erschließung der Verstehbarkeit mit Hilfe von Modellen der „Ursprungsdisziplinen“ des Usability Engineering ist Grundlage für die Zusammenführung von Usability Engineering und der Anwendungsentwicklung. Ausgehend vom Modell von Sauer sind im Hinblick auf die Usability Dimensionen in erster Linie die physiologische und die emotionale Dimension angesprochen. Eine Beeinflussung der Verstehbarkeit erscheint jedoch vor allem über die epistemologische Dimension möglich.

2.4.6 Acceptance Engineering zur systematischen Verfolgung von Akzeptanzzielen

2.4.6.1 Einordnung und Begriffsverständnis

„Acceptance“ bzw. „Akzeptanz“ (lat. „accipere“ = annehmen, billigen, i.ü.S. auch kapieren) ist handlungsbezogen zu verstehen als annehmen, billigen, anerkennen, gutheißen, mit etwas einverstanden sein. Dementsprechend kann *Akzeptanz* definiert werden als Bereitschaft, etwas zu akzeptieren (Duden7, 2001, S. 27), als das Annehmen von Handlungsstrukturen bzw. Bedeutungsinhalten durch ein Individuum. Akzeptanz ist dabei nicht zu verwechseln ist mit „sturer“ Regelkonformität. Akzeptanz im materiellen Bereich bedeutet eine bejahende oder tolerierende Einstellung von Personen oder Gruppen gegenüber der Entwicklung und Verbreitung neuer Techniken, Technologien oder Konsumprodukte. Das Verhalten und Handeln, das diese Akzeptanz zum Ausdruck bringt, ist in dieser Begriffsauffassung inkludiert (Brockhaus,

2006, S. 308). Weil *Akzeptanz* zwar wahrnehmbar ist, aber dennoch nicht „greifbar", wird sie oft auch als Phänomen (Wissen durch Wahrnehmung)[13] bezeichnet.

Abhängig von der Disziplin oder dem Kontext, in welchem der Begriff „Akzeptanz" Anwendung findet, erfolgt die Eingrenzung geringfügig unterschiedlich und ist entsprechend des Anwendungsgebietes motiviert. Aus Sicht der Politikwissenschaft bezeichnet Akzeptanz die aktive oder passive Zustimmung zu Entscheidungen oder Handlungen anderer. Im unternehmerischen Kontext[14] kommt der Akzeptanzbegriff bei der Einführung von neuen Produkten, im Zusammenhang mit Maßnahmen zur Organisationsentwicklung und der damit vielfach verbundenen Einführung von Informationssystemen vor (Bürg & Mandl, 2004, S. 5). Endruweit & Trommsdorff (2002) definieren „Akzeptanz" als „die Eigenschaft einer *Innovation*, bei ihrer Einführung positive Reaktionen der davon Betroffenen zu erreichen". Kritisch ist hier die Verwendung des Begriffes „Eigenschaft" zu sehen. Betont wird der Einführungsprozess, d.h. etwas Neues wird als akzeptiert betrachtet, wenn bei der Einführung zustimmend reagiert wird. Nach dieser Definition gibt es keine Nicht-Akzeptanz von etwas Bestehendem und es wird zudem außer Acht gelassen, dass es sich beim Akzeptieren um einen „länger" dauernden Prozess (inklusive mehrmaliger Nutzung) entlang des sogenannten Innovations-Entscheidungsprozesses (Rogers, 1983, S. 163-206) handelt (Kollmann, 1998). Diese Einschränkung des Akzeptanzverständnisses auf die Einführung von Innovationen tritt auch in der Akzeptanzforschung mehrfach zu Tage (z.B. bei Reichwald, 1982, S. 36; Hillmann, 2007). Kollmann (2006, S. 267) verallgemeinert die Begriffsauffassung von Akzeptanz, abgeleitet aus seinen Forschungsarbeiten in Verbindung mit einem „Kaufprozess als einmalige Handlung", wie folgt (Kollmann, 1998, S. 69):

> Akzeptanz ist die generelle Verknüpfung einer inneren Begutachtung und Erwartungsbildung (Einstellungsebene), einer Übernahme bzw. eines Kaufs (Abschluss) des Produkts (Handlungsebene) und einer freiwilligen – gemessen am Nutzungsverhalten der Teilnehmer – überdurchschnittlich intensiven Nutzung (Nutzungsebene) bis zum Ende des gesamten Akzeptanzprozesses (System wird vom Markt genommen bzw. ersetzt).

Zusätzlich interessant erscheint die Auffassung der Medienpsychologie, wo Krämer Akzeptanz mit Zufriedenstellung (im Sinne der dritten Eigenschaft von Usability ne-

13 Phänomen als Wissen durch Wahrnehmung im Gegensatz zu Noumenon als abstraktes Wissen. Dies ist eine Unterscheidung, wie sie in der Philosophie der Antike vorgenommen wurde (Schopenhauer, Kritik der Kant'schen Philosophie). In einem zeitgemäßen Verständnis von Wissen kann hinsichtlich des Bedeutungsfeldes von Begriffen zwischen „wissen", „meinen" und „glauben" unterschieden werden (Mittelstraß, 2004c, S. 718).

14 Damit sind auch Non-Profit-Organisationen sowie die öffentliche Verwaltung eingeschlossen.

ben *Effizienz* und *Effektivität*) gleichsetzt (Krämer, 2004, S. 655-656). Nach DIN EN ISO 9241-110 (2006) ist mit Zufriedenstellung die subjektive *Zufriedenheit* eines Benutzers gemeint. Die Begriffsauffassung wird also auf Individuen eingegrenzt. Inwieweit organisationale Zufriedenheit damit ebenfalls angesprochen wird, bleibt diese Definition schuldig. Krämer hebt auch hervor, dass es zu den Eigenschaften *Effizienz* und *Effektivität* zahlreiche Untersuchungen gibt, die dritte Eigenschaft, die *Zufriedenheit*, bis dato jedoch stiefmütterlich behandelt wurde. Die Anforderung einer besseren, intensiven und differenzierten Auseinandersetzung mit Zufriedenstellung ergab sich aber auch erst in den letzten Jahren durch die Veränderung der Mensch-Technik-Schnittstellen. Deren Vielfalt bringt einen möglichen und sehr oft bereits gegebenen Zugang von nahezu allen Bevölkerungsschichten mit sich, womit wiederum die zunehmende Bedeutung erklärt werden kann. Mit dieser Auffassung ist für den Begriff Akzeptanz der Rückschluss zur DIN EN ISO 9241-110 gelungen. Inwieweit hier weitere Differenzierungen und Spezifikationen bzw. Konzepterweiterungen vorzunehmen sind, um z.B. Systemeigenschaften im Entwicklungsprozess direkt und aus verschiedenen Blickwinkeln adressieren zu können, wird in Abschnitt 3.5.3.3 (S. 169ff) erörtert.

Auf einen integrierten Ansatz zwischen Arbeitswissenschaften und Betriebswirtschaftslehre zielt jene Begriffsauffassung nach Heinrich et al. (2004, S. 50) ab, die Akzeptanzforschung als Teildisziplin der Wirtschaftsinformatik bezeichnet und ihre Kernaufgabe darin sieht, das Phänomen der *Akzeptanz* aus Benutzersicht zu untersuchen. Es wird dabei nach den Ursachen der vorhandenen oder nicht vorhandenen Bereitschaft der Benutzer gesucht, ein angebotenes Techniksystem zu nutzen. Die gewonnenen Erklärungen dienen schließlich dazu, die Technologieentwicklung so zu beeinflussen, dass unerwünschte Auswirkungen auf die Akzeptanz vermieden werden.

Alle genannten Definitions- und Erklärungsansätze und damit in Zusammenhang stehende Modelle orientieren sich an Veränderungen in Verbindung mit Technik, Technologien bzw. Innovationen und fokussieren den Benutzer. Die Relevanz der Akzeptanzforschung für die benutzerzentrierte Anwendungsentwicklung ist somit gegeben. Fragen nach Akzeptanz hatten schon immer die Betrachtung der Nutzung bzw. Nutzungsbedingungen für das Individuum zum Gegenstand, auf dieser Ebene sind sie Gegenstand des vorliegenden Integrationsvorhabens.

Die Akzeptanzforschung befasst sich mit der Identifizierung von Akzeptanzfaktoren (= Ergebnisse der Akzeptanzforschung) und ist für die Wirtschaftsinformatik ein wichtiges Instrument zur Analyse soziotechnischer Systeme (Wilde, Hess & Hilbers, 2008, S. 1031), wie es Anwendungssysteme typischerweise sind. Es kommen dabei generell methodische Ansätze in Frage, die sich über das gesamte vorhandene Spektrum des

sozialwissenschaftlichen Instrumentariums erstrecken, wobei die Tendenz eindeutig in Richtung Sekundäranalysen und standardisierte Befragungen geht (Quiring, 2006, S. 9).

Die in der Akzeptanzforschung verwendeten Modelle sind mehrheitlich als theoretische Ansätze zur Betrachtung der Akzeptanz von Technologie im Allgemeinen konzipiert und auf keinen speziellen Anwendungsbereich zugeschnitten. Die Industrie entwickelt eigene Akzeptanzmodelle. Diese können im Rahmen eines Integrationsmodells keine Berücksichtigung finden, da sie meist (wenigstens derzeit noch) geheim sind. In Zukunft könnten sich jedoch auch von dieser Seite Standards (so genannte de facto Standards) entwickeln.

Die Erfahrung aus vielen Projekten zeigt, dass die meisten Menschen eine Veränderung nicht auf Basis von wissenschaftlichen Studien (wo die Konsequenzen beschrieben wären) bewerten. Auch wenn solch objektive Bewertungsansätze nicht irrelevant sind, nehmen die Anwender (insbesondere Erstanwender) ihre Einschätzungen und Bewertungen sehr subjektiv vor und lehnen sich überwiegend an Erfahrungsberichte an, die sie von anderen Anwendern, die bereits Veränderungen gleicher Art mitgemacht haben, erhalten (Rogers, 1983, S. 18). Diese Abhängigkeit stellt einen wesentlichen Punkt für die methodische Unterstützung der Beeinflussung der Benutzerakzeptanz bereits ab einer sehr frühen Entwicklungsphase dar und untermauert die Notwendigkeit, die Bedürfnisse des Individuums (also des Anwenders) und nicht einer Organisationseinheit oder einer gesamten Organisation zu berücksichtigen und damit auch zu be-einflussen. Das ist zu ergänzen durch die Erkenntnisse von Venkatesh, Morris, Davis & Davis (2003), die die Performance-Erwartung als die stärkste Wirkungsvariable im Akzeptanzmodell bezeichnen, wobei diese ihrerseits belegt ist durch die Eigenschaften bzw. Erwartungen hinsichtlich Zweckmäßigkeit, extrinsischer Motivation, Aufgaben-Fit, relativen Vorteil, der durch die Nutzung entsteht, und Ergebniserwartungen.

Für die Anwendungsentwicklung ist in einem ersten Schritt das Wechselspiel zwischen Technologie und Individuum von Bedeutung. Sie bildet den globalen Kontext für die Adressierung der Verbesserung der Akzeptanz und damit der Usability (als Teil der Akzeptanz) von Anwendungssystemen. Die Beeinflussung kann sich dabei nicht nur auf Faktoren der „practical Acceptability“ beziehen, unter denen Nielsen unter anderem die Einflussfaktoren für Usability im engeren Sinn subsumiert (Nielsen, 1993, S. 25). Für die Anwendungssystementwicklung sind also nicht nur Faktoren von Akzeptanz im Sinne von Usability (nach Nielsen) interessant, sondern auch Faktoren der „social Acceptability“, da das Phänomen „Akzeptanz“ (oder einzelne Faktoren davon) bereits in einer frühen Entwicklungsphase bewusst und möglichst ganzheitlich adressiert werden soll. Wie diese Beeinflussung systematisch erfolgt, sollte im Rah-

men der Methodenintegration erörtert werden. Methoden zur Beeinflussung von Akzeptanz sind jedoch in keinem der besprochenen Modelle beschrieben.

Die verhaltensorientierte Betrachtung öffnet uns den Raum zum Acceptance Engineering innerhalb der Anwendungssystementwicklung, also zur systematischen Beschreibung und Beeinflussung von relevanten Aktivitäten und Systemeigenschaften die *Akzeptanz* betreffend. Das *Acceptance Engineering* ist demnach als ein Konzept zu verstehen, das weit über das Usability Engineering hinausgeht und diesem auch übergeordnet ist, legen wir die Definition von Nielsen (vgl. Abschnitt 2.4.1, S. 58ff) zugrunde. Mit dieser Auffassung sind konsequenterweise auch ethische Aspekte des Acceptance Engineering darzulegen. Diese Diskussion kann allerdings im vorliegenden Rahmen nicht geführt werden.

Innerhalb der sprachkritisch-organisationszentrischen Anwendungssystementwicklung wird der Akzeptanzbegriff nicht auf die Akzeptanz von Technologien bzw. Innovationen eingeschränkt und auch nicht auf eine politische oder philosophische Sichtweise reduziert. Es geht in der Anwendungssystementwicklung aus Sicht desjenigen, für den es etwas zu akzeptieren gibt, grundsätzlich um die *Akzeptanz* von „Veränderungen“, vorerst unabhängig davon, wie sich diese darstellen. Die hier verwendete Begriffsauffassung umfasst somit – im Sinne von Rogers (1995, S. 11) – die Akzeptanz von Ideen, Arbeitsweisen und Objekten, die nicht tatsächlich neue Erfindungen sein müssen, sondern lediglich von Organisation oder Individuum als neu wahrgenommen werden, demnach also objektiv gesehen nicht einmal eine Neuerung darstellen müssten (es handelt sich um subjektive *Innovationen*). *Akzeptanz* wird in Anlehnung an Kollmann (2006, S. 265-266) als ein dynamisches Konstrukt verstanden, welches in der Anwendungsentwicklung als zentrales Qualitätsziel in Verbindung mit Veränderungen in Organisationen verfolgt wird. Für die Entwicklung von Anwendungssystemen kann das Qualitätsziel „Akzeptanz“ über die Qualitätsmerkmale von Akzeptanz nach dem UTAUT-Modell greifbar werden. Diese Faktoren gilt es bei Bedarf im Entwicklungsprozess systematisch zu beeinflussen. Akzeptanz stellt in Anlehnung an Nielsen (1993, S. 24-25) für die Anwendungsentwicklung den übergeordneten Begriff zu Usability dar, d.h. die Verfolgung von Akzeptanzzielen als Qualitätsziele kann auch die Verfolgung der Verbesserung der Gebrauchstauglichkeit (Usability) umfassen. Die Hinführung zu einem grundlegenden Verständnis der systematischen Beeinflussung von *Akzeptanz*, im Sinne eines ingenieurmäßigen Vorgehens als Acceptance Engineering bezeichnet, erfolgt in den folgenden Abschnitten.

Aufgabe der Akzeptanzforschung ist die Klärung der Akzeptanzbereitschaft bzw. wie *Akzeptanz* erreicht respektive beeinflusst werden kann. Es finden dabei Untersuchungsmethoden aus dem Bereich der empirischen Sozialforschung ebenso Einsatz wie markt-, werbe- und betriebspsychologische sowie betriebssoziologische Untersu-

chungen (Brockhaus, 2006, S. 308). Aus Sicht der Psychologie steht der Begriff „Akzeptanzforschung“ als kritische Bezeichnung für einen Forschungs- und Anwendungsbereich der Arbeits- und Organisationspsychologie. Bei der Einführung neuer Technologien besteht die Aufgabe in diesem Kontext nicht selten darin, innerhalb des Unternehmens Marketing für diese neuen Technologien zu betreiben, um sie für die vorgesehenen Benutzer akzeptabel zu machen (Schuler & Sonntag, 2007; von Rosenstiel, 2007, S. 109-110).

Nach Reichwald (1982, S. 36), Hillmann (2007) und Endruweit & Trommsdorff (2002) sollte die sozialwissenschaftliche Akzeptanzforschung vor allem auf der Nutzerseite bei Innovationen ansetzen. Dabei gilt es, die Gründe für eine Annahme bzw. eine Ablehnung einer konkreten *Innovation* durch die potenziellen Nutzer zu erforschen. Somit unterscheidet Akzeptanzforschung zwei Zielsetzungen:

1) **Empirisch-analytische Zielsetzung** – zur Erklärung der Wechselbeziehung zwischen Technologieanwendung und Technologiefolgen.
2) **Pragmatisch-konstruktive Zielsetzung** – zur Beeinflussung der Technologieentwicklung und der Technologienutzung, im Hinblick auf individuelle und organisationale Ziele.

Für empirisch-analytische Forschungsfragen haben Venkatesh et al. das UTAUT-Modell als Forschungsrahmen entwickelt (vgl. Abschnitt 2.4.6.3, S. 94ff). Es ist ein weitgehend statisches Modell, welches die nachgewiesenen Einflussfaktoren für *Akzeptanz* systematisch in Relation zueinander setzt. Für pragmatisch-konstruktive Zielsetzungen sind die Erkenntnisse von Venkatesh & Morris (2000) bzw. Venkatesh et al. (2003) ebenfalls grundlegend. Kollmann (1998, S. 135) legt in dem von ihm entwickelten Akzeptanzmodell zusätzlich Wert auf dynamische Aspekte von *Akzeptanz*, spezifiziert jedoch in diesem Rahmen keine Vorgehensweisen zur Beeinflussung von Akzeptanz. Spezielle Modelle zur systematischen Beeinflussung von Akzeptanz in der Anwendungsentwicklung existieren nicht.

Verwandte Theorieansätze, wie z.B. die Einstellungsforschung, befassen sich lediglich mit der inneren Begutachtung eines Objektes, ohne jedoch mit einer konkreten Handlung verbunden zu sein (Rogers 1962 u. 1983, S. 163ff; Kollmann, 2006, S. 264, mit Verweis auf Meffert, 1976; Kroeber-Riel & Weinberg, 2003, S. 168ff). In diesem Kontext ebenfalls anzuführen ist die Wirkungsforschung. Sie befasst sich im Rahmen der Kommunikationswissenschaften vorzugsweise mit Phänomenen von *Akzeptanz* auf gesamtgesellschaftlicher Ebene, diese Ebene kann in Anlehnung an Loos (2005, S. 4) auch als Makroebene bezeichnet werden. Betrachtungen auf Makroebene sind für die vorliegende Problemstellung nicht relevant.

Im Kontext der Akzeptanzforschung stehen die Adoptions- und die Diffusionsforschung. Eine Begriffsabgrenzung wird trotz vorhandener Unterschiede oft nicht vorgenommen, auch wenn die *Adoptionsforschung* eher der Psychologie und die Akzeptanzforschung eher der Wirtschaftsinformatik zugeordnet werden könnte. Grundlegende Modelle der *Adoptionsforschung* und Diffusionsforschung beruhen weitgehend auf den Forschungsarbeiten von Rogers & Shoemaker (1971 u. 1983).

Für die Anwendungsentwicklung relevant ist die systematische Begleitung von Adoptionsprozessen von Anwendungssystemen in Unternehmen, wobei hier die *Adoption* bereits zu Beginn der Entwicklungsarbeit startet. Besondere Bedeutung kommt darin der systematischen Beeinflussung von Akzeptanz zu. Benutzer-Akzeptanz-Modelle können unterstützend eingesetzt werden.

2.4.6.2 Basiskonzept von Benutzer-Akzeptanz-Modellen

Aus den von Venkatesh et al. (2003, S. 427) untersuchten acht Modellen für Benutzerakzeptanz (Tab. 5, S. 92) kann als gemeinsamer Nenner ein Basiskonzept herausgefiltert werden (Abb. 23, S. 91; Venkatesh et al., 2003, S. 427). Das Konzept geht von der individuellen Reaktion als Ergebnisbündel von Einflüssen bestimmter Faktoren auf eine bevorstehende Nutzung von Informationstechnologie aus und geht über zur *Akzeptanz* bei der tatsächlichen Nutzung. Dazwischen tritt optional noch die Betrachtung der Intentionen zur geplanten Nutzung. Dieses Konzept liegt allen Benutzer-Akzeptanz-Modellen zugrunde, die in das UTAUT-Modell Eingang gefunden haben. Auch das dynamische Akzeptanzmodell von Kollmann (2004, S. 143) unterscheidet, analog zum gezeigten Basiskonzept, in jeder Phase die maßgebenden Indikatoren im Hinblick auf erwartete und erreichte Betrachtungsebenen. Die Rückkoppelung (in Form einer ex-ante-Betrachtung mit Feedback) fehlt bei Kollmann, da seine Ausführungen auf den einmaligen Kaufprozess von Telekommunikationssystemen abzielen (Kollmann, 2000, S. 71).

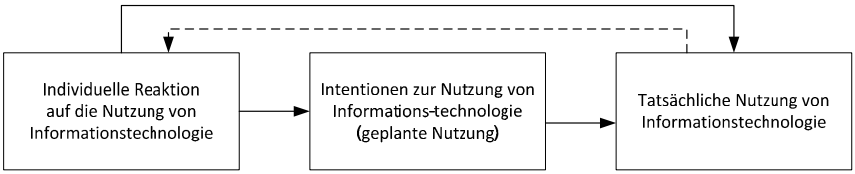

Abb. 23: Basiskonzept von Benutzer-Akzeptanz-Modellen

Alle untersuchten Modelle (vgl. Tab. 5, S. 92) gehen davon aus, dass die Intention zur geplanten Nutzung, welche ihrerseits wiederum von den verschiedensten Parametern beeinflusst wird, von der tatsächlichen Nutzung zu differenzieren ist. Beide Faktoren sind zentrale Bausteine auf dem Weg zur Beurteilung von *Akzeptanz*. Diese

grundlegende Einteilung kann auch als Unterscheidung zwischen einem Einstellungsaspekt und einem Verhaltensaspekt verstanden werden. Der Einstellungsaspekt bildet dabei eine affektive (gefühlsorientierte) Komponente und berücksichtigt vorwiegend motivational-emotionale Aspekte. Der Verhaltensaspekt bildet eine kognitive (verstandesorientierte) Komponente und bedingt in diesem Rahmen die Gegenüberstellung von Kosten und Nutzen einer *Innovation* im jeweils individuellen Kontext. Der Verhaltensaspekt ist im Gegensatz zum Einstellungsaspekt beobachtbar (Müller-Böhling & Müller, 1986; Bürg & Mandl, 2004, S. 5; Simon, 2001).

Modellbezeichnung	**Abkürzung**	**Maßgebende Quellen**
Theory of Reasoned Action	TRA	Fishbein & Ajzen, 1975; Ajzen & Fishbein, 1980; Sheppard, Hartwick & Warshaw, 1988; Davis, Bagozzi & Warshaw, 1989
Technology Acceptance Model	TAM	Davis, Bagozzi & Warshaw, 1989; Davis, 1989; Davis & Venkatesh, 2000
Motivational Model	MM	Davis, Bagozzi & Warshaw, 1992; Vallerand, 1997; Venkatesh & Speier, 1999
Theory of Planned Behavior	TPB	Taylor & Todd, 1995b; Ajzen, 1991; Mathieson, 1991; Harrison, Mykytyn & Riemenschneider, 1997
Combined Technology and Planned Behavior Model	C-TAM-TPB	Taylor & Todd, 1995a
Innovation Diffusion Theory	IDT	Tornatzky & Klein, 1982; Rogers, 1995; Moore & Benbasat, 1996
Model of PC Utilization	MPCU	Triandis, 1977; Thompson, Higgins & Howell, 1991
Social Cognitive Theory	SCT	Bandura, 1986; Compeau & Higgins, 1995a; Compeau & Higgins, 1995b

Tab. 5: Benutzerakzeptanzmodelle als Grundlage zur Entwicklung des UTAUT-Modells

Die Theory of Reasoned Action wurde von den Sozialpsychologen Ajzen & Fishbein entwickelt und entsprang aus Untersuchungen von Einstellungen und Verhaltensweisen (Fishbein & Ajzen, 1975; Ajzen & Fishbein, 1980). Das TRA-Modell besteht aus drei grundlegenden Bausteinen:

- **Behavioral Intention** – Beabsichtigtes Verhalten (BI).
- **Attitude** – Einstellung (A).
- **Subjective Norm** – Subjektive Normen (SN).

Im TRA Modell wird behauptet, dass diese drei Bausteine wie folgt miteinander in Beziehung stehen: BI = A + SN. Das heißt, das beabsichtigte Verhalten einer Person hängt von der Einstellung zum Verhalten und von deren subjektiven Normen ab. Mit BI wird also die relative Stärke der Absicht gemessen, ein gewisses Verhalten an den Tag zu legen bzw. wie eine Person sich in einer gewissen Art und Weise zu verhalten

beabsichtigt. Anders gesagt, das gewollte Verhalten einer Person ist vorausbestimmt durch ihre Einstellung, das Verhalten selbst und wie sie darüber denkt, dass andere Leute dieses wahrnehmen. Die Einstellung einer Person kombiniert mit deren subjektiven Normen formt das beabsichtigte Verhalten, wobei die Einstellung und die subjektiven Normen verschieden gewichtet sein können.

Im Bereich der Untersuchung von Konsumentenverhalten ist das TRA-Modell bedeutend, nicht nur weil mit Hilfe des Modells eventuell die Intentionen von Konsumenten vorhergesagt werden können, sondern auch weil das Modell eine relativ einfache Grundlage dafür bietet, zu identifizieren, wo und wie versucht werden kann das Verhalten von Zielgruppen zu ändern (Sheppard et al., 1988, S. 325). Zu beachten ist, dass das TRA Modell nicht für *Adoptionen* im Business Bereich, sondern eben für den Endverbraucherbereich erstellt wurde. Trotzdem sind die Einflussfaktoren teilweise von so großer Bedeutung, dass sie auch im UTAUT-Modell (Unified Theory of Acceptance and Use of Technology) Berücksichtigung finden. Das bestärkt auch die Wichtigkeit der Adressierung dieser Einflussfaktoren innerhalb der Systementwicklung, da auch hier Einflussfaktoren für Endverbraucher maßgebend sein können, z.B. bei der Entwicklung von „off the shelf" Produkten.

Das Technology Acceptance Model (TAM) ist eine der populärsten Weiterentwicklungen der Theory of Reasoned Action und des Motivational Models (MM) von Davis et al. Ende der 1980er bzw. Anfang der 1990er Jahre des letzten Jahrhunderts (Davis et al., 1989; Bagozzi et al., 1992). In beiden Modellen (TRA und MM) sind wesentliche verhaltensorientierte Elemente verankert und es wird davon ausgegangen, dass, wenn jemand sich in einer bestimmten Weise verhalten will, er das ohne Einschränkung tun kann. Das steht natürlich im Gegensatz zu den Gegebenheiten in der Realität, wo es immer gewisse Einschränkungen und Bedingungen zu berücksichtigen gilt (beschränkte Fähigkeiten, organisationale Einschränkungen, unbewusste Gewohnheiten usw.).

Unter den vielen aufgestellten Modellen, die dazu dienten ein besseres Verständnis davon zu erhalten, welche Faktoren die *Akzeptanz* von Informationstechnologie beeinflussen, ist das Technology Acceptance Model eines der einflussreichsten, aber auch robustesten, was die Erklärung von Verhalten in Verbindung mit Einführung von IT-Systemen betrifft. Der Hauptzweck von TAM war es, eine Grundlage für die Erforschung von Einflüssen von externen Variablen auf interne Absichten, Meinungen und Verhalten im Allgemeinen zu haben (Davis, 1989; Chau, 1996; Davis & Venkatesh, 2000).

Im TAM ging man davon aus, dass Nutzungsintentionen, insbesondere zur Zweckmäßigkeit und zur einfachen Nutzung, immer die elementaren Bestimmungsgrößen bei der Einführung von IT in Organisationen sind. Dies führte dazu, dass viele Messgrößen

für Verhalten in den Vorgängermodellen ersetzt wurden durch zwei Messgrößen für Technologieakzeptanz und zwar: „Ease of Use“ und „Usefulness“. Diese beiden Eigenschaften hob Davis (1989, S. 320) besonders hervor und zwar als *Perceived Usefulness* und als *Perceived Ease of Use.* Wahrgenommene Zweckmäßigkeit (*Perceived Usefulness*) ist definiert als ein Spielraum, den eine Person wahrnimmt, die davon ausgeht, dass ein Anwendungssystem ihre eigene Leistungsfähigkeit im Anwendungsbereich verbessert. Wahrgenommene einfache Nutzung (*Perceived Ease of Use*) bezieht sich auf einen Rahmen innerhalb dessen eine Person davon ausgeht, dass die Nutzung eines Systems keiner mentalen Anstrengung bedarf (Davis, 1989). Diese zwei Bestimmungsgrößen können als Basis für die Untersuchung von Einstellungen und Verhalten im Hinblick auf die Benutzung von speziellen Systemen dienen, da sie ihrerseits die Absichten bei der Benutzung beeinflussen und in der Folge das Benutzerverhalten generieren. Alle Erkenntnisse aus TAM fanden auch adäquate Berücksichtigung im später entstandenen UTAUT-Modell.

Die Untersuchungen von Davis ergaben starke kausale Zusammenhänge zwischen „Ease of Use“ → „Usefulness“ → „Usage“ (und zwar in genau dieser Reihenfolge und Richtung). Um diese zwei Einflussgrößen auch in den Kontext von sozialen Einflüssen sowie instrumentellen, kognitiven Prozessen zu stellen, erweiterten Davis & Venkatesh (2000) das originale TAM zu TAM2. Beide Technology Acceptance Models wurden gebaut, um Einführungen von IT-Systemen in Unternehmen zu untersuchen. Dementsprechend darf die Relevanz der hier betrachteten Faktoren zur Beeinflussung von *Akzeptanz* aus Sicht der Entwicklung von Anwendungssystemen eingeschätzt werden (Marchewka, Liu & Kostiwa, 2007, S. 94-95). Von Venkatesh et al. (2003, S. 446-447) wurde das TAM mit sieben weiteren Modellen der Akzeptanzforschung empirisch verglichen (Tab. 5, S. 92). Aus den Erkenntnissen daraus wurde schließlich das UTAUT-Modell formuliert.

2.4.6.3 UTAUT-Modell

Das UTAUT-Modell wird nachfolgend in seinen wesentlichen Zügen vorgestellt. Es ist eine Aggregation und Konsolidierung von acht bestehenden Akzeptanzmodellen. In einem ersten Schritt wurden Gemeinsamkeiten der acht zugrundeliegenden Modelle begutachtet. In jedem Modell konnte eine Einflussgröße gefunden werden, die in allen Betrachtungen signifikant war, und dieses Konstrukt zeigte jeweils auch die stärksten Auswirkungen bei den Untersuchungen. Als Beispiele seien hier folgende Konstrukte angeführt:

- **Attitude** – Einstellung, Verhalten in TRA und TPB.
- **Perceived Usefulness** – wahrgenommene Brauchbarkeit, Nützlichkeit in TAM/TAM2 und C-TAM-TPB – es wird wahrgenommen, dass etwas imstande (geeignet, tauglich) ist, zweckmäßig (vorteilhaft, nützlich) eingesetzt zu werden.
- **Extrinsic Motivation** – Extrinsische Motivation in MM.
- **Job-fit** in MPCU.
- **Relative Advantage** – relative Wettbewerbsvorteile in IDT.
- **Outcome Expectations** – Ergebniserwartungen in SCT.

In einem weiteren Schritt wurde festgestellt, dass es einige andere Konstrukte gab, die sich zu Beginn als signifikant erwiesen, sich aber mit der Zeit als unsignifikant erwiesen haben. Dazu gehören:

- **Perceived Behavioral Control** – wahrgenommene Verhaltenssteuerung in TBP/DTBP und C-TAM-TPB.
- **Perceived Ease of Use** – Wahrgenommene einfache Nutzung in TAM/TAM2.
- **Komplexität** in MPCU.
- **Ease of Use** – einfache Nutzung in IDT.
- **Self-Efficacy and Anxiety** – Selbstvertrauen und Angst in SCT.

Schließlich hatte Spontanität bzw. Freiwilligkeit versus Pflichterfüllung im Kontext einen Einfluss auf die Signifikanz von Konstrukten, die zu den sozialen Einflüssen zählen:

- **Subjektive Normen** in TBP/DTBP, C-TAM-TBP und TAM2.
- **Soziale Faktoren** in MPCU.

Image (IDT) war nur in Umsetzungen signifikant, in denen Pflichterfüllung eine Rolle spielte. Sieben Konstrukte scheinen also zur direkten Beeinflussung der Einstellung zur Systemnutzung signifikant zu sein (Abb. 24; Venkatesh et al., 2003, S. 447). Details zu den Einflussgruppen, deren weitere Unterteilung sowie deren Definition und mögliche Messskalen finden sich in Anhang 5.3 (S. 196ff).

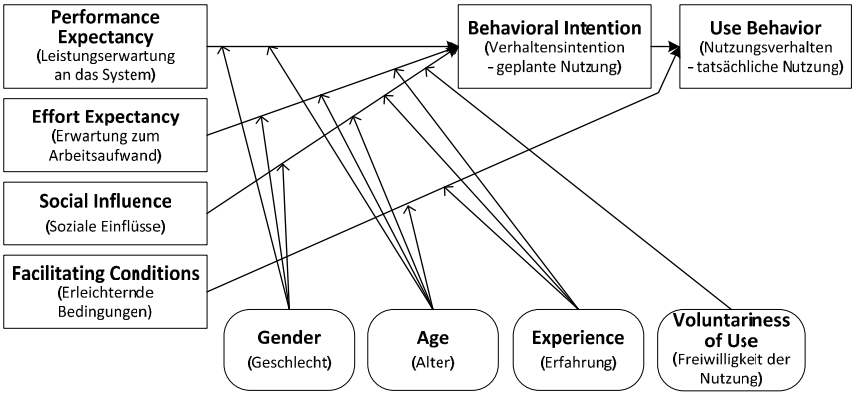

Abb. 24: UTAUT-Modell

Venkatesh et al. (2003) unterteilen diese Einflussgrößen im UTAUT-Modell in drei Gruppen, die jeweils direkten Einfluss auf die sich schließlich ergebende Verhaltensintention bei den Anwendern haben. Die Intensität des Einflusses der jeweiligen Gruppe auf die Verhaltensintention hängt ihrerseits wiederum von Geschlecht, Alter und der Erfahrung der Anwender ab. Die Freiwilligkeit der Nutzung beeinflusst zusätzlich die sozialen Einflüsse. Auf die sich daraus ergebenden Verhaltensintentionen können noch erleichternde Bedingungen wirksam werden, wie z.B. das Angebot zusätzlicher individueller Schulungen. Erst nach dem Wirksamwerden dieser Bedingungen liegt das eigentliche Nutzungsverhalten vor. Demnach sind alle in diesem Modell systematisch gebündelten Faktoren Einflussgrößen auf die *Akzeptanz* einer Informationstechnologie, da sich diese im Nutzungsverhalten äußert.

Das UTAUT-Modell konsolidiert vorangehende Studien über Technologieakzeptanzmodelle. Leistungserwartungen und Erwartungen den Arbeitsaufwand betreffend, verkörpern im UTAUT-Modell die elementaren Einflussfaktoren. Wahrgenommene Zweckmäßigkeit (Perceived Usefulness) und wahrgenommene einfache Nutzung (Perceived Ease of Use) nach dem TAM-Modell sind in diesen Parametern integriert. Im UTAUT-Modell gilt, dass die Erwartungen den Arbeitsaufwand betreffend, im Einfluss auf die *Akzeptanz* von Anwendungssystemen auch dann signifikant sein können, wenn die einfache Nutzung durch eine erweiterte und lang anhaltende Nutzung nicht mehr signifikant ist. Demnach kann davon ausgegangen werden, dass wahrgenommene einfache Nutzung nur in den Anfangsstadien der Nutzung einer neuen Technology herausragen wird und dass sie auch nur dann einen positiven Effekt auf die wahrgenommene Zweckmäßigkeit einer Technologie haben kann.

Darüber hinaus versucht das UTAUT-Modell zu erklären, wie individuelle Unterschiede die Technologienutzung beeinflussen. Genauer gesagt, die Beziehung zwischen wahrgenommener Zweckmäßigkeit, einfacher Nutzung und Nutzungsintentionen kann durch Alter, Geschlecht und Erfahrung der Benutzer mäßig verändert werden (Abb. 24). So variiert z.B. die Stärke des Zusammenhangs zwischen Leistungserwartungen und Verhaltensintention mit Alter und Geschlecht des Benutzers und ist signifikant stärker für männliche und jüngere Anwender. Der Effekt zwischen wahrgenommener einfacher Nutzung und Verhaltensintention ist ebenfalls beeinflusst durch Alter und Geschlecht und zwar derart, dass er signifikant ist für weibliche und ältere Anwenderinnen. Diese Effekte nehmen mit steigender Erfahrung ab.

Vom UTAUT-Modell als Grundlage für Untersuchungsanordnungen, um ein besseres Verständnis der Einflussfaktoren für Technologienutzung zu erhalten, verspricht man sich viel. In seinen empirischen Überprüfungen kommen Venkatesh et al. (2003) zu der Erkenntnis, dass UTAUT die jeweils vorliegenden Daten besser erklärt als jedes der acht ursprünglichen Modelle. Das UTAUT-Modell erklärt dabei bis zu 70 % der individuellen Adoptionsentscheidungen (Loos, 2005, S. 6).

Die im UTAUT-Modell verwendeten Beurteilungsskalen sind neu, da sie eine Kombination von einigen älteren bzw. bestehenden Skalen darstellen. Empirische Tests zur Bestätigung solcher Skalen sind im vorliegenden Rahmen nicht vorgesehen. Ungeachtet dieser fehlenden Tests wird versucht die Einflussgrößen, die das UTAUT-Modell für Technologieakzeptanz bündelt, auf konzeptioneller Ebene in den Entwicklungsprozess zu integrieren. Dazu bietet sich vordergründig die Integration auf Methodenebene an.

2.4.6.4 Akzeptanzmodell nach Kollmann

Kollmann identifizierte Defizite bei den klassischen Akzeptanzansätzen zur Erfassung von Vermarktungsbesonderheiten, insbesondere fehlte ihm der dynamische Aspekt von Akzeptanz. Kollmann (2006, S. 265, mit Bezug auf Kollmann 1998, S. 135) entwickelte ein alternatives Akzeptanzmodell und unterscheidet darin die Begriffe *Attitude*, *Adoption*, *Diffusion* und *Akzeptanz* in Verbindung mit Untersuchungen zum Konsumentenverhalten beim Kauf von Produkten. Er nimmt neben einer inhaltlichen, begrifflichen Erschließung auch eine phasenorientierte Abgrenzung der genannten Begriffe vor (Kollmann, 1999, S. 136-141).

Die Entwicklung von *Akzeptanz* bezogen auf Individuen gliedert Kollmann phasenorientiert in *Attitude* (= Einstellungsphase - mit dem Ergebnis der Einstellungsakzeptanz) gefolgt von Adoption (= Handlungsphase - mit dem Ergebnis der Handlungsakzeptanz) und Acceptance (= Nutzungsphase - mit dem Ergebnis der Nutzungsakzeptanz). Innerhalb der einzelnen Phasen wiederum werden identifizierte,

problemspezifische bzw. domänenspezifische Indikatoren herangezogen, um eine multidimensionale Interpretation von „Akzeptanzzuständen“ (Einstellungs-, Handlungs- bzw. Nutzungsakzeptanz) auf den verschiedenen Ebenen zu ermöglichen. Eine Übersicht von Phasen und Zuständen sowie Einflussfaktoren bietet Abb. 25 (S. 98; Kollmann, 2006, S. 266).

Akzeptanz wird von Kollmann als dynamisches Konstrukt verstanden, welches sich über den Betrachtungszeitraum verändern kann. Er strukturiert diesen Betrachtungszeitraum nach Phasen und ermöglicht dabei eine multidimensionale Interpretation nach Einflussfaktoren auf unterschiedlichen Ebenen. Kollmann unterscheidet neben der Einstellungsebene, Handlungsebene und Nutzungsebene auch noch eine Akzeptanzebene, Konstruktebene und Prognoseebene, die ihrerseits wiederum von der Makroökonomischen Umwelt, der sozio-kulturellen Umwelt, der technologischen Umwelt und der politisch-rechtlichen Umwelt beeinflusst werden. *Akzeptanz* wird als Nutzungskontinuum interpretiert. Die Skalierung reicht von hoher Akzeptanz mit tendenziell hoher Nutzungshäufigkeit bzw. –intensität bis zu niedriger Akzeptanz mit tendenziell geringer Nutzungshäufigkeit bzw. –intensität. Die Intensität der Nutzung ist eine variable Größe, die ebenfalls einer zeitlichen Veränderung unterliegt. Der Akzeptanzprozess, wie ihn Kollmann skizziert (Abb. 25), resultiert zwar aus den Gegebenheiten des Untersuchungsbereiches für Telekommunikationsprodukte, die Elemente können jedoch als grundlegend betrachtet und daher auf den Entwicklungsprozess übertragen werden.

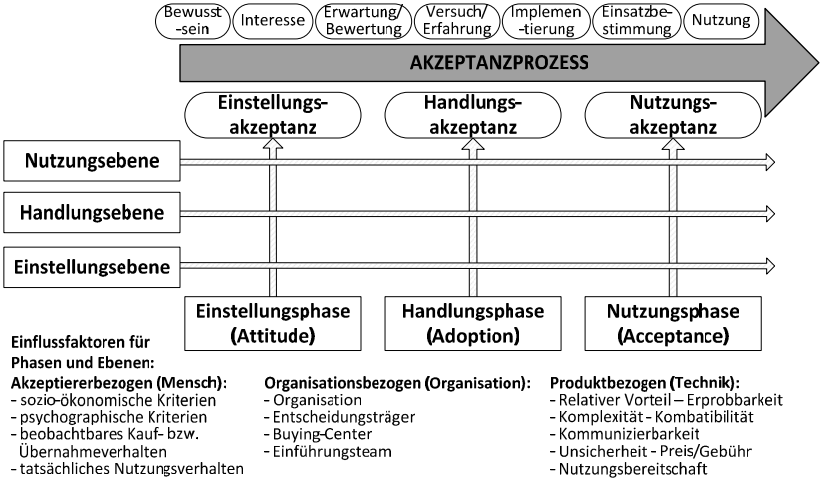

Abb. 25: Wesentliche Elemente des Akzeptanzmodells nach Kollmann

3 Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik

> Wir wollen in unserem Wissen vom Gebrauch der Sprache eine Ordnung herstellen: eine Ordnung zu einem bestimmten Zweck; eine von vielen möglichen Ordnungen; nicht die Ordnung.
>
> (Ludwig Wittgenstein)

> Indem ein Geschehnis auf ein vereinbartes Schema gebracht wurde, ist es verfügbar (einsetzbar, benutzbar) wie ein verwendbares Gerät.
>
> (Kamlah & Lorenzen, 1967, S. 58)

3.1 Einleitung und Übersicht

Anwendungsentwicklung bedeutet, Anwendungssysteme zu schaffen (vgl. Abschnitt 2.3.1, S. 35). Anwendungssysteme dienen der Lösung von Problemen in Anwendungsbereichen. Es geht implizit um die Gestaltung von Arbeit.[15] Um diesen Zweck erfüllen zu können, werden Anwendungssysteme auf den jeweiligen Anwendungsbereich zugeschnitten. Dafür sind vor allem adäquate Beschreibungen eines Anwendungsbereichs notwendig, u.a. sind das dynamische oder statische Beschreibungen der Arbeit, der Arbeitsgegenstände und der Arbeitsumgebung.

Liegen relevante Beschreibungen eines Anwendungsbereichs in einer Form vor, dass sie in eine Sprache der Informatik übersetzt werden können, so kann die Informatik daraus ein Anwendungssystem erzeugen. Beschreibungen von Anwendungsbereichen liegen oft nicht in geeigneter Form vor, sodass vorhandene Darstellungen in eine andere Form gebracht werden müssen. Dies kommt der Übersetzung einer vorliegenden Darstellung von einer Sprache in eine andere Sprache gleich. Meist existiert jedoch überhaupt keine Beschreibung eines Anwendungsbereichs oder relevanter Ausschnitte daraus. In diesem Fall ist ein Modell zu rekonstruieren. Zur Modellrekonstruktion werden Gegenstände, deren Beziehung zueinander und Geschehnisse des Anwendungsbereichs erhoben.

---

15 Eine einfache Dreiteilung möglicher Unterstützungsgrade von Menschen bei ihrer Arbeit ist die Unterscheidung nach Arbeitsabnahme, Arbeitserleichterung und Arbeitsbefähigung (Ortner & Eller, 2008; Steitz, 2008). Software und Wissen sind dabei als Potenzialfaktoren zu betrachten. „Software“ als implementiertes Wissen auf Technikseite und „Wissen“ als Orientierungswissen und Verrichtungswissen auf Seiten des Menschen (Ortner & Eller, 2008, S. 57).

Rekonstruktion bedeutet für die sprachbasierte Informatik die vernünftige und systematische Erarbeitung von Wissen. Die Erarbeitung des Wissens erfolgt in natürlicher Sprache. Das erarbeitete Wissen wird in Form von Aussagen dokumentiert. Diese Sammlung von Aussagen stellt ein erstes Modell des Anwendungsbereichs in der Sprache des Anwendungsbereichs dar (vgl. Abschnitt 2.2.3, S. 24ff). Wesentlich ist dabei eine Präzisierung der Fachterminologie des Anwendungsbereichs im Verlauf der *Rekonstruktion*. Die entstehende gemeinsame Fachsprache wird systematisch verwaltet und für die weitere Entwicklungsarbeit zur Verfügung gestellt. Als Werkzeug wird dazu idealerweise ein *Repository* eingesetzt.

In weiteren Schritten wird das rekonstruierte Modell vom anwendersprachlichen Kontext in einen informatiksprachlichen Kontext bzw. entwicklersprachlichen Kontext transferiert. Diesen Gestaltungsprozess unterstützen Vorgehensweisen der *Modellierung*, bis schließlich ein maschinenlesbares Modell vorliegt. Im Anschluss daran steht das erarbeitete Modell des Anwendungsbereichs als Anwendungssoftware zur Verfügung. Die Anwendungssoftware ist auf eine geeignete Informationstechnologie abzustimmen und wird dann in den Kontext des Anwendungsbereichs zurückgeführt. Diese Rückführung passiert vor allem in Form einer Stützung des Nutzers in seinem praktischen Tun und ist eine Stabilisierung der Entwicklungsergebnisse. Erst nach erfolgter Stabilisierung kann von einem Anwendungssystem im Sinne des Ebenenmodells der Informationsverarbeitung (vgl. Abb. 10, S. 36) gesprochen werden. Diese grundlegenden Zusammenhänge der Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik verdeutlicht Abb. 26 als systematisches Vorgehen in drei Teilen über drei Sprachebenen. Das Vorgehen folgt dem konstruktivistischen Ansatz: „Aus dem Gebrauch - für den Gebrauch“. Es erfordert einen konstruktiven und zugleich pragmatischen Umgang mit Sprache.

Der erste Teil (*Rekonstruktion*) beschäftigt sich mit dem hochhieven der Gebrauchssprache des Anwendungsbereichs auf eine fachsprachliche Ebene. Damit werden Probleme aus der Praxis mittels Rekonstruktion von Unklarheiten bereinigt (geklärt) und auf eine Sprachebene vom Standpunkt des Anwendungsbereichs gehoben. Im zweiten Teil (*Modellierung*) gilt es, die Problembeschreibung auf eine Sprachebene vom Standpunkt der Informatik zu heben. Dies erfolgt mittels Modellierung auf anwendersprachlicher und entwicklersprachlicher Ebene. Aufgabe des dritten Teils (*Stützung*) ist es, die Ergebnisse von Rekonstruktion und Modellierung in Form einer Lösung dem Gebrauch zuzuführen. Diese Stabilisierung der Entwicklungsergebnisse erfolgt mit Fokus auf das Verfügungs- und Orientierungswissen der Nutzer zum Zweck der Stützung ebendieser in ihrem praktischen Tun im Anwendungsbereich.

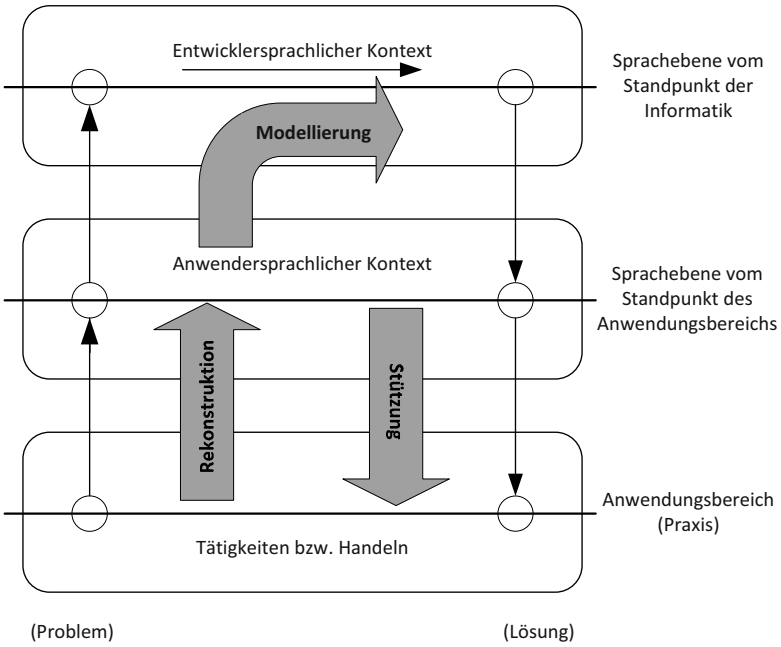

Abb. 26: Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik

Damit die Rekonstruktion gelingen kann, greift die sprachbasierte Informatik auf die sprachlogischen Grundlagen der Informatik zurück. Wedekind et al. (2004a-2004e u. 2005) stellten diese in sechs Aufsätzen als Plädoyer für einen Informatikunterricht als Grundbildung zur Diskussion. Sie fordern darin, dass Informatik im Unterricht konstruktiv einzuführen ist, d.h. schrittweise, zirkelfrei und alles explizit machend. Konstruktiv beschreiben sie die Rolle von Sprache und den Umgang mit Sprache aus deren Gebrauch heraus für die Informatik. Erweitert und vertieft wurde diese Beschreibung von Ortner (2005) zur Grundlegung der sprachbasierten Informatik in der gleichnamigen Monografie.

Dass die Sprache und damit die Sprachwissenschaft als Grundlagendisziplin herangezogen werden, ist auf einen Paradigmenwechsel in der Sprachphilosophie zurückzuführen. Gemeint ist damit der Übergang von der klassischen Sprachauffassung zur linguistisch-pragmatischen Sprachdeutung, welche als Ergänzung zur analytischen Sprachphilosophie nach Frege (1848-1925) verstanden werden kann. Die erste Phase

dieses Paradigmenwechsels vollzog sich im sogenannten Linguistic Turn[16]. Es war die Auffassung geboren, dass alle Problemstellungen mit dem Medium Sprache auszudrücken sind. Dies betrifft sowohl sprachliche als auch nichtsprachliche Objekte. Die später folgende pragmatische Wende (Pragmatic Turn) war inspiriert vom späten Wittgenstein (1899-1951). Es standen nicht mehr nur formale Sprachen und deren Semantik im Vordergrund, sondern es wurde ein pragmatisches Verständnis von Sprache ermöglicht, das den Akt des Sprechens als eine Form des Handelns in den Mittelpunkt stellt. Austin (1911-1960) und Searle (geb. 1932) präzisierten und systematisierten diese Auffassung in der Sprechakttheorie (vgl. Abschnitte 2.2.3, S. 24ff u. 5.1, S. 185ff). Über das Medium Sprache kann somit eine pragmatische Reflexion des Denkens erfolgen. Das bis dort herrschende Primat der Wahrheitsbedingungen trat etwas zurück zugunsten einer an kommunikativer Kooperation orientierten Vorgehensweise. Diese sprachlogischen Grundlagen mit ihren konstruktivistischen Wurzeln begründen die sprachbasierte Informatik.

Im Rahmen der Anwendungsentwicklung in der sprachbasierten Informatik kommen verschiedene Sprachtypen zur Anwendung:

- Kommunikative Sprachen
- Normierte Sprachen
- Spezifikationssprachen

Kommunikative Sprachen sind die Gebrauchssprachen, in denen sich Menschen miteinander unterhalten (z.B. natürliche Sprache, Alltagssprache, Umgangssprache). Dazu gehören auch die Fachsprachen der Unternehmen mit ihrem spezifischen Fachvokabular. Der Einsatz kommunikativer Sprachen dominiert die Rekonstruktion.

Eine normierte Sprache[17] ist der natürlichen Sprache sehr ähnlich, jedoch sind sowohl deren Grammatik als auch deren Wortschatz im Vergleich zur natürlichen Sprache eingeschränkt. Die erarbeiteten geklärten Begriffe werden in einer einfachen Satzbauweise miteinander verbunden (Lehmann, 1999; Ortner, 2003a), es entstehen Elementarsätze (Wedekind et al., 2004b). Durch die Normierung besitzt diese Sprache, im Gegensatz zur natürlichen Sprache, eine eindeutige Pragmatik (Zweck), eine eindeutige Syntax (Form) und eine geklärte Semantik (Inhalt bzw. *Bedeutung*). Es handelt sich um eine sogenannte *Zwischensprache.* Eine Zwischensprache wird eingesetzt mit dem Ziel, die natürlichsprachlichen Aussagen, die für Entwicklungszwecke

16 Eine ausführliche Herleitung und Erörterung dazu findet sich bei Heinemann (2006, S. 41-43), welche wiederum Bezug nimmt auf Graeser (2002, S. 30) sowie Bergmann, Moore, Russell, Wittgenstein, Carnap und andere.

17 Lehmann (1999) verwendet den Begriff „genormte Unternehmensfachsprache“ synonym zum Begriff „reglementierte Sprache“ bzw. „normierte Sprache“.

oft missverständlich und nicht eindeutig sind, in eindeutige Aussagen zu überführen. Das ist notwendige Voraussetzung dafür, dass die normierten Aussagen zunächst in eine Systemspezifikation und schließlich in Programmiersprache überführt werden können. Eine normierte Sprache kann gleichermaßen als entwicklernahe und anwendernahe Sprache verstanden werden. Mittels Normsprache können Sachverhalte klar und eindeutig beschrieben werden. Der Übergang von der Sprachebene der Informatik zur Sprachebene des Anwendungsbereichs kann damit bewältigt werden.

Spezifikationssprachen sind Kunstsprachen (= konstruierte Sprachen wie z.B. UML, BPMN, XML, C#), die zur Modellierung und Implementierung von Vorgängen benutzt werden. Die Schemabildung beginnt mit der Rekonstruktion und wird in der Modellierung fortgesetzt, wobei eine Überleitung der Schemata von natürlicher Sprache in Normsprache und schließlich in Spezifikationssprache erfolgt. Laut Lehmann (1999) ist eine Spezifikationssprache ein Entwurfs- und Beschreibungsinstrument, das der genauen Bestimmung der Eigenschaften eines Systems dient. Diagrammsprachen sind eine spezielle Form von Spezifikationssprachen, welche u.a. die Visualisierung von relevanten Sachverhalten zum Ziel haben.

Das skizzierte Modell der Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik (Abb. 26, S. 101) dient als Grundlage für die Sicht auf die systematische Integration des Usability Engineering auf Basis von Sprache (semantische Integration) in die Entwicklungsarbeit. Darauf aufbauend können neue Theorien und Modelle entwickelt werden oder auch bestehende Theorien wie z.B. die Objektorientierung Anwendung finden. In einem konstruktiven Sinn werden in den folgenden Unterabschnitten die wesentlichen Teile des Vorgehens auf den verschiedenen Ebenen der Anwendungsentwicklung vom Standpunkt der sprachbasierten Informatik beschrieben. Zu jedem Vorgehensschritt wird unmittelbar die sprachbasierte Integration des Usability Engineering besprochen. Die systematische und grundlegende Integration über Sprache auf interner Ebene (semantische Integration), ergänzt durch eine Zusammenführung von Prozessmodellen und einen Gestaltungsrahmen für Methodenbeschreibungen im Sinne einer strukturellen Integration auf konzeptioneller Ebene (vgl. Tab. 1, S. 7), kann die praktische Integration im Anwendungsbereich nachhaltig stützen.

3.2 Rekonstruktion auf fachsprachlicher Ebene

3.2.1 Zweck und Ablauf

Bereits in den 1980er Jahren beschrieb Ortner die Dringlichkeit einer systematischen Rekonstruktion von Sachzusammenhängen im Anwendungsbereich, die von einer durch die Beschreibung manueller Techniken überladenen Darstellung informationsverarbeitender Prozesse befreit ist (Ortner, 1983, S. 159). Damals wurde diese Tätigkeit als „fachsprachliche Rekonstruktion“ und später als „sprachkritische Rekonstruktion“ oder, ohne Adjektiv, auch nur als Rekonstruktion bezeichnet. Es geht dabei um eine vernünftige, systematische Erarbeitung von Wissen.

Die Rekonstruktion ist keine technische Herausforderung, diese liegt eher bei der Konfigurierung von Komponenten, bei Systemarchitekturen und Implementierungsaspekten (technische Integration). Bei der Rekonstruktion stehen *Methodenneutralität*, *Normsprachlichkeit* und deren materialer Charakter (*Materialsprachlichkeit*) und damit verbunden die semantische Integration im Vordergrund. Im Rekonstruktionsprozess gelangt der Entwickler von Aussagen, ausgedrückt in der im Anwendungsbereich üblichen Gebrauchssprache, zu einer Sammlung relevanter Aussagen. Diese Aussagen beruhen auf einer gemeinsamen Fachsprache, die als Sprachstandard des Anwendungsbereichs gilt. Die Bildung von Elementarsätzen im Sinne einer Normierung bildet den Übergang zu Modellierung. Abb. 27 (S. 105) zeigt schematisch den Rekonstruktionsprozess in Anlehnung an Ortner (2003b, S. 30).

Der skizzierte Rekonstruktionsprozess ist an sich bereits mehrteilig, parallel und verzweigt. Mit der Umsetzung in einem Anwendungsbereich wird die Rekonstruktion zusätzlich komplex. In Anlehnung an Ortner (1993, S. 23ff) kann diese, auf der Gebrauchssprache basierende, systematische Erarbeitung von Wissen (= Schematisierung), in drei Vorgehensschritten passieren:

1) Sammlung relevanter Aussagen (aktuelles Geschehen im Anwendungsbereich; Reden über das Handeln im Anwendungsbereich; erste problemorientierte Selektion von Aussagen).
2) Klärung und Rekonstruktion von Fachbegriffen (Reden über das Handeln im Anwendungsbereich; Festlegung von Termini und Prädikatorenregeln; Modifikation von Sprachstandards).
3) Ableitung von Schemata (Einführung einer rationalen Grammatik; Normierung von singulären und allgemeinen Aussagen; problemorientierte Selektion von Aussagen; Prüfung der Aussagen auf Vollständigkeit).

Durch das Vorgehen in Stufen bzw. Schritten erfolgt eine Zerlegung der Gesamtaufgabe in Teilaufgaben. Damit bleibt die Aufgabenstellung überschaubar. Die einzelnen Rekonstruktionsschritte werden in einem Entwicklungsprojekt vielfach durchlaufen.

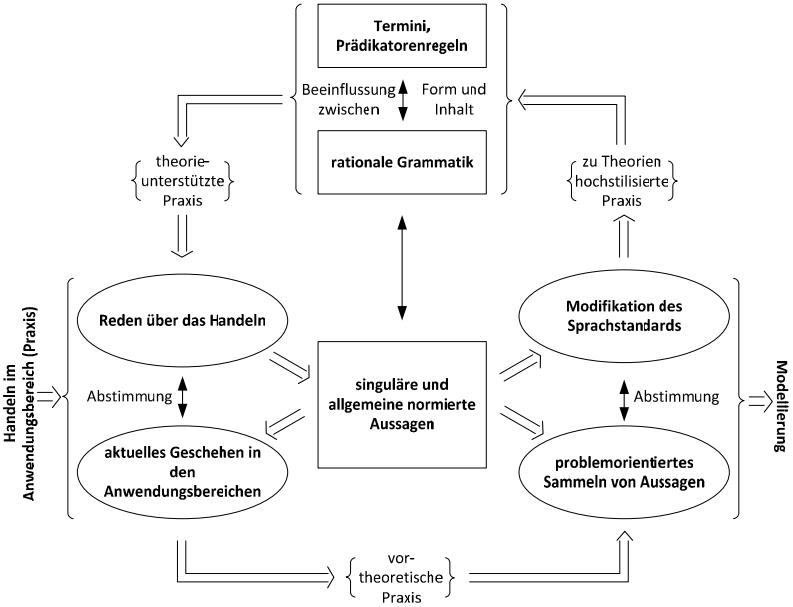

Abb. 27: Rekonstruktionsprozess

In Anlehnung an diese Tätigkeiten zu Beginn der Entwicklungsarbeit wird der erste Teil der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik Rekonstruktion genannt. Die Rekonstruktion geht weit über einen „Nachbau“ im eigentlichen Sinne des Wortes hinaus. Eine gelungene Rekonstruktion antizipiert künftige Ansprüche von Anwendern und Innovationen für den Anwendungsbereich und integriert diese bereits in dieser frühen Entwicklungsphase.

Die Sprachebenen (Abb. 26, S. 101) verdeutlichen die Anwendung unterschiedlicher Sprachen im Verlauf der Entwicklungsarbeit. Die Ebene des Anwendungsbereichs repräsentiert die Praxis. In der Praxis sprechen die Menschen in der natürlichen Sprache in Ausprägung einer Alltagssprache bzw. Umgangssprache miteinander. Zusätzlich zur Umgangssprache werden in einem Anwendungsbereich Fachsprachen, aber auch Fachjargon verwendet. Fachsprachen sind branchenspezifische oder auch unternehmensspezifische, oft standardisierte Sprachen. Unter einem Fachjargon wer-

den kontextspezifische Ausdrucksweisen und Wortbildungen verstanden.[18] Alltagssprache bzw. Umgangssprache und Fachsprache[19] zusammen ergeben die *Gebrauchssprache* eines bestimmten Anwendungsbereichs.

Oft sind Begriffe und deren Anwendung in Fachsprachen nicht eindeutig festgelegt, sondern im Laufe der Zeit „gewachsen". In diesem Fall liegt keine geklärte Fachsprache vor. Eine Klärung von Begriffen ist vorzunehmen. Diese Klärung erfolgt im Rahmen der Rekonstruktion mit Bezug auf einen bestimmten Anwendungsbereich. Dabei werden alle relevanten Ungereimtheiten im Sprachgebrauch von Anwendern und Entwicklern gemeinsam geklärt. Das Ergebnis ist eine gemeinsam geklärte Fachsprache. Mit dieser Klärung erfolgt der Übergang vom Anwendungsbereich auf die Sprachebene vom Standpunkt des Anwendungsbereichs (Abb. 26, S. 101). Der entwicklungsrelevante Anwendungsbereich wird in natürlicher Sprache auf Basis geklärter Begriffe abgebildet.

Damit der nächste Übergang von der Sprachebene vom Standpunkt des Anwendungsbereichs zur Sprachebene vom Standpunkt der Informatik bewältigt werden kann, werden zunächst sogenannte Zwischensprachen eingesetzt. Sie helfen dabei, schrittweise zu informatiksprachlichen Darstellungen zu gelangen, die schließlich eine direkte Implementierung ermöglichen. Eine *Zwischensprache* ist z.B. die *Normsprache.* Abhängig vom Kontext können auch Sprachen aus dem Sprachkanon der OMG (z.B. UML oder BPMN) unterstützend eingesetzt werden. Dies ist möglich, um einerseits eine Visualisierung zu erreichen oder um den Entwickler in seiner gewohnten Vorgehensweise zu unterstützen. Für den zweiten Fall sind Werkzeuge notwendig, die die grafischen Sprachen in Aussagen übersetzen, um eine ausreichende Kommunikation mit dem Benutzer zu gewährleisten (z.B. BPMN to text, TECHNUM, 2009).

3.2.2 Sammlung von Aussagen

Die Sammlung von Aussagen bedeutet Kommunikation über Gegenstände und Geschehnisse eines Anwendungsbereichs in der Gebrauchssprache. Das notwendige Wissen zur Anwendungsentwicklung wird vor allem in kommunikativen Prozessen zwischen Entwicklern und Menschen aus den Anwenderorganisationen gesammelt. Zweck der Sammlung von Aussagen ist es, ein vollständiges Bild über die relevante Ist-Situation eines Anwendungsbereichs zu erhalten und zugleich Anforderungen sowie Rahmenbedingungen für ein zu entwickelndes System zu ermitteln. Dazu wer-

18 Eine klare Trennung zwischen Fachsprache und Fachjargon ist nicht immer möglich.

19 Im Gegensatz zu klaren Abgrenzungen bei standardisierten Fachsprachen, wie sie z.B. in manchen technischen Bereichen möglich sind, wird hier unter der Fachsprache eines Anwendungsbereichs eine jeweilige Fachsprache inklusive des Fachjargons verstanden.

den Erhebungsmethoden eingesetzt, die den Kommunikationsprozess systematisch unterstützen.

Die Auswahl der einzusetzenden Methoden kann die Entwicklung maßgeblich beeinflussen. In Abschnitt 2.3.4.2 (S. 54ff) wurden grundlegende Erhebungstechniken und solche, die im Rahmen des Multipfad-Vorgehensmodells Anwendung finden, aufgezählt. Im Anhang (Abschnitt 5.4, S. 199ff) werden ausgewählte Erhebungsmethoden in Anlehnung an Wedekind (1976) und Lehmann (1999) charakterisiert. Eine umfangreiche Auflistung von Erhebungsmethoden findet sich z.B. bei Boose (1993). Erhebungsmethoden werden nicht nur für die Sammlung von Aussagen eingesetzt, sondern finden über den gesamten Entwicklungsprozess Anwendung.

Bei der Erstellung der Aussagensammlung sind sowohl *empraktische* Erhebungstechniken als auch *epipraktische* Erhebungstechniken anzuwenden. Als epipraktische Erhebungstechniken werden jene Techniken bezeichnet, bei denen die Gegenstände der Rede in der Erhebungssituation nicht unmittelbar präsent sind (Bühler, 1978). Das heißt Erhebungen finden losgelöst von aktuellen Geschehnissen statt, wie dies z.B. in Interviews, Berichten, Dokumentenanalysen, Besprechungen oder Diskussionen der Fall ist. Bei *empraktischen* Erhebungstechniken hingegen ist der Bezug zu aktuellen Geschehnissen unmittelbar gegeben, wie z.B. bei der teilnehmenden Beobachtung. Erhebungstechniken beider Kategorien können interaktiv sein, d.h. das Wissen wird dialogisch rekonstruiert, indem Entwickler, Experten und Anwender über das Handeln reden.

Als Königsmethode zur Erhebung von Aussagen gilt das Interview. Jedes Interview sollte sorgfältig vorbereitet werden. Vor dem eigentlichen Interview gilt es, Interviewleitlinien zu erarbeiten und Informationen über den/die Befragten einzuholen, wie z.B. Informationen zur Funktion und Stellung des Interviewten in der Organisation oder Informationen zu persönlichen Eigenheiten. Der Interviewer oder Analytiker muss sicherstellen, dass die Gesprächsteilnehmer sich vorbereiten können, indem er ihnen die Arbeitsunterlagen rechtzeitig zur Verfügung stellt. Das Ziel eines Interviews sollte möglichst genau vorgeplant und den Teilnehmern bekannt gemacht werden. Bei der Erstellung eines Leitfadens und der Zusammenstellung von Fragen gibt es einige grundsätzliche Dinge zu beachten:

- Die Fragen sollten verständlich, möglichst einfach und nicht zu lang formuliert sein.
- Die Fragen dürfen die befragte(n) Person(en) nicht überfordern, dementsprechend muss darauf geachtet werden, dass kein Wissen implizit vorausgesetzt wird, über welches der Befragte eventuell gar nicht verfügt.

- Die Fragen dürfen nicht suggestiv sein, also nicht von vornherein eine bestimmte Antwort nahelegen.
- Die Eingangsfragen sollten möglichst leicht und wenig kontrovers zu beantworten sein, um dem Befragten den Einstieg zu erleichtern.

Das zentrale Problem der Interviewtechnik ist die Art und Weise, wie die Antworten des Befragten dokumentiert werden (Wedekind, 1976). Die Dokumentation der Befragung ist entscheidend, da darauf aufbauend oft weit reichende Entscheidungen getroffen werden. Grundsätzlich stehen drei Wege für die Dokumentation offen:

- **Gedächtnisprotokoll** – Die Ergebnisse werden später aus dem Gedächtnis protokolliert.
- **Begleitendes Protokoll** – Das Interview wird während des Gesprächs wörtlich protokolliert.
- **Aufzeichnung auf Datenträger** – Das Interview wird auf einen Tonträger oder zusätzlich auch als Video aufgezeichnet.

Wurde kein begleitendes Protokoll erstellt bzw. wurden nur stichwortartige Notizen gemacht, ist eine nachträgliche Dokumentation in Form eines Gedankenprotokolls möglichst zeitnah zum Interview erforderlich. Die Erstellung eines Gedächtnisprotokolls bringt zwar auf den ersten Blick eine Zeiteinsparung, da die laufenden Aufzeichnungen während des Interviews Zeit von allen Beteiligten in Anspruch nehmen. Der Nachteil besteht jedoch darin, dass durch unvollständiges Erinnerungsvermögen oder subjektive Selektion wichtige Sachverhalte verloren gehen können. Ein begleitend erstelltes, wörtliches Protokoll, eine Tonbandaufnahme oder eine Videoaufzeichnung erfordern im Gegensatz dazu eine (oft) mühevolle Auswertung im Anschluss an das Interview. Solche Auswertungen können zeitaufwändig sein und verlangen eine gute Strukturierung der Auswertungstätigkeit. Ein Vorteil von Audio- bzw. Videoaufzeichnungen ist, dass die Auswertung bei Bedarf wiederholt werden kann. Eine Videoaufzeichnung bringt den zusätzlichen Vorteil, dass auch Signale der Körpersprache erfasst werden können. Das ist bei einer begleitenden Protokollierung nur dann sehr gut möglich, wenn eine Person das Gespräch führt und eine zweite Person das Gespräch und zusätzliche Eindrücke protokolliert. In der Regel sind mindestens zwei Interviews bei einem Befragten erforderlich um noch offene Fragen und Widersprüche klären zu können und um Niederschriften zu überprüfen und zu validieren. Weitere Aspekte zur Interviewführung und Verweise auf einschlägige Literatur finden sich im Anhang (Abschnitt 5.4, S. 199ff).

Den einzelnen Kategorien der Erhebungsmethoden (z.B. epipraktisch, empraktisch) werden unterschiedliche Eigenschaften zur Unterstützung der Entwicklungsarbeit zugeordnet. Die Methoden aus den verschiedenen Kategorien ergänzen sich, deshalb

ist eine geeignete Kombination von dialogischen, hermeneutischen und praktischen Erhebungen für die Rekonstruktion sinnvoll und notwendig, um das Anwenderfachgebiet verstehen und Begriffe eindeutig klären zu können. Die konkrete Methodenauswahl erscheint erst im Kontext eines Projektes sinnvoll.

Die einzelnen Erhebungstechniken stoßen schnell an ihre Grenzen, wenn gleichzeitig unterschiedliche Arten von Aussagen erhoben werden (Gorguen & Linde, 1993). Auch die Beachtung des Erhebungsziels, wie z.B. Sammlung von Aussagen, Verfeinerung von Aussagen oder Validierung von erhobenen Aussagen, ist relevant (Guida & Tasso, 1994). Beispielsweise ist es beim Einsatz von passiven oder indirekten Erhebungsmethoden von Vorteil, dass die Befragten nicht beeinflusst werden können. Die Ergebnisse unterliegen keiner Verzerrung aus diesem Grund. Bei solchen Methoden treten Interviewer und Befragte nicht miteinander in Kontakt (z.B. Fragebogen, Dokumentenanalyse, indirekte Beobachtung). Ein Nachteil beim Einsatz von indirekten Methoden ist, dass bei der Beantwortung von Fragen niemand anwesend ist, der Unsicherheiten bezüglich einer Fragestellung klären kann oder der zu einer ausführlichen Antwort motivieren kann. Beim Einsatz von direkten Erhebungstechniken treten Interviewer und Befragter miteinander in Kontakt (z.B. Interview, teilnehmende Beobachtung). Erfolgt die Dokumentation zeitlich versetzt, entsteht ein möglicher Nachteil dadurch, dass die Ergebnisse durch mangelndes Erinnerungsvermögen des Interviewers nicht vollständig dokumentiert werden. Weitere Informationen zu Vor- und Nachteilen von Erhebungsmethoden können z.B. bei Friedrichs (1990) nachgelesen werden.

Ein weiterer Aspekt, den es bei der Auswahl von Erhebungsmethoden zu berücksichtigen gilt, ist der Erhebungsaufwand einschließlich des Aufwands für Vor- und Nachbereitung. Mit einer günstigen Methodenkombination kann der Erhebungsaufwand minimiert werden. Beispielsweise werden exemplarische Rekonstruktionsverfahren als sehr effektiv, aber auch als aufwendig bezeichnet (Ortner, 1994b), da sie oftmals nur im Rahmen von Prototyping oder *Simulationen* anwendbar sind. Bei der Wahl der Erhebungstechniken gilt es außerdem den Wissensstand und das Bildungsniveau der befragen Personen (Novize, Anwender, Experte) zu berücksichtigen (Lenz, 1991).

Ein einfaches Beispiel in Form einer „virtuellen Rekonstruktion“ soll im Folgenden helfen, wesentliche Aspekte der Rekonstruktion zu verdeutlichen. Dazu werden ausgewählte Aussagen aus Interviews realer Projekte herangezogen und punktuell bearbeitet. Auf eine Erörterung der Gesprächsführung wird verzichtet. Ebenso muss die komplexe Darstellung der zeitlichen Tiefe der Rekonstruktionsarbeit im Projektverlauf unterbleiben.

Beispiel:

Verbale Aussage des Interviewpartners: „Bei uns in der Auftragsbearbeitung läuft alles bestens, es sind keine Änderungen der Abläufe erforderlich."

Körpersprachliche Aussage des Interviewpartners: „Sichtbare Ironie; Verdrehen der Augen; …"

Wird allein die verbale Aussage des Beispiels ausgewertet, könnte davon ausgegangen werden, dass keine zusätzlichen Untersuchungen des Arbeitsbereichs Auftragsbearbeitung z.B. in Form von Interviews oder teilnehmender Beobachtung erforderlich sind. Zusammen mit den Eindrücken, die die Körpersprache des Interviewten vermittelt, erhält die Aussage gegenteilige Bedeutung. Sind Interviewer und Auswerter nicht ident, können ohne Visualisierung beispielsweise ironisch gemeinte Aussagen kaum identifiziert werden.

Sowohl die eingesetzten Erhebungstechniken als auch die Fachsprache des jeweiligen Anwendungsbereichs sollte von den durchführenden Experten beherrscht und verstanden werden. Zudem dürfen die erhebenden Personen nicht voreingenommen sein. Das gilt insbesondere für Methoden mit direktem Benutzerkontakt. Über das Sammeln von Aussagen hinaus werden die Erhebungsarbeiten auch dazu genutzt, ein gegenseitiges Verständnis zwischen Entwickler, Anwender und anderen Beteiligten am Entwicklungsprojekt zu erreichen.

Welche Aussagen sind nun relevant? Die Feststellung der Relevanz von Aussagen kann nur kontextbezogen erfolgen. Entwicklungsrelevante Aussagen haben den erforderlichen Sachbezug, sind hinreichend genau und die Vertreter des Anwendungsbereichs stimmen den Aussagen beständig und uneingeschränkt zu. Ist die Relevanz einer Aussage nicht eindeutig, sollte diese Aussage trotzdem erfasst werden bzw. vorerst nicht aus der Aussagensammlung eliminiert werden. Die mögliche Nicht-Relevanz von Aussagen wird meist im weiteren Projektverlauf deutlich bzw. kann im Gespräch zwischen Entwickler und Anwender deutlich werden.

Nach der Selektion entwicklungsrelevanter Aussagen und deren ansatzweise Gruppierung, z.B. nach Arbeitsbereichen oder Aufgaben, entsteht eine Sammlung der erhobenen und relevanten Aussagen in der Gebrauchssprache des Anwendungsbereichs. Es handelt sich dabei um eine Sammlung von Formulierungen über relevante Sachverhalte, Thesen und Meinungen zum Anwendungsbereich.

Eine erstellte Aussagensammlung ist nicht statisch. Die Aussagen sollten laufend auf Gültigkeit und Aktualität geprüft, gegebenenfalls überarbeitet und bei Bedarf erneut einem Rekonstruktionsprozess zugeführt werden. Auch die Festlegung und Sicherung des entsprechenden Geltungsbereiches einer Aussage ist wiederholt zu prüfen.

Beispiel:

Nachdem die Ironie in der Beispielaussage identifiziert werden konnte, wurden in weiteren Erhebungen u.a. folgende Aussagen zur Auftragsbearbeitung eines zu entwickelnden Warenwirtschaftssystems erfasst. Von den Anwendern wurden sowohl Aussagen zur Darstellung der Ist-Situation gemacht als auch Forderungen an ein neues Warenwirtschaftssystem gestellt. Relevanz und Geltung sind für die nachfolgend angeführten Aussagen gegeben.

Nr.	Aussage
1	Es darf kein Lieferschein erstellt werden können, wenn der Lagerbestand nicht ausreichend ist.
2	Wenn der Kunde bestellt, wird ein Lieferschein im Warenwirtschaftssystem erstellt.
3	Ein Lieferschein ist änderbar, bis die Ware ausgeliefert wird. Dies gilt für Preisänderungen und für Mengenänderungen.
4	Es können auftragsspezifisch Rabatte vergeben werden.
5	Die Darstellung von Artikel-, Preis- und Mengenhistorien pro Kunde ist kompliziert.
6	Eine Unterscheidung von Auftrag und Lieferschein ist im Warenwirtschaftssystem nicht möglich.
7	Es ist möglich, Waren vom Kunden zurückzunehmen, die er nicht gekauft hat.
8	Der Belegfluss von der Bestellung bis zur Rechnung und Zahlung soll künftig nachvollziehbar sein.
9	Der Fahrer eines LKW bringt den Lieferschein zum Wareneingang, damit wird die angelieferte Ware kontrolliert.
10	Bestellt ein Kunde Ware, erhält er eine Auftragsbestätigung.
11	Ein Lieferschein enthält neben den Auftragspositionen auch eine Lieferadresse, ein Lieferdatum und einen Platz für die Empfangsbestätigung durch den Kunden.

Gesammelte Aussagen stammen in der Regel von unterschiedlichen Interviewpartnern, von denen jeder über ungleiche *Sprachkompetenz* und *Sprachperformanz* verfügt. Allein aus diesem Grund haben die Aussagen andersgearteten Charakter. Zudem unterscheiden Anwender nicht von vornherein zwischen einem Ist-Zustand und einem Soll-Zustand. Darstellungen der Ist-Situation und Forderungen an ein neues Anwendungssystem werden gerne vermischt. Die Aussagen sind aus Entwicklersicht ungenau und bedürfen einer Präzisierung.

Eine Aussagensammlung wird bereits bei relativ kleinen Projekten sehr umfangreich. Die Komplexität der darin abgebildeten Sachverhalte erfordert eine übersichtliche und systematische Zusammenstellung. Eine Werkzeugunterstützung von Beginn an ist deshalb geboten. Idealerweise erfolgt eine solche Unterstützung mittels eines Repository (z.B. Ortner, 2005, S. 187-235). In einem Repository können Aussagen gespeichert und semantisch verbunden werden. Die Entwicklungsergebnisse werden

damit sukzessive synchronisiert und stehen im Verlauf der Entwicklung systematisiert für die Rückführung zum Gebrauch, z.B. zur Stützung der Nutzer, zur Verfügung.

Das Handeln im Anwendungsbereich wird im Zuge der Sammlung von Aussagen durchleuchtet, verstanden und in Form von Aussagen dokumentiert. Eindeutig nicht relevante Sachverhalte werden nicht erfasst. Der Anwendungsbereich wird in der Sprache der Anwender beschrieben. Die Aufnahme von Aussagen kommt einer Bildung von Sprachartefakten (Sätzen) gleich. In Summe stellen die gesammelten Aussagen den Anwendungsbereich dar und können als Gesamtheit wiederum als Sprachartefakt im Sinne einer Aggregation von Sätzen verstanden werden. Die Erhebung von Aussagen ist eine notwendige Vorstufe der Schemabildung. Aus Sicht der sprachbasierten Informatik heißt das, die Aussagen bedürfen einer systematischen Vereinfachung. Diese erfolgt durch Bildung von Elementarsätzen. Der Vorgang wird Normierung genannt. Die Normierung geht einher mit einer Ableitung von Schemata aus den Anwenderaussagen. Damit dies gelingen kann, sind vorab Begriffe zu klären sowie *Begriffsdefekte* zu identifizieren und auszuräumen.

3.2.3 Klärung und Rekonstruktion von Fachbegriffen

3.2.3.1 Einführung und Überblick

Fachbegriffe zu klären bedeutet Sprache und Sprachartefakte zu rekonstruieren. Parallel dazu werden Begriffsklärungen vorgenommen. Sprachartefakte sind Beschreibungen von Anwendungsbereichen. Damit werden überprüfbare Konstruktionen von Grundbegriffen und Beziehungen geschaffen, welche nachvollziehbar und zirkelfrei einer rationalen Verwendung für den weiteren Entwicklungsprozess zur Verfügung stehen. Für die Informatik kann diese Bildung einer gemeinsamen Fachsprache auch als Protovorgehensweise (Hartmann, E. A., 2005) verstanden werden. Für die sprachbasierte Informatik ist sie unabdingbarer Bestandteil der Entwicklungsarbeiten. Die Fundierung und Geltung der Beschreibungen werden damit verstärkt. Die erarbeitete gemeinsame Fachsprache eines Anwendungsbereichs gilt für diesen als Inhaltsstandard und führt zu einer Repräsentation auf der Sprachebene vom Standpunkt des Anwendungsbereichs (vgl. Abb. 26, S. 101).

Sind Aussagen nicht klar in ihrer Geltung, so ist dies meist auf unklare Begriffe zurückzuführen. Unklare Begriffe sind im jeweiligen Kontext nicht klar zugeordnet und definiert. Eine Klärung und eindeutige Definition solcher Begriffe wird durch die sprachliche Rekonstruktion im engeren Sinn erreicht. Diese erfolgt anhand der möglichen Begriffsrepräsentationen[20] (Lehmann, 1999, S. 138) in Verbindung mit einem

20 Begriffsrepräsentationen sind Gegenstände oder Benennungen.

Begriffsmodell (Abb. 8, S. 33). Zur *Begriffsklärung* werden Lexika und Begriffsstandards (z.B. branchenspezifische Nomenklaturen) herangezogen. Ist dies nicht ausreichend, werden hermeneutische Verfahren eingesetzt. Hermeneutik heißt, das Begriffssystem eines Anwendungsbereichs zu ergründen und auf dieser Grundlage ein Gesamtverständnis aufzubauen (Apel, 1976). Diese Verfahren anzuwenden gehört zu den Aufgaben der Entwickler und setzt ein Vorverständnis dieser Personen für das Fachgebiet des Anwendungsbereichs voraus (Bultmann, 1988). Verfahren, die zu dieser Klasse gezählt werden, sind z.B. das Erstellen und die Auswertung von Fragebögen, die Dokumentenanalyse, wie z.B. das Auswerten von Organisationsunterlagen und Protokollsätzen, und das Studieren von Fachliteratur. Die Klärung geschieht bei hermeneutischen Verfahren durch die Analyse der verwendeten Benennungen.

Über exemplarische Erhebungsmethoden wird primär die Verwendungsweise eines Begriffs rekonstruiert. Solche Methoden werden genutzt, um die Extension eines Begriffs mittels Gegenständen des Anwendungsbereichs zu veranschaulichen. Zur Klasse dieser Methoden gehören beispielsweise die Beobachtung, die Mitarbeit bzw. die teilnehmende Beobachtung, das Experiment, die Präsentation von Gegenständen, das Aufzählen von Beispielen und Gegenbeispielen und das Geschichten erzählen.

Im Übrigen werden kommunikative Rekonstruktionsverfahren eingesetzt. Bei diesen Verfahren handelt es sich um eine vorwiegend kommunikative, meist ohne anschauliche Extension praktizierte Form der Erarbeitung von Begriffsdefinitionen (Lehmann, 1999). Zu dieser Kategorie von Erhebungsmethoden zählen z.B. Besprechung, Brainstorming, Interview, Seminar, Workshop und Tutorium. Es sind populäre Erhebungstechniken, bei denen im Gespräch die Benutzer aktiv miteinbezogen werden.

Die Begriffsklärung ist ein systematisches Verfahren zur Feststellung von möglichst eindeutigen Begriffsdefinitionen. Begriffsdefinitionen sind nicht starr. Das Verhalten von Begriffsdefinitionen in der Anwendung ist zu beobachten. Der Benutzer sollte nicht den Eindruck bekommen, dass eine einmal festgelegt Definition nicht mehr geändert werden kann. Dies würde sich negativ auf die *Akzeptanz* von Begriffsklärungen generell auswirken. Bei Bedarf sind Festlegungen gestützt durch Anwender und Anwendungsbereich wieder und wieder zu bestätigen (Holzer, 2005, S. 38-39). Diese Geltungssicherung auf Begriffsebene führt zu einer Stabilisierung des Begriffssystems und der Unternehmensfachsprache.

Arbeiten verschiedene Disziplinen zusammen, ist ein gemeinsames Begriffsverständnis nicht von vornherein gegeben. Es existiert ein gewisser Begriffsrelativismus (Holzer, 2005, S. 21), d.h. Begriffe zwischen verschiedenen Systemen, Anwendungsbereichen und Disziplinen sind nicht immer 1:1 und direkt zu übersetzen. Über die Begriffsklärung hinaus notwendige Regelungen in diesem Zusammenhang können in Regelwerken wie z.B. in einer SOA-*Governance* institutionalisiert und über Unter-

nehmensarchitekturen mit anderen Regelwerken verbunden werden. Mit den angeführten Maßnahmen kann der Begriffsrelativismus weitgehend egalisiert werden.

> Beispiel:
>
> Im Beispiel der Auftragsbearbeitung ist ein aus fachlicher Sicht „nachvollziehbarer Belegfluss“ gefordert. Damit ist gemeint, dass zu jeder Zeit der Status eines Belegs und damit verbunden ggf. der Ort der Verfügbarkeit des physischen Dokuments ermittelt werden kann.
>
> Nachvollziehbarkeit bedeutet in einem anderen Kontext z.B. das Verstehen von etwas, ohne dies formal belegen zu müssen. Auch ein „sich in etwas Hineindenken“ wird in der Alltagssprache als „nachvollziehen“ bezeichnet.
>
> Für die Anwendungsentwicklung sind solche Begriffe als vage zu bezeichnen und entsprechend zu klären. Vage Bezeichnungen stellen einen Begriffsdefekt dar und bedürfen einer Klärung.

Unabhängig davon, welche Methoden zur Begriffsklärung eingesetzt werden, lässt sich unter Zuhilfenahme eines Begriffsmodells (Abb. 8, S. 33) der Grad der Vollständigkeit der Klärung feststellen. Ein Begriff gilt dann als geklärt, wenn alle angesprochenen Aspekte des Begriffs im Hinblick auf Begriffsumfang und Begriffsinhalt, das sind Benennung, Intensionen und Extensionen, festgelegt sind und er von *Begriffsdefekten* bereinigt ist. Für diesen Fall existiert eine möglichst eindeutige Beziehung zwischen Begriff und Benennung. Eine vollständige Klärung ist nicht immer möglich. In solchen Fällen hilft das Begriffsmodell, für Transparenz über verbleibende Begriffsdefekte zu sorgen. Der Aufwand für die Klärung von Begriffen und die Erfassung der Ergebnisse in einem Wörterbuch (Lexikon, Glossar – unterstützt durch ein Repository) darf nicht unterschätzt werden.

Die hier grundlegend eingeführte und nachfolgend anwendungsbezogen vertiefte, systematische *Begriffsklärung* stellt ein Koordinationssystem dar, das u.a. dem angesprochenen Begriffsrelativismus entgegenwirkt. Sie schafft Eindeutigkeit von Kennzeichnungen für einen bestimmten Anwendungsbereich. Die zugehörigen Tätigkeiten können auch als semantische Modellierung oder semantische Normierung bezeichnet werden.

3.2.3.2 Definition von Begriffen

Jede Definition von Begriffen spielt sich im Rahmen des vorgestellten Begriffsmodells ab (vgl. Abb. 8, S. 33). Ziel einer Definition ist es, den zu definierenden Begriff eindeutig zu bestimmen und gegen andere Begriffe abzugrenzen (DIN 2330, 1993; Lehmann, 1999, S. 220). Vom Entwickler und Anwender werden Definitionen zur Begriffsfest-

legung aber auch zur Überprüfung von Aussagen benötigt. Schienmann (1997, S. 124) zeigt für die Festlegung einer Definition drei Möglichkeiten auf:

1) **Exemplarische Einführung** – Sie dient dazu, unbekannte Benennungen in einem gemeinsamen Handlungskontext mit Fachexperten situationsabhängig und empraktisch einzuüben.
2) **Anwendung von Prädikatorenregeln** – Sie stellen eine formale Methode dar, um die Verwendungsweise von Benennungen (Prädikatoren) untereinander zu regulieren. Prädikatorenregeln (= sprachliche Normen) können aber auch zur expliziten Definition eingesetzt werden (Schienmann, 1997, S. 128-131).
3) **Explizite Definition** – Durch explizite Definition werden die Verwendungsweise und damit die *Bedeutung* von eingeführten und durch Prädikatorenregeln regulierten Termini fixiert.

   Beispiel:

   Definition des Prädikators „Benutzer“ (Schienmann, 1997, S. 135) im Kontext der Ausleihe einer Bibliothek:

   Ein Benutzer ist entweder eine Person oder eine Institution. Ein Benutzer hat eine Kennung, eine Anschrift, einen Benutzerstatus, wurde zu einem bestimmten Datum als Benutzer aufgenommen und kann Exemplare entleihen oder vormerken.

   Die Definition des Prädikators „Benutzer“ im Kontext der Entwicklung eines Warenwirtschaftssystems ist bereits wieder etwas anders gelagert:

   Ein Benutzer ist eine Person. Ein Benutzer nimmt eine oder mehrere Rollen im Warenwirtschaftssystem wahr. Ein Benutzer wurde zu einem bestimmten Datum angelegt.

Damit eine explizite Definition korrekt ist, müssen alle verwendeten Prädikatoren eingeführt sein und eventuell verwendete Variablen bestimmt sein. Für das Beispiel der expliziten Definition von „Benutzer“ im Kontext eines Warenwirtschaftssystems heißt das, dass alle verwendeten Prädikatoren von „Person“ bis „Rolle“ und „Warenwirtschaftssystem“ eingeführt sein müssen. Im genannten Beispiel ist der Prädikator „Rolle“ eine Variable, d.h. es sind alle Rollen zu nennen und zu definieren.

Als prinzipielle Möglichkeit der Geltungssicherung eines Begriffs ist auch eine exemplarische Einführung möglich. Unbekannte Wörter eines Fachgebiets werden an Beispielen und Gegenbeispielen, die im täglichen Leben eine Rolle spielen, eingeführt (Janich, 1996). Das kommt einer Definition aus dem praktischen Handeln heraus gleich und entspricht durchaus den Prinzipien des konstruktivistischen Vorgehens.

Durch das Erzählen von Geschichten und das Darstellen von Verwendungsbeispielen können Begriffe weiter veranschaulicht werden. Wenn eine solche exemplarische Einführung zu ungenau ist, können neue zusammengesetzte Begriffe eingeführt werden, wie z.B. „zitronengelb“ bzw. „Kundenlieferschein“, oder es werden Regeln zur Verknüpfung mit bereits exemplarisch eingeführten Termini angegeben, wie z.B. „Soda ist ein Salz“ (Lehmann, 1999, S. 222).

Definitionen können auch auf einem definitorischen Regress von bereits klar vordefinierten Prädikatoren gründen, ohne dass deren Verwendung noch einmal dargestellt werden muss. Ein Hinweis auf die Grundlage ist jedoch als benutzerfreundlich zu unterstützen. Anforderungen für die Formulierung von Definitionen wurden z.B. von Felber & Budin (1989), in DIN 2330 (1993), bei Lehmann (1999, S. 202 u. S. 223-224) und von Arntz, Picht & Mayer (2004) formuliert.

Für den sprachbasierten Ansatz der Anwendungsentwicklung bedeutet eine Klärung und Rekonstruktion von Fachbegriffen konkret, dass die Gebrauchssprache eines Anwendungsbereichs im Rahmen der Rekonstruktion punktuell hinterfragt und neu erschlossen wird. Die gesammelten und erarbeiteten Aussagen sind oft missverständlich, widersprüchlich und unvollständig, da die zugrunde liegenden Begriffe der natürlichen Sprache teilweise missverständlich und widersprüchlich verwendet werden (Ortner & Söllner, 1989). Das liegt einerseits an der Nicht-Eindeutigkeit der natürlichen Sprache an sich, und andererseits an spezifischen Defekten von Fachbegriffen.

Nicht-Eindeutigkeit führt zu Missverständnissen in der Kommunikation und weiter zu fehlerhaften oder nicht optimalen Handlungen und Abläufen. Sie geht durchwegs mit *Begriffsdefekten* einher**.** Solche Begriffsdefekte sind im Zuge der Rekonstruktion zu bereinigen. Im Zuge einer Begriffsdefinition werden Begriffsdefekte (synonym: *Begriffsanomalien*) dann augenscheinlich, wenn die Zuordnung zwischen Begriff und Benennung nicht eindeutig ist. Arten von Begriffsdefekten sind z.B. *Synonymie*, *Homonymie*, *Äquipollenz* sowie *Vagheiten* und nicht mehr korrekte Benennungen im Zeitverlauf. Welches die wichtigsten *Begriffsdefekte* sind, beschreiben u.a. Ortner & Söllner (1989) sowie Jablonski et al. (1999, S. 157f). Eine ausführliche Betrachtung von Begriffsdefekten, eine weitere Unterteilung sowie Hinweise zur Behandlung dieser Defekte finden sich bei Lehmann (1999, S. 146-163). Exemplarisch werden nachfolgend ausgewählte Begriffsdefekte und ihre Handhabung erläutert sowie klärungsbedürftige Begriffe beispielhaft identifiziert und behandelt.

3.2.3.3 Handhabung von Begriffsdefekten

Synonymie

Von totaler Synonymie spricht man, wenn eine bedeutungsgleiche Benennung vorliegt. Bei partieller Synonymie liegt eine Bedeutungsähnlichkeit der Benennung vor (Lehmann, 1999, S. 147-153). Mit bedeutungsähnlich ist gemeint, dass eine Benennung unter bestimmten Voraussetzungen und Einschränkungen für eine andere Benennung verwendet werden kann. Dies trifft zu, sofern die Intension der Begriffe hinreichend ähnlich ist. Das Kriterium für Synonymie ist dabei stets die Austauschbarkeit der fraglichen Benennungen unter Beibehaltung des gleichen Begriffskerns im Kontext. Nominalisierungen stellen ebenso Synonyme dar. Auch die Benutzung verschiedener Schreibweisen für eine Benennung kann als Synonymie aufgefasst werden. Partielle Synonyme treten häufiger auf als totale Synonyme.

Beispiel:

Totale Synonymie: Tierarzt / Veterinär
synonym / bedeutungsgleich
Fotografie / Photographie

Partielle Synonymie: sterben / entschlafen
Sonnabend / Samstag
zum Abschluss bringen / abschließen
Lieferung / Sendung
in Erwägung ziehen / erwägen

Die Verwendung von Synonymen stellt ein erhebliches Hindernis in der fachlichen Verständigung dar. Das Problem der Synonymie besteht darin, dass Anwender und Rechner hinter jeder anderen Benennung auch einen anderen Begriff vermuten, solange ihnen die entsprechende Synonymie-Beziehung unbekannt ist (Lehmann & Ortner, 1999). Ein Nichterkennen von Synonymen führt zwangsläufig zu Konsistenzproblemen (Lehmann, 1999, S. 201).

Begriffe mit bedeutungsgleicher Intension und Extension müssen in der Rekonstruktionsarbeit transparent gemacht und nach Möglichkeit ausgeräumt werden. Beim Auffinden von Synonymen sollte man sich auf eine Vorzugsbenennung einigen (De Antonellis & Demo, 1983; Klein, 1989), um die Sprachbenutzung zu vereinheitlichen. Die Auswahl einer geeigneten Benennung ist dennoch eine Herausforderung und sollte unter Beachtung nachfolgend aufgelisteter Forderungen vorgenommen werden (Felber & Budin, 1989; Lehmann, 1999, S. 202):

- **Sprachliche Richtigkeit** – Benennung den Sprachnormen entsprechend z.B. hinsichtlich Aussprechbarkeit, was wiederum die *Akzeptanz* eines Begriffs beeinflusst.
- **Eindeutigkeit** – Anstreben einer normierten *Terminologie* d.h. eine Benennung = ein Begriff.
- **Knappheit** – kurze, prägnante Benennungen fördern das Verstehen.
- **Aussicht auf Durchsetzung bei den Sprachbenutzern** – hier ist nochmals die Akzeptanzproblematik eingehend zu durchdenken.
- **Leichtfasslichkeit** – Das Lesen und Verstehen durch den Benutzer soll gewährleistet werden.
- **Einpassung in das gesamte sprachliche (begriffliche) Gefüge** – Die Wahl des Terminus soll bei Gegenüberstellung von semantisch benachbarten Termini klar und nachvollziehbar sein.
- **Treffende Erfassung des Sachverhalts** – Ein Sachverhalt soll mit der Benennung treffend erfasst sein und die Intension zweckmäßig ausdrücken d.h. sie soll weder zu kompliziert noch zu simplifiziert sein.

Bei der Wahl von Vorzugsbezeichnungen ist der Einschätzung der Akzeptanz der gewählten Benennung durch den Benutzer entsprechendes Gewicht zu verleihen. Dies gilt insbesondere dann, wenn synonyme Benennungen verschiedene Fachgebiete berühren wie dies beispielweise bei der Integration des Usability Engineering in die Anwendungsentwicklung der Fall ist. Zusätzlich zu der ohnehin zwingend erforderlichen Hinterlegung der erarbeiteten Vorzugsbezeichnung in einem Wörterbuch sind eng verwandte und ähnliche Begriffe bzw. andere Benennungen in unterschiedlichen Disziplinen als zusätzliche Einträge mit dem Verweis auf die Vorzugsbezeichnung in das Wörterbuch aufzunehmen. Liegen für Begriffe genormte Benennungen vor, wie z.B. für chemische Formeln, so sind auf jeden Fall die genormten Benennungen vorzuziehen. Partiell synonyme Benennungen aus unterschiedlichen Fachgebieten sind zu vermeiden.

Synonyme Bezeichnungen können zulässig sein, wenn sie bekannt und transparent gemacht sind und somit kontrolliert verwendet werden können. Es ist dann z.B. eine systeminterne automatische Übersetzung in die vorab festgelegte Vorzugsbezeichnung denkbar.

Beispiel:

Es wurde identifiziert, dass die Begriffe „Lieferschein“ und „Auftrag“ und „Bestellung“ in Richtung des Kunden synonym verwendet werden. Diese Verwendung ist „historisch gewachsen“ und resultierte daraus, dass in einem Altsystem nur ein Dokument für den Kunden angelegt wurde, d.h. es gab keine Bestellung und keinen Auftrag, sondern es wurde umgehend ein Lieferschein erstellt. Dieser repräsentierte verschiedene Auftragsstati im Zeitverlauf. Die Handhabung wurde als keinesfalls repräsentativ erkannt. Neue Mitarbeiter hatten Probleme mit dieser Begriffshandhabung. Die rechtliche Klärung eines Kundenvorgangs macht auf dieser Basis Schwierigkeiten.

Behandlung:

Für die unterschiedlichen Stati eines Kundenauftrages wurden explizit Dokumente definiert, die klar einem Kundenvorgang zugeordnet werden können. Es handelt sich dabei um das Dokument Auftrag und das Dokument Lieferschein. Eine Bestellung als Dokument wird auf Kundenseite nicht generiert. Bei einer Bestellung durch den Kunden erfolgt umgehend die Erfassung eines Auftrages. Der Auftrag kann unterschiedliche Stati haben, ebenso der Lieferschein.

Lexikalische Ambiguität (z.B. Homonymie)

Ambiguität oder Mehrdeutigkeit eines Begriffs bedeutet, dass einem Sprachartefakt (z.B. einem Wort oder einem Satz) mehrere Interpretationen zugeordnet werden können. Dazu zählt vor allem die Homonymie. *Homonyme* beziehen sich auf gleichlautende jedoch bedeutungsverschiedene Wörter. Es handelt sich um einen Begriff, für den im Wörterbuch bereits ein gleichlautender Eintrag vorliegt, der diskutierte Begriff wird jedoch anders verwendet, als dies im Wörterbuch definiert ist.

Beispiel:

Kiefer (Teil des Kopfes / Nadelbaumart)
Lieferschein (Begleitdokument Warenausgänge / Begleitdokument für Wareneingänge)
Bank (Geldinstitut / Sitzmöbel)
Fuß (Teil des Beines / Sockel / Längenmaß)

Lexikalische Ambiguitäten müssen aufgelöst werden bzw. müssen die Begriffe eindeutig unterscheidbar gemacht werden. Dies geschieht, indem die betreffende Benennung für einen bestimmten Begriff reserviert wird und für alle anderen homonymen Begriffe neue Benennungen eingeführt werden. Die konkrete Lösung im Anwendungsbereich sollte von einer Prognose hinsichtlich der besseren Akzeptanz der

Sprachbenutzer abhängen (De Antonellis & Demo, 1983; Lehmann, 1999). Weiters können mehrdeutige Benennungen durch Zusätze unterscheidbar gemacht werden. Beispielsweise kann statt der Verwendung von „Ort", von „Wohnort" oder „Geburtsort" gesprochen werden (Ortner & Söllner, 1989). Die Auflösung von Homonymen, insbesondere im Bereich der Intension, kann derzeit kaum durch einen Rechner unterstützt werden (Zimmermann, 1997, S. 390) und stellt daher eine Aufgabe dar, die von Menschen zu lösen ist.

Beispiel:

Nach Erstellung oder bereits im Zuge der Aussagensammlung wurden für die exemplarisch aufgelisteten Aussagen zur Entwicklung eines Warenwirtschaftssystems u.a. nachfolgend aufgelistete Begriffe zur Klärung identifiziert. Einschlägige Fachbegriffe waren in dem Kontext, aus dem das Beispiel, stammt über eine Nomenklatur (= Inhaltsstandard) klar definiert:

Lieferschein, Lagerbestand, Rabatte, Preisänderung, Auftrag, nachvollziehbarer Belegfluss, …

Erläuterungen zum Begriff Lieferschein:

Für diesen Begriff ist aus dem Kontext der geführten Interviews hervorgegangen, dass der Begriff Lieferschein nicht nur homonym, sondern auch in unterschiedlicher Bedeutung über den Zeitverlauf verwendet wird. So wird ein erfasster Kundenauftrag als Lieferschein bezeichnet. Dies führte im weiteren Verlauf der Auftragsabwicklung zu Statusproblemen in der Abwicklung, da nicht alle möglichen Auftragsstati mit der Belegart „Lieferschein" dargestellt werden konnten, und verursachte punktuell erheblichen Mehraufwand. Weiters wurde das Begleitpapier für Wareneingänge ebenfalls als Lieferschein bezeichnet.

Defekt: Es liegt Homonymie in Verbindung mit nicht korrekter Benennung im Zeitverlauf vor.

Behandlung: Zur Beseitigung der Homonymie wurde festgelegt, dass kontextabhängig die Vorzugsbezeichnungen Kundenlieferschein bzw. Lieferantenlieferschein eingeführt und verwendet werden. Zur Behebung der korrekten Abbildung der Auftragsstati in Verbindung mit relevanten Dokumenten wurde das Dokument Auftrag mit verschiedenen zuordenbaren Stati (erfasst, in Bearbeitung, erledigt, …) neu eingeführt. Der damit verbundene Kundenlieferschein wurde als Begleitdokument definiert und stellt selbst keinen Status in der Auftragsabwicklung dar.

Äquipollenzen

Eine *Äquipollenz* liegt vor, wenn zwei oder mehrere Begriffe die gleiche Extension haben, d.h. den Begriffen werden die gleichen Gegenstände zugeordnet. In ihrer Intension, d.h. in den wesentlichen Merkmalen, weichen sie aber voneinander ab. Anders ausgedrückt lässt sich sagen, dass das gleiche Objekt aus verschiedenen Perspektiven betrachtet wird.

Beispiel:

Einwohner / Konsument
Wiederkäuer / Paarzeher
Lagerbestand wertmäßig / Lagerbestand mengenmäßig

Äquipollenzen dienen vor allem der Komplexitätsreduktion von Begriffsebenen in einem spezifischen Anwendungszusammenhang. So kann ein Mensch z.B. als Angeklagter, Einwohner oder Mieter betrachtet werden, dem situativ und problemspezifisch jeweils nur einige wesentliche Merkmale eines Menschen zugeordnet werden.

Bei den angeführten Beispielen handelt es sich um sogenannte „echte Äquipollenzen". Im Rahmen einer Rekonstruktion in einem Anwendungsbereich sind diese ver-gleichsweise selten zu berücksichtigen. Treten sie jedoch auf, kann im Sinne einer benutzerfreundlichen Lösung auf die jeweils äquipollenten Begriffe hingewiesen werden. Eine Klärung, wie sie für Synonyme erfolgt, ist nicht erforderlich. Bei sogenannten „unechten Äquipollenzen" handelt es sich eigentlich um partielle Synonyme. Die Klärung hat entsprechend zu erfolgen.

Vagheiten

Bei der sogenannten *Vagheit* handelt es sich um vage oder falsche Benennungen eines Begriffes. Ob eine Vagheit vorliegt, ist oftmals nur subjektiv zu beurteilen. Für den Aufbau einer Unternehmensfachsprache ist Vagheit im Sinne von Randbereichsunschärfe besonders relevant. Begriffe haben meist einen eindeutigen Kernbereich, sind aber am Rand unscharf (Lyons, 1969). Eine vage Benennung bedeutet demnach, dass keine klare Trennung zwischen Begriffen erfolgt, sodass bei der Zuordnung von Gegenständen zu diesen Begriffen Unklarheiten und Unsicherheiten auftreten (Ortner & Söllner, 1989). Ein Begriff hat eine vage Extension, wenn nicht bei jedem möglichen Gegenstand entschieden werden kann, ob er zu dessen Extension gehört oder nicht. Im Unternehmen können z.B. Rollenbenennungen als vage aufgefasst werden, wenn diese nicht eindeutig beschrieben sind, und auf der anderen Seite Aufgabenbeschreibungen für Mitarbeiter vorliegen, über die wiederum nicht eindeutig auf dafür relevante Rollen Bezug genommen werden kann.

Beispiel:

Beim Begriff Berg ist unklar, ob ein Hügel noch zur Extension von „Berg“ gehört oder nicht.

Bürger versus Einwohner – hier ist unklar, ob die betreffende Person nur Einwohner einer Stadt ist oder bereits als Bürger jener gezählt werden könnte. Das kann zu Unklarheiten z.B. hinsichtlich der Frage der Wahlberechtigung führen, wenn diese am Begriff „Bürger“ festgemacht ist. Eine Präzisierung des Inhalts führt hierbei zu einer scharfen Trennung des Umfangs, z.B. Bindung der Wahlberechtigung an die Staatsangehörigkeit (Lehmann & Ortner, 1999).

In Verbindung mit dem Beispiel aus der Auftragsbearbeitung lag folgende Aussage vor: Der Sachbearbeiter Verkauf darf Aufträge für Risikokunden nur nach Rücksprache mit dem Niederlassungsleiter erfassen.

Nun galt es zu klären, was ein „Risikokunde“ ist. In der Diskussion stellte sich heraus, dass die Auffassungen darüber auseinandergingen. Nach eingehenden Diskussionen wurde dann ein stufenweises Vorgehen definiert. Kunden werden abhängig von der Höhe der aktuellen offenen Posten unterschiedlich behandelt. Mit der Festlegung von Kreditlimits in Stufen sollte eine eindeutige Abgrenzung für den Sachbearbeiter möglich sein.

Sind Vagheiten erkannt, lassen sie sich durch Präzisierung leicht klären. So kann beispielsweise für einen Berg angegeben werden, ab welcher Seehöhe von einem Berg gesprochen wird und bis zu welcher Seehöhe von einem Hügel gesprochen wird. In manchen Fällen ist die Angabe einer Maßeinheit möglich und vorteilhaft. Carnap (1962) plädiert in diesem Zusammenhang dafür, den unscharfen Begriff durch einen exakten Begriff zu ersetzen. Zu beachten ist dabei immer, dass Vagheit oft gleichzeitig mit anderen Begriffsdefekten auftritt, diese sind dann entsprechend zuerst zu klären. Oft besteht in der Folge keine Notwendigkeit mehr, den Begriff zu präzisieren. Um eine Präzisierung zu erreichen ist auch eine Nennung von Beispielen und Gegenbeispielen möglich.

Bedeutungswandel von Benennungen

Begriffe einer Sprache unterliegen einer ständigen Entwicklung. Sie ändern sich z.B. durch neue Erkenntnisse oder neue Technologien, oft aber allein durch den Gebrauch. Falsche Benennungen können festgestellt werden, wenn eine Abweichung der „suggerierten“ Wortbedeutung von der tatsächlichen Wortbedeutung auftritt.

Beispiel:

„Winker“ wandelte sich über die Jahre in „Blinker“ (Beides sind Begriffe für die Anzeige für Änderung der Fahrrichtung).

Es wird oft von „Glühlampe“ statt von „Glühbirne“ gesprochen.

Die Herausforderungen, die in diesem Zusammenhang auftreten können, sind äquivalent zu denen, die bei den partiellen Synonymien vorkommen. Im Zuge des Bedeutungswandels erscheint dann der verwendete Begriff oft als falscher Bezeichner.

Falsche Bezeichner
Benennungen, deren Bedeutung von der tatsächlich angenommenen Bedeutung abweichen, müssen ersetzt werden. Für veraltete Benennungen muss als Vorzugsbenennung unbedingt die nicht veraltete Benennung gewählt werden. Ziel ist hierbei die Aufgabe einer vertrauten Benennung zugunsten der Verwendung einer passenderen Benennung. Hier sollte auch kein Verweis auf die unlogische oder veraltete Benennung neben der Vorzugsbenennung vorhanden sein, damit diese nicht weiterhin verwendet wird. Es sollte vermieden werden, Benennungen für Begriffe zu verwenden, die andere als die tatsächlich vorliegende Bedeutung suggerieren (Lehmann, 1999, S. 209; Ortner, 1997). In der Diskussion mit den jeweiligen Begriffsbenutzern sind mögliche Akzeptanzprobleme zu beachten.

3.2.4 Ableitung von Schemata

3.2.4.1 Bildung von Elementarsätzen als Normierung

Mit der Ableitung von Schemata ist die Modifikation und Transformation von relevanten Aussagen zu Elementarsätzen auf Basis geklärter Begriffe unter Zuhilfenahme einer *Rationalen Grammatik* (Lorenzen, 2000; Wedekind et al., 2004b, S. 265) gemeint. Dieser Vorgang wird etwas einfacher auch als Normierung von Aussagen bezeichnet. Normierte Aussagen erleichtern eine Klassifizierung zur Überprüfung der Vollständigkeit der Aussagensammlung bzw. erlauben sie diese überhaupt erst. Mit der Schemabildung und der Prüfung der Aussagensammlung auf Vollständigkeit erfolgt der Übergang zur Modellierung. Eine klare Abgrenzung ist oft nicht möglich.

Die Normierung der vorliegenden Aussagen aus struktureller (d.h. grammatikalischer) Sicht ist der nächste Schritt der Rekonstruktionsarbeit. Den Aussagen wird in diesem Schritt eine einfache klare Grammatik zugrunde gelegt, eine *Rationale Grammatik.* Die Grundlagen dafür gehen auf die *logische Propädeutik* von Kamlah & Lorenzen (1967) zurück. Durch die Verwendung dieser Grammatik entsteht eine sehr einfache, reglementierte Sprache, auch Normsprache genannt. Das Bestreben in der Schemabildung geht dahin, die Sätze aus der Aussagensammlung möglichst so zu verein-

fachen und zu transformieren, dass sie als Elementarsätze im Sinne einer Rationalen Grammatik gelten (Wedekind et al., 2004b, S. 266). Die *Rationale Grammatik* stellt uns nur zwei Ausdrucksschemata zur Verfügung, dies sind einerseits eine Aufforderung (Request) und andererseits eine Aussage (*Proposition*). Sprachtheoretisch lassen sich diese Ausdrucksschemata auf Basis der verwendeten Verben auf bestimmte Sprechakte bzw. Handlungsarten zurückführen (vgl. Searle, 1976; Abschnitt 5.1, S. 185ff).

Praktisch geht es einerseits darum, die Eindeutigkeit einer Aussage zu erreichen, andererseits ist festzustellen, ob es sich um eine allgemeine Aussage oder eine singuläre Aussage handelt. Schemata sind allgemeine Aussagen und damit universelle Beschreibungen von Gegenständen und Geschehnissen. Ausprägungen sind singuläre Aussagen und stellen mögliche Konkretisierungen von Schemata dar. Sie können auch beispielhaft zur Erläuterung von Schemata dienen, z.B. um ein Schema für einen Anwender besser verstehbar zu machen.

> Beispiel:
>
> Allgemeine Aussage: „Kunden, die eine Bestellung machen, erhalten eine Auftragsbestätigung."
>
> Singuläre Aussagen: „Firma Oberhuber erhält eine Auftragsbestätigung für die Bestellung vom 31.01.2008. Firma Dorfmeister bestellt 300 m² Dachziegel, lieferbar in 6 Monaten. Herr Müller holt ab Lager 20 m² Dämmplatten und bezahlt diese sofort."
>
> Offen bleibt in diesem Beispiel, ob jeder Kunde, der eine Bestellung macht, eine Auftragsbestätigung erhält. Nach diesem Schema könnte es auch Kunden geben, die keine Auftragsbestätigung erhalten. Dies könnte z.B. den Kunden Müller betreffen, der seine Ware an der Theke in Auftrag gibt, gleich mitnimmt und sofort bezahlt.

Im Zuge der Schemabildung werden die gesammelten Aussagen auf Basis nunmehr geklärter Begriffe genauer untersucht, um allgemeine Aussagen zu identifizieren. Menschen sprechen und denken in Schemata und/oder in Ausprägungen. Sie treffen diese Unterscheidung beim Sprechen und Denken in der Regel nicht oder zumindest nicht bewusst. Ihre Aussagen sind eine Repräsentation jenes Wissens, das sie im Kopf haben. Diese Schemata wurden bereits als mentale Modelle eingeführt (vgl. Abschnitt 2.4.5, S. 80ff). Sie sind subjektives Orientierungs- und Verrichtungswissen und begrenzen gleichzeitig die Sprachkompetenz des jeweiligen Menschen. Diese Auffassung geht zurück auf den amerikanischen Linguisten N. Chomsky (geb. 1928) (Wedekind et al., 2004a, S. 176). Chomsky verwendete dafür nicht das Begriffspaar Schema und Ausprägung, sondern „Competence" für Schema und „Performance" für Aus-

prägung. Ortner verwendete diese Begriffe in der Spracharchitektur (Abb. 1, S. 18) in der sprachbasierten Informatik. Die Sprachkompetenz steht für die Fähigkeit des Sprachbenutzers, den inhaltlichen, ethischen und fachlichen Aufbau der gewählten Sprache zu verstehen, und gründet auf seinen mentalen Schemata. In der Sprachperformanz zeigt sich die Fähigkeit des Sprachbenutzers zum Einsatz seiner mentalen Schemata (vgl. Abschnitt 2.1.2, S. 17ff).

Im Zuge der Normierung werden die Aussagen zum Anwendungsbereich nach Satzbauplänen in einfache Sätze (Elementarsätze) einer durch rationale Grammatik und Gegenstandseinteilung (vgl. Abb. 9, S. 34) reglementierten Sprache transformiert. Die gemeinsame Form der Gegenstandseinteilung für die Wortarten des objektorientierten Entwurfs macht klar, dass hier bereits die Grundsteinlegung des Übergangs zur Klassifizierung und damit das Verlassen der Methodenneutralität erfolgt, dies entspricht dem Übergang vom anwendersprachlichen zum entwicklersprachlichen Kontext (vgl. Abb. 26, S. 101). Die Satzbaupläne repräsentieren den Dokumentationsbereich „Grammatik“. Eine detaillierte Beschreibung von Satzbauplänen, wie sie hier verwendet werden, findet sich bei Ortner (1997, S. 86ff). Die Gegenstandseinteilung dient als grundlegende Struktur für den Satzbau auf Basis geklärter Begriffe. In der exemplarischen Rekonstruktion finden einfache Satzbauformen folgender Art Verwendung:

Beispiel:

DING – BEZIEHUNG – BESCHAFFENHEIT.

Dachziegel ist aus Ton.
Auftrag hat Status.
Auftrag hat Auftragsnummer.
Kunde hat Kundennummer.

DING – BEZIEHUNG – DING.

Auftragsbestätigung enthält Lieferdatum.
Auftragsbestätigung dokumentiert Bestellung.

Die Sätze einer rationalen Sprache (Normsprache oder reglementierten Sprache) sind nicht fließend zu lesen und eignen sich nicht unmittelbar zur Kommunikation. Dazu sind sie viel zu primitiv. Sie dienen dazu, die komplexe Aussagensammlung schrittweise zu vereinfachen, was den Übergang zum Entwicklersprachlichen Kontext systematisiert und erleichtert. Zur besseren Verständlichkeit und zur Konkretisierung der Beziehungen zwischen Dingen und Beschaffenheiten (Merkmalen) werden die einfachen Sätze mit der Angabe der *Multiplizität* ergänzt. In etwas besser lesbarer Form und bereits mit Angabe der *Multiplizität* sehen die normierten Sätze folgendermaßen aus.

Beispiel:

Jeder Auftrag enthält ein Lieferdatum.
Eine Auftragsbestätigung dokumentiert eine oder mehrere Bestellungen.

Die Transformation von Aussagen beschränkt sich nicht auf die Umformung eines vorhandenen Satzes. Im Zuge der Transformation wird gleichzeitig darauf geachtet, dass der beschriebene Sachverhalt eindeutig wird. Es kommt vor, dass eine Transformation in einer oder mehreren Iterationen erfolgt, weil Eindeutigkeit nicht in einem Schritt erreicht werden kann. Dies hängt von der Komplexität der ursprünglichen Aussage ab. Die Zwischenergebnisse werden mit den Anwendern abgestimmt. Beispielhaft werden nachfolgend zwei Aussagen aus der exemplarischen Aussagensammlung zur Entwicklung eines Warenwirtschaftssystems (vgl. Abschnitt 3.2.2, S. 106) in sogenannte „normierte Aussagen“ transformiert. Es wird unterstellt, dass im Beispiel identifizierte Fachbegriffe vorab geklärt wurden. Die Nummerierung der Aussagen entspricht der Nummerierung in der Beispieltabelle (S. 97).

Beispiel:

Ursprüngliche Aussage Nr. 10: „Bestellt ein Kunde Ware, erhält er eine Auftragsbestätigung.“

Normierte Aussage Nr. 10:

10 a) Ein Kunde kann Ware bestellen.
10 b) Es gibt auch Kunden, die keine Ware bestellen.
10 c) Die Bestellung eines Kunden wird als Auftrag erfasst.
10 d) Für jeden Auftrag wird eine Auftragsbestätigung erstellt.
10 e) Es gibt auch Aufträge, zu denen keine Auftragsbestätigung erstellt wird.
10 f) Jede Auftragsbestätigung wird dem Kunden zugesendet.

Der Sachverhalt, der sich hinter Aussage 10 verbirgt, ist komplex. Eine erste Transformation bringt etwas mehr Transparenz. Jedoch war es in einem Schritt nicht möglich, alle Sachverhalte in Elementarsätze zu transformieren. So enthält beispielsweise Aussage 10f das Verb „zugesendet“. Dahinter verbergen sich unterschiedliche Möglichkeiten, wie der Kunde seine Auftragsbestätigung erhalten kann, z.B. mit einem Brief oder via E-Mail. Auch wie die Erstellung eines Auftrags erfolgt, ist hier nicht klar.

Ursprüngliche Aussage Nr. 11: Ein Kunden-Lieferschein enthält neben den Auftragspositionen auch eine Lieferadresse, ein Lieferdatum und eine Empfangsbestätigung durch den Kunden.

Normierte Aussage Nr. 11:

11 a) Ein Kunden-Lieferschein kann eine oder mehrere Auftragspositionen enthalten.
11 b) Ein Kunden-Lieferschein enthält genau eine Lieferadresse.
11 c) Ein Kunden-Lieferschein enthält genau ein Lieferdatum.
11 d) Der Warenempfang muss durch den Kunden auf dem Kunden-Lieferschein bestätigt werden.
11 e) Auf dem Kunden-Lieferschein ist Platz für diese Empfangsbestätigung vorgesehen.
11 f) Die Empfangsbestätigung besteht aus dem Datum des Empfangs der Ware und der Unterschrift des Kunden oder einer berechtigten Person.

Auch in diesem Fall ist eine Normierung in einem Schritt nicht möglich. Es sind in diesem Fall Begriffe aufgetaucht, die im Zuge einer Iteration mit dem Anwender geklärt werden müssen, z.B. wer eine berechtigte Person ist, die den Empfang der Ware bestätigen kann. Auch die Frage, ob eine solche Empfangsbestätigung auch elektronisch erfolgen kann, stellt sich in diesem Zusammenhang. Darf die Ware beim Kunden oder auf einer angewiesenen Baustelle auch abgeladen werden, wenn niemand den Empfang bestätigen kann?

Werden Aussagen transformiert, die Gegenstände und deren Eigenschaften beschreiben, so ist die Transformation einer Aussage in Elementarsätze noch relativ überschaubar. Sie kann als Vorstufe der Datenmodellierung verstanden werden. Eine sofortige Berücksichtigung der *Multiplizität* ist speditiv und dient der Klarheit der gewonnenen Schemata. Die Transformation von Aussagen, die Geschehnisse beschreiben, ist nicht trivial und erfolgt meist in mehreren Iterationen. Diese Transformation kommt einer Vorstufe der Prozessmodellierung gleich.

Im Zuge der Bildung von Elementarsätzen wird klar, bei welchen Aussagen es sich um Schemata, d.h. um allgemein gültige Aussagen, handelt. Allgemeine Aussagen gewinnen mit der Transformation zu Elementarsätzen an Eindeutigkeit. Für singuläre Aussagen ist das nicht der Fall. Elementarsätze allgemeiner Aussagen sind eindeutig und daher immer validierbar.

Beispiel:

Ursprüngliche Aussagen:

Kunden erhalten eine Kundennummer. Kunde Müller erhält die Nummer 4711.

Normierte Aussagen:

Kunde hat Kundennummer.
Müller hat 4711.

„Kunde hat Kundennummer" ist eine allgemeine Aussage und eindeutig. Eine Verbesserung der Aussage kann durch die Ergänzung der Multiplizität erreicht werden. Die Aussage „Müller hat 4711" kann ohne den Kontext der allgemeinen Aussage nicht zugeordnet werden. Es ist nicht klar, welche Rolle „Müller" hat, ob er ein Kunde, ein Lieferant oder ein Mitarbeiter ist. Welche Bedeutung hat die Zahl 4711? Es kann mit der Normierung keine Eindeutigkeit für diese Aussage erreicht werden. Es handelt sich um eine singuläre Aussage.

Anwender werden idealerweise in einer weiteren Iteration erneut mit jenen, nunmehr normierten Aussagen, die sich aus den gemeinsamen Gesprächen im Zuge der bisherigen Entwicklungsarbeit ergeben haben, konfrontiert. Der Entwickler fordert den Anwender auf, die Aussagen hinsichtlich ihrer Gültigkeit und ihres Geltungsbereichs zu überdenken und gegebenenfalls zu bestätigen. Bestätigt der Anwender die Gültigkeit nicht oder kann der Geltungsbereich nicht festgelegt werden, ist die Aussage entweder nicht relevant oder sie ist anzupassen. Der Rekonstruktionszyklus ist für diese Aussage in Abstimmung mit verbundenen Aussagen erneut zu durchlaufen. Diese Iteration entspricht einer begleitenden Validierung der Entwicklungsergebnisse in Form einer Geltungssicherung. Mit einer eher technischen *Terminologie* ausgedrückt, könnte man dies als einen Schritt der Qualitätssicherung der Entwicklungsarbeit bezeichnen.

Unter den gesammelten Aussagen finden sich in der Regel nicht nur Aussagen im Sinne von Beschreibungen eines Zustandes. Es sind auch Aussagen darunter, die als Aufforderung im Sinne einer *Forderung* (Request) an das zu entwickelnde Anwendungssystem zu verstehen sind. In diesem Fall ist im Zuge der Schemabildung gemeinsam zu klären, welche konkreten Anforderungen aus diesen Forderungen resultieren. Sind Aussagen nicht auf Elementarsätze zu reduzieren, so ist in der Formulierung darauf zu achten, dass sie zumindest als *validierbare Anforderung* gelten können (Rupp, 2007).

Durch das beschriebene Vorgehen gestaltet sich der Entwicklungsprozess in den frühen Phasen vorerst aufwendiger. Das Entwicklungsergebnis ist jedoch in seiner Gültigkeit und im Geltungsbereich vom Anwender bestätigt, wodurch von einer besseren *Akzeptanz* der Entwicklungsergebnisse im Anwendungsbereich ausgegangen werden kann (z.B. durch frühzeitig wahrgenommene Zweckmäßigkeit oder Verbesserung der Einstellung zur Technologienutzung, vgl. Tab. 9, S. 197, bzw. Tab. 11, S. 199). Sowohl die Klärung von Begriffen als auch die entwicklungsrelevante Darstellung des Anwendungsbereichs mittels Aussagen in normierter natürlicher Sprache entsprechen einer semantischen Integration. Die sprachliche und fachliche Effizienz steht hier im Vordergrund. Diese Ableitung gelingt nur dann, wenn die Benutzer explizit und systematisch in diesen Prozess einbezogen werden.

Menschen können sich nicht in einer rationalen Sprache unterhalten, dazu ist sie viel zu primitiv. Aufgrund dieser Einfachheit, sind sie jedoch in der Lage, diese Sprache zu verstehen. Der Entwickler ist gefordert, in Elementarsätzen der rationalen Grammatik zu denken, um die schrittweise und vollständige Transformation von Aussagen in natürlicher Sprache in eine normierte Sprache zu gewährleisten. Sind alle relevanten Aussagen normiert und deren Geltung bestätigt, könnten wir bereits vom Vorliegen eines Fachentwurfs für das zu entwickelnde Anwendungssystem sprechen. Bei konsequenter Einhaltung der empfohlenen Vorgehensweise ist dieser Entwurf auch durch Benutzer und Experten des Anwendungsbereichs bestätigt. Zuvor ist allerdings noch zu überprüfen, ob die Darstellungen vollständig sind, sofern dies nicht bereits begleitend erfolgt ist.

3.2.4.2 Prüfung der Aussagensammlung auf Vollständigkeit

Die Überprüfung des Aussagenbestandes auf Vollständigkeit erfolgt mittels einer Klassifizierung der Aussagen. Die erarbeiteten Beschreibungen liegen als normierte Aussagen vor. Mit Hilfe eines Klassifikationsrahmens (Abb. 28, Ortner, 2005, S. 69) werden die Aussagen sortiert. Mit dem Klassifikationsrahmen lässt sich die Überdeckung aller Gegenstände der Ablauf- und Aufbauorganisation eines Unternehmens mit den entwicklungsrelevanten Aussagen überprüfen. Die Überprüfung kann entwicklungsbegleitend bereits vor oder auch im Zuge der Normierung und der Begriffsklärung durchgeführt werden. Die gezielte Erfassung von Aussagen kann ebenfalls mit diesem Rahmen gesteuert werden.

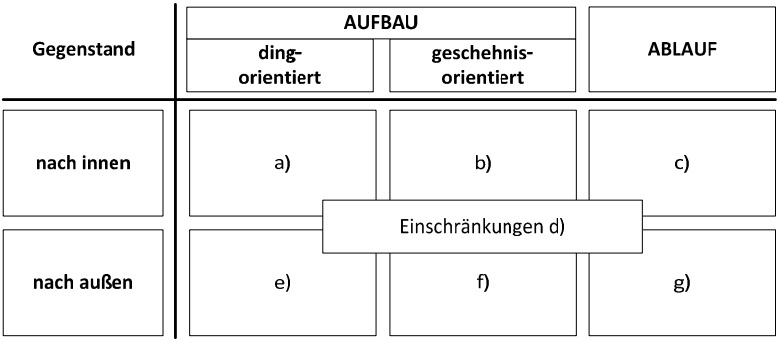

Abb. 28: Klassifikationsrahmen für entwicklungsrelevante Aussagen

Aus der Ermittlung von relevanten Aussagen zur Entwicklung eines Warenwirtschaftssystems können folgende Beispiele als Ergebnisse für die einzelnen Felder des Klassifikationsrahmens angeführt werden.

Beispiel für die Zuordnung teilweise noch nicht normierter Aussagen:

a) Ein Kunde hat eine Kundennummer. Ein Auftrag hat eine Auftragsnummer.
b) Ein Kunde wird angelegt. Ein Auftrag wird erstellt.
c) Die Erstellung eines Auftrages führt zu einem noch nicht erledigten Auftrag.
d) Die noch nicht erledigten Aufträge eines Kunden dürfen zusammen mit den erledigten und noch nicht bezahlten Aufträgen des Kunden das festgelegte Kreditlimit des Kunden nicht übersteigen.
e) Ein Auftrag ist einem Kunden zugeordnet.
f) Die Erledigung des Auftrags steht in Verbindung mit der Erstellung und Verbuchung einer Ausgangsrechnung.
g) Der Auftrag wird erledigt, indem die Ware auf den LKW geladen und zum Kunden gebracht wird.

Ist ein relevanter Ablauf aus dem Anwendungsbereich nicht in jedem Feld des Klassifikationsrahmens repräsentiert, so kann das als ein Hinweis auf Unvollständigkeit der Aussagensammlung gewertet werden. Sind die Aussagen den Entwicklungszweck betreffend vollständig, so liegt ein durch Rekonstruktion erstellter *methodenneutraler* Fachentwurf vor.

Das Vorgehen der (sprachkritischen) Rekonstruktion, erstmals von Ortner (1993) skizziert und erläutert, wurde permanent weiterentwickelt und auch formal-analytisch detailliert beschrieben von Schienmann (1997), Lehmann (1999) und Ortner (2005).

Die Charakterisierung der Methode in diesem Rahmen erfolgte konstruktiv und anwendungsorientiert, aufbauend auf den erwähnten wissenschaftlich fundierten Grundlegungen von Ortner, Schienmann und Lehmann. Zur Reduktion der Komplexität und als Ansatz zur Umsetzung der grafischen Darstellung in Abb. 27 (S. 105) in einen idealtypischen Ablauf der Rekonstruktion wird die Vorgehensweise nachfolgend in einen 10-Stufen-Ablauf aufgelöst. Eingebaute Iterationen verdeutlichen die Vernetzung, sie sind nur bei Bedarf zu durchlaufen (Eller, 2004):

Stufe 1: Sammlung relevanter, noch nicht normierter Aussagen – eine klare Abgrenzung zu Voruntersuchung ist nicht möglich.

Stufe 2: Auswahl der Einbeziehung von bereits vorhandenen Terminologien und Inhaltsstandards sowie Auswahl einer rationalen Grammatik.

Stufe 3: Bearbeitung der noch nicht normierten Aussagensammlung hinsichtlich der Normierung.

Stufe 4: Identifikation von Lücken in verwendeten Inhaltsstandards und Klärung bzw. Rekonstruktion der Begriffe.

Stufe 5: Darauf aufbauend Modifikation des Inhaltsstandards (Sprachstandards) und Iteration der Stufen 2-4.

Stufe 6: Identifikation von Lücken über Aussagen aus dem Anwendungsbereich.

Stufe 7: Falls erforderlich Schließung der Wissenslücken durch zusätzliche Besprechungen zu Handlungen und Geschehnissen im Anwendungsbereich (dies entspricht einer vereinfachten, klar abgegrenzten Iteration der Voruntersuchung).

Stufe 8: Für neu hinzugekommene bzw. adaptierte Aussagen (aus Stufe 7) zum Anwendungsbereich nochmaliges Durchlaufen der Stufen 2 bis 7.

Stufe 9: Normierung der problemorientierten Aussagensammlung und Bildung von Schemata – allgemeine Aussagen).

Stufe 10: Klassifikation der Aussagen zur Prüfung der Vollständigkeit.

Zusammenfassend und angelehnt an die *logische Propädeutik* kann man sagen, dass damit die anwendersprachliche Erschließung des Anwendungsbereiches durch Eigennamen und Prädikatoren abgeschlossen ist. Sie ist einerseits erfolgt durch das Reden als Gespräch zwischen Anwendern, Entwicklern und anderen am Entwicklungsprojekt beteiligten Menschen. Andererseits ist sie durch das Reden als inneres Reden des Einzelnen (= Denken) geprägt. Die Ergebnisse des „Redens“ sind Schemata in der Sprache des Anwendungsbereichs. Ihre Vollständigkeit wird gewährleistet über die gezeigte Klassifizierung. Die Arbeiten der Rekonstruktion sind schwerpunktmäßig in der Anwenderorganisation anzusiedeln.

Der Übergang zur Modellierung und damit der Übergang auf die Sprachebene vom Standpunkt der Informatik erfolgt mittels einer weiteren Klassifizierung. Diese Klassifikation bedeutet das Verlassen der Methodenneutralität und stellt gleichzeitig den Übergang zur Modellierung dar (vgl. Abb. 26, S. 101).

3.2.5 Integration des Usability Engineering in der Rekonstruktion auf der Sprachebene vom Standpunkt des Anwendungsbereichs

Sowohl die Entstehung von Wissen als auch die Bewährung von Wissen in der sozialen Wirklichkeit sind Sprachvorgänge (Sprachhandlungen). Nach Nicolini (2001, S. 20-21) ist die Sprache dabei Quelle und Wurzel der Erkenntnis, Erfahrung und Kommunikation für das Forschungsgeschehen. Diese Auffassung von Sprachhandlungen wird mit dem *Paradigma* der sprachkritischen Entwicklung auf das Geschehen in der Anwendungsentwicklung übertragen. Der Einsatz sowohl der gesprochenen als auch der geschriebenen Sprache soll von Anfang an auf Deutlichkeit, Verständigung und Dringlichkeit ausgerichtet sein. Dies betrifft die Kommunikation zwischen allen am Entwicklungsprojekt Beteiligten über den gesamten Entwicklungsprozess. Grundlage hierfür ist das Verständnis und die Anwendung von Sprache als System (Abb. 1, S. 18). Das schlägt sich in Sprachkompetenz nieder und äußert sich in der Sprachperformanz der Beteiligten. Die Entwicklungsarbeiten werden insbesondere in der Rekonstruktion durch gute Sprache (Sprachkompetenz und Sprachperformanz) erfolgreich. Die gemeinsame Sprache bildet einen verlässlichen Ort in der Ungewissheit und Unsicherheit von interdisziplinärer Tätigkeit.

Sprache ist der (semantischen) Integration immanent (Ortner, 2003b, S. 3). Das heißt, jede Integration erfolgt über Sprache bzw. über sprachliche Repräsentationen von etwas. Mittels des materialsprachlichen Ansatzes (= Sprache als System), wie er von Ortner (1983) grundlegend und (1997) ausführlich beschrieben wurde, erfolgt Integration über Sprache. Sprache ist dabei nicht Mittel zum Zweck, sondern selbst Integrationsgegenstand. Die systematische Unterstützung dieser semantischen Integration ist ein herausragendes Merkmal der sprachbasierten Informatik. Sie erfolgt schwerpunktmäßig in der Rekonstruktion. Über die gezeigte, konstruktivistisch fundierte und systematische Verankerung sowie Handhabung von Sprache in der Entwicklungsarbeit bietet die Rekonstruktion ein bedeutendes und bisher nicht erkanntes Integrationspotenzial für das Usability Engineering. Das gilt in erster Linie für die sehr frühen Phasen der Entwicklungsarbeit. Erst wenn die sprachbasierte Arbeit in diesen frühen Phasen konsequent umgesetzt wird, ergeben sich Synergien für die Phase der Stabilisierung zur Stützung der Nutzer mittels Wissen. Die Sprachbasierung der Rekonstruktion ist in der Lage, das Verstehen von Systemen (vgl. Abschnitt 2.4.5,

S. 80ff) grundlegend, systematisch und über den Einsatz der natürlichen Sprache zu begünstigen bzw. überhaupt erst zu ermöglichen.

Im Usability Engineering wird die Wichtigkeit der Beteiligung des Benutzers wiederholt über Modelle und Methoden zum Ausdruck gebracht (vgl. Abschnitte 2.4.3, S. 66ff u. 2.4.4, S. 70ff). Auch die Wichtigkeit des Einsatzes der natürlichen Sprache kommt implizit zum Ausdruck. Als Beispiel sei hier die natürlichsprachliche Darstellung eines Nutzungskonzepts in Form einer Aussagensammlung genannt (DATech, 2008). Eine systematische Unterstützung dieser komplexen Arbeiten, vergleichbar mit jener, wie sie dem Anwendungsentwickler der sprachbasierten Informatik u.a. mit der Rekonstruktion angeboten wird, wird dem Usability Engineer nicht angeboten. Der Usability Engineer kann in den frühen Entwicklungsphasen auf umfangreiche Literatur zurückgreifen, insbesondere im Hinblick auf Web-Usability und das Design von Benutzeroberflächen (z.B. Manhartsberger & Musil, 2001; Heinsen & Vogt, 2003; Usability first, 2008; usability.gov, 2008). Die Bearbeitung von Themen zur Beeinflussung der Sensual Usability (vgl. Abschnitt 2.4.2, S. 62ff) stehen dabei im Vordergrund. Aspekte der Emotional Usability werden im Usability Engineering bestenfalls am Rande erwähnt (z.B. Hassenzahl & Hofvenschiöld, 2003), können jedoch Eingang finden über das Verständnis von Usability als Teilaspekt von *Akzeptanz*. Eine Diskussion der Intellectual Usability erfolgt nicht, demnach bleibt auch deren systematische Beeinflussung außer Acht.

In der Disziplin des Usability Engineering gibt es keinen vergleichbaren und fundierten sprachbasierten Ansatz der grundlegenden Systematisierung der Arbeit eines Usability Engineers, wie es das für einen Entwickler, oder besser einen *Enterprise Engineer*, in der sprachbasierten Informatik gibt. Der Sprache ist Integration immanent. Für eine nahtlose Zusammenführung der Aufgaben des Usability Engineering und der Anwendungsentwicklung auf der internen Ebene (Tab. 1, S. 7) bedarf es demnach „lediglich“ einer gemeinsamen und grundsätzlichen Begriffsklärung zwischen den Disziplinen und einer Anwendung der resultierenden Begriffsstandards in der nutzerorientierten Entwicklungsarbeit. Durch eine konsequente Umsetzung der beschriebenen Vorgehensweise der Rekonstruktion wird die semantische Integration sichergestellt.

Existierende Integrationsansätze zwischen Usability Engineering und Anwendungsentwicklung sind struktureller Natur und betreffen überwiegend Vorgehensmodelle (vgl. Abschnitt 3.5, S. 154ff). Dies kommt einer Integration auf konzeptioneller Ebene gleich. Konzepte greifen auf Theorien zurück. Vorgehensmodelle und Methoden sind Darstellungen von Theorien. Die Integration von Theorien benötigt eine Grundlage, genauso wie Theorien eine Grundlage haben sollten. Eine Integration auf konzeptioneller Ebene wird im vorliegenden Kontext als notwendige Ergänzung und Unter-

stützung der semantischen Integration aufgefasst. Die semantische Integration ist Grundlage und wurzelt in der logischen Propädeutik nach Kamlah & Lorenzen (1967).

3.3 Modellierung auf anwender- und entwicklersprachlicher Ebene

3.3.1 Zweck und Ablauf

Die Anwendungsentwicklung ist eng verbunden mit der Unternehmensmodellierung und wird begründet mit dem Selbstverständnis, dass die Organisation, in der das zu entwickelnde System eingesetzt wird, in den Entwicklungsprozess integriert wird (vgl. Abschnitt 2.3.1, S. 35ff). Mit ProCEM® steht ein Rahmen zur Verfügung, innerhalb dessen die Unternehmensmodellierung stattfinden kann, wobei der Mensch, die Organisation und die Informationstechnologie (genau in dieser Reihenfolge) jeweils im Zentrum der Betrachtungen stehen.

Systeme, also Unternehmen und Anwendungsbereiche verstanden als soziotechnische Systeme, sind im Allgemeinen komplexe Gebilde, deren vollständige Beschreibung schwierig oder unmöglich ist. Im Zuge der Anwendungsentwicklung, respektive im Abschnitt der Modellierung, werden *Modelle* entwickelt, in denen die untersuchten Systeme auf relevante Merkmale reduziert werden. Die Diskussion über diese Systeme geschieht in diesem Merkmalsraum. Die Ergebnisse solcher Untersuchungen werden dann aus dem Modell in Form eines Anwendungssystems wieder auf das reale System, den Anwendungsbereich, projiziert (Wedekind et al., 1998, S. 266-267, mit Bezug auf Kowalk, 1996, S. 28). Der als Modellierung bezeichnete Abschnitt der Entwicklungsarbeit in der sprachbasierten Informatik umfasst alle Zwischenschritte, die aus dem anwendersprachlichen Kontext kommend den entwicklersprachlichen Kontext betreffen, bevor die Ergebnisse wieder dem Anwendungsbereich zur Verfügung gestellt werden (Abb. 26, S. 101).

Ungeachtet dessen kann die Unternehmensmodellierung aus unterschiedlichen Standpunkten vorgenommen werden. Beim naiven Standpunkt wird davon ausgegangen, dass ein mehr oder weniger einfacher Repräsentationsformalismus, wie z.B. eine grafische Notation (Graphenmethode) oder eine Diagrammtechnik (UML), zur angemessenen Darstellung von komplexen Begriffen und Begriffsbeziehungen in Fachabteilungen eingesetzt werden kann. Dieser Repräsentationsformalismus wird auch als Kommunikationsmittel zwischen Anwendungsbereich und Entwicklungsabteilung verwendet. Der spezifische Rekonstruktionsprozess unterbleibt in der Regel. Die Benutzer stimmen den grafischen Darstellungen irgendwann zu. Das Ziel unternehmensweit konsensfähiger Fachbegriffe wird selten erreicht. So erläutern neben

Ortner (1993, S. 17) auch Jablonski et al. (1999, S. 170), dass allein mit solchen Darstellungsmitteln selten adäquate Modellierungsergebnisse erzielt werden.

Bei der Modellierung nach dem dogmatischen Standpunkt wird ein ausdrucksstarker Repräsentationsformalismus, der für die Implementierung der Systeme zur Verfügung steht, bereits zur Modellierung der fachlichen Lösung eingesetzt. Beispiele hierfür sind das Relationenmodell, eine Skriptsprache oder eine andere geeignete Programmiersprache. Eine direkte Beteiligung der Benutzer ist hier nicht mehr möglich, da sich der Formalismus in keiner Weise als Kommunikationsmittel mit dem Anwender eignet. Die Ergebnisse sind vergleichbar den Ergebnissen des naiven Standpunkts (Ortner, 1993, S. 18).

Der normsprachliche Entwurf unterscheidet sich grundlegend von den vorher genannten Standpunkten. Er ist gekennzeichnet durch seine Methodenneutralität, seine *Normsprachlichkeit* und seinen materialen Charakter (Jablonski et al., 1999, S. 155). Es wird davon ausgegangen, dass Sprache und deren Verwendung (Sprachartefakte) die wesentlichen und integrativen Elemente bei der Entwicklung und der Anwendung eines Anwendungssystems darstellen. Der normsprachliche Entwurf genügt den Ansprüchen der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik und fördert den Übergang von der Rekonstruktion zur Modellierung.

Streng genommen beginnt die Modellierung aus Sicht der betroffenen Sprachebenen (vgl. Abb. 26, S. 101) bereits beim Fachentwurf in Form normalisierter Aussagen auf Basis geklärter Fachbegriffe. Aufgabe der Modellierung ist es, aus diesem fachlichen Modell ein maschinell ausführbares Modell zu erstellen. Diesem Ziel folgend, wird ein sprachliches Modell in Ausprägung der normierten Aussagensammlung zweckorientiert fragmentiert und in weitere sprachliche Darstellungen transformiert, bis es schließlich nach der Implementierung bzw. Konfigurierung als maschinell ausführbares Modell wieder in den Anwendungsbereich zurückgeführt wird. Die Transformation ist nicht nur eine Übersetzung. Die Modellierung muss sich zum einen den empirischen Beschreibungen der Aussagensammlung fügen, zum anderen sind die Vorgaben von Theorien zu beachten. Es geht dabei nicht darum, ein deskriptives Idealmodell (Wedekind et al., 1998, S. 267) zu erstellen, sondern um ein theoriebasiertes Modell aus einer gelungenen Rekonstruktion zweckorientiert weiterzuentwickeln. Die Grundlage der Modellierung aus Sicht der sprachbasierten Informatik wird durch die Konstruktionslehre innerhalb einer Methodologie (z.B. ProCEM®) gesichert. Eine mögliche *Theorie* für die Modellierung ist z.B. die objektorientierte Entwicklung mit dem zugehörigen Modellierungsinstrumentarium. Für die folgenden Betrachtungen und Beispiele wählen wir diese *Theorie* und setzen sie auf die in den Abschnitten 2.1 (S. 15ff) und 2.2 (S. 22ff) in wesentlichen Zügen eingeführten Grundlagen der sprachbasierten Informatik auf. Spätestens mit dem Erreichen des entwick-

lersprachlichen Kontexts geht sowohl die Arbeitslast als auch die Verantwortung für das Ergebnis auf die Herstellerorganisation über.

3.3.2 Klassifizierung und Formalisierung der Aussagen

Mit dem Übergang vom anwendersprachlichen Kontext in den entwicklersprachlichen Kontext ist das Verlassen der Methodenneutralität verbunden. Das heißt, in dieser Entwicklungsphase wird entschieden, mit welchem Lösungsparadigma die Problemlösung des Anwendungsbereichs erfolgen soll. Es wird entschieden, ob z.B. ein Datenbanksystem oder ein Workflowsystem gebaut wird und/oder ob die Lösung durch Orchestrierung von Services erfolgen kann. Für den Übergang von der bisher vorherrschenden Problemorientierung zu einer Lösungsorientierung bedarf es einer lösungsorientierten Sortierung der Aussagen. Die Aussagen werden, ausgerichtet auf den zu entwickelnden Anwendungssystemtyp, geordnet. Dazu dient der Spezifikationsrahmen nach Schienmann (1997, S. 51-55; Abb. 29). Er legt die notwendigen Teilbeschreibungen eines objektorientierten Anwendungssystems fest. In der Modellierung finden dabei primär die Prinzipien der *Abstraktion* und der Komposition Anwendung. Die entwicklungsrelevanten und normierten Aussagen aus dem Fachentwurf werden den Ergebnisfeldern des Spezifikationsrahmens zugeordnet. Der Spezifikationsrahmen korrespondiert mit dem Klassifikationsrahmen zur Prüfung der Vollständigkeit der Aussagen aus der Rekonstruktion (vgl. Abb. 28, S. 130). Die Arbeitsschritte der Vollständigkeitsprüfung und der lösungsorientierten Aussagenklassifikation können also auch verbunden werden.

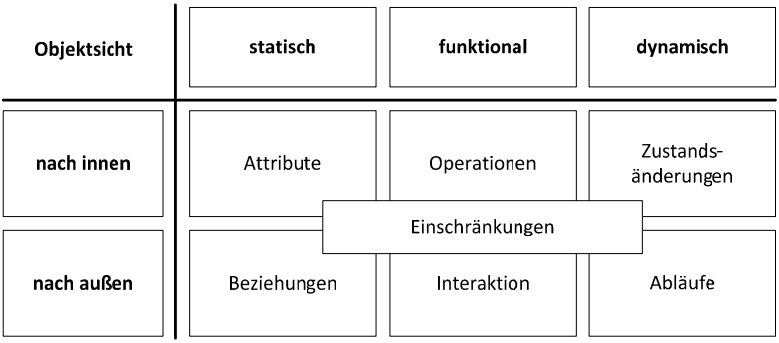

Abb. 29: Spezifikationsrahmen eines objektorientierten Softwareentwurfs

Im gleichen Zuge erfolgt die Umsetzung der natürlichsprachlichen, normierten Aussagen in diagrammsprachliche Darstellungen. Dafür steht uns mittlerweile ein umfassendes Instrumentarium zur Verfügung (vgl. Abb. 16, S. 56). Diese Umsetzung

kommt einer Transformation von „Inhalt“ (Fachsemantik) in verschiedene Sprachen gleich. Sie wird durch die Verwendung des Spezifikationsrahmens erleichtert, da sowohl zwischen den einzelnen Feldern des Spezifikationsrahmens und UML-Diagrammen bzw. BPMN-Diagrammen und damit auch zwischen den, den Feldern zugeordneten „Inhalten“ (Entwicklungsergebnissen) und UML-Diagrammen bzw. BPMN-Diagrammen eine Verknüpfung besteht. Für den objektorientierten Softwareentwurf sieht die Zuordnung wie folgt aus (Ortner et al., 2008):

- **Klassendiagramme** – Sie repräsentieren die statische Sicht nach innen und außen (Attribute und Beziehungen) sowie die funktionale Sicht nach innen (Operationen).
- **Sequenzdiagramme** – Sie dienen der Beschreibung der funktionalen Sicht nach außen (funktionaler Zusammenhang verschiedener Klassen zur Erledigung einer Aufgabe).
- **Zustandsdiagramme** – Sie dienen für die Abbildung der internen Zustände der Objekte einer Klasse (dynamische Sicht nach innen).
- **Aktivitätsdiagramme** – Sie bilden die Reihenfolge (Abläufe) der Zusammenarbeit verschiedener Objekte unterschiedlicher Klassen in einem Computer-Prozess (dynamische Sicht nach außen) ab.
- **Geschäftsprozess- und Arbeitsablaufdiagramme** – Sie eignen sich für die Abbildung von Interaktionen und Reihenfolgen (funktionale und dynamische Sicht nach außen) und für die Abbildung der Beziehungen von Prozesstypen auf Metaebene (statische Sicht auf Prozesse und deren Beziehungen).

Aus dem Sprachumfang der UML bilden Use Case Diagramme ein geeignetes Instrument für die statische Modellierung. Für die Modellierung dynamischer Sichten bieten sich Sprachen wie z.B. die Business Process Modeling Notation (BPMN) an. Die beiden genannten Modellierungssprachen können hinsichtlich ihrer Einsetzbarkeit für die Modellierung unterschiedlicher Systemtypen noch als weitgehend „neutral“ bezeichnet werden. Andere Teilsprachen der UML, wie z.B. Klassendiagramme, Aktivitäts- und Sequenzdiagramme, eignen sich dagegen überwiegend für den objektorientierten Softwareentwurf und sind inzwischen auch für die serviceorientierte Softwareentwicklung im Einsatz (Ortner et al., 2008). Zweck der Anwendung von Modellierungssprachen ist es, Schemata über das Orientierungs- und Verfügungswissen des Anwendungsbereichs zu integrieren und die Inhalte im Hinblick auf eine Implementierung zu systematisieren.

Bei der detaillierten Erstellung der einzelnen Diagramme ist zu beachten, dass stets Referenzen zu den Spezifikationen der im Spezifikationsrahmen skizzierten Beschrei-

bungssprachen (= konkreter Bezug zu anderen Diagrammen) gemacht werden, um ein zusammenhängendes Bild zu erhalten. In letzter Konsequenz kann dies so weit gehen, dass beispielsweise Softwaremodelle in Verhältnis zur Unternehmenspolitik gesetzt werden.

Die Ergebnisse dieses Klassifikations- und Modellierungskanons sind fachlich spezifizierte Lösungen. Sind diese z.B. als Services bzw. Komponenten am Markt verfügbar, bedarf es keiner weiteren Detailspezifikation oder Eigenentwicklung. Ist dies nicht der Fall, sind die Diagramme weiter zu verfeinern und schließlich in maschinell lesbare Beschreibungen zu übersetzen. Abhängig vom Diagrammtyp gibt es für die genannten Sprachen aus dem Sprachkanon der OMG unterschiedliche Ausführungssprachen. Beispielsweise definiert der BPMN-Standard wie ein BPMN-Diagramm in BPEL (Business Process Execution Language) übersetzt werden sollte, damit die beschriebenen Prozesse durch eine Software ausgeführt werden können. Es gibt noch weitere XML-basierte Sprachen, die dafür in Frage kommen. Technologien und Werkzeuge für die Implementierung stehen ausreichend zur Verfügung, wie z.B. die EJB-Technologie in Form von Session-Beans und Entity-Beans. Bei der Auswahl von Werkzeugen muss nicht unbedingt auf mitunter kostspielige Lösungen einzelner Hersteller zurückgegriffen werden. Für alle Aufgaben der technischen Realisierung sind mittlerweile ausgereifte Open Source Produkte (z.B. J2EE Applicationserver von JBOSS, Geronimo von Apache, Glassfish oder Datenbankserver wie MySQL, Max DB von SAP, PostgreSQL etc.) frei verfügbar, welche die Umsetzung der Ergebnisse aus Abb. 29 (S. 136) ermöglichen. Dass die Open Source Systeme heutigen Qualitätsansprüchen durchaus entsprechen wird auch dadurch unterstrichen, dass kommerzielle Anbieter ihrerseits immer öfter auf Open Source Produkte zurückgreifen. Open Source Produkte sind in der Regel auch gut „theorie-stabilisiert“, während die proprietären Systeme häufig „theorie-frei“ auf dem Markt gehandelt werden (Ortner et al., 2008).

Aus Sicht der Sprachebene vom Standpunkt der Informatik steht nach der Implementierung bzw. nach einer eventuell erfolgten Orchestrierung von Services bzw. Komponenten eine Anwendungssoftware zur Verfügung. In einem weiteren Arbeitsschritt gilt es nun, diese Anwendungssoftware für den Gebrauch zur Verfügung zu stellen. Dazu sind die Ergebnisse der Modellierung aus dem Standpunkt der Sprachebene des Anwendungsbereichs heraus zu betrachten und dem Anwendungsbereich, der Praxis, näherzubringen. Die Ergebnisse der Modellierung und die Verantwortung für die Entwicklungsarbeit gehen mit diesem Schritt von der Herstellerorganisation wieder in die Anwenderorganisation über.

3.4 Stützung der Nutzer für den Gebrauch

3.4.1 Rückführung des Anwendungssystems in den Gebrauch

Die implementierten Modellierungsergebnisse als Anwendungssoftware dem Nutzer zum Gebrauch zur Verfügung zu stellen erfordert die Rückführung der Entwicklungsergebnisse auf die anwendersprachliche Ebene und weiter auf die Ebene des Anwendungsbereichs (vgl. Abb. 26, S. 101). Auf diesem Pfad steht die Stützung der Nutzer für die Nutzung des Anwendungssystems im Vordergrund. Um von der entwicklersprachlichen Ebene auf die Sprachebene vom Standpunkt des Anwendungsbereichs zu gelangen, ist anfänglich ein Abgleich zwischen den Rekonstruktionsergebnissen und den Ergebnissen der Modellierung bis hin zur Implementierung vorzunehmen. Dieser Abgleich ist notwendig, um die im Zuge der Modellierung vorgenommenen Änderungen des Rekonstruktionsergebnisses transparent zu machen. Identifizierte Abweichungen werden herausgehoben und den Anwendern zusammen mit der Anwendungssoftware als zusätzliches Verrichtungs- bzw. Orientierungswissen nähergebracht.

Die Anwender waren in die Rekonstruktion eingebunden bis die anwendersprachliche Ebene verlassen wurde. Abhängig von der konkret gewählten Vorgehensweise sind die Benutzer am Rande der Modellierungsphase eventuell mit Tests von Prototypen konfrontiert worden. An der eigentlichen Modellierung waren sie nicht beteiligt. Der Anwender kennt also das natürlichsprachliche Modell des relevanten Anwendungsbereichs in Gestalt eines Fachentwurf als Ergebnis der Rekonstruktion und ist im Rahmen der Stabilisierung, so wird die Stützung der Nutzer für den Gebrauch allgemein genannt, mit dem implementierten Modell konfrontiert.

Hat sich im Zuge der Modellierungsphase keine Änderung des Modells ergeben, wird das implementierte Modell anhand des natürlichsprachlichen Modells (normierte Aussagen) validiert. Die Anforderungen, die im natürlichsprachlichen Modell festgelegt sind, beruhen auf einem bestimmten Gebrauch des entwickelten Systems. Anhand des implementierten Modells im Gebrauch kann nun festgestellt werden, ob die Anforderungen erfüllt sind. Dieser Vorgang wird als Validierung des Anwendungssystems gegen die gestellten Anforderungen bezeichnet. Durch die spezifische Ausgestaltung von Anforderungen in der sprachbasierten Informatik (vgl. Abschnitt 3, S. 99ff) wurde bereits im Rahmen der Rekonstruktion sichergestellt, dass validierbare Anforderungen vorliegen.

Im Zuge der Modellierung kann es zu Abweichungen zum ursprünglich mit den Benutzern erarbeiteten Modell kommen. Diese Abweichungen resultieren u.a. aus softwaretechnischen oder implementierungsspezifischen Gegebenheiten bzw. Ent-

scheidungen und werden in der Regel nicht mit dem Benutzer akkordiert. Die Änderungen des Modells müssen dem Benutzer zusätzlich zum Anwendungssystem als Ganzes nähergebracht werden. In passend gestalteten Schulungen wird das Orientierungs- und Verrichtungswissen der Nutzer auf den aktuellen, der Umsetzung im Anwendungssystem entsprechenden Stand gebracht. Erst dann sind die Nutzer in der Lage, die entwickelte Lösung zu validieren und schließlich effektiv und effizient zu nutzen.

Die Zugänglichkeit eines Anwendungssystems bestimmt sich auf globaler Stufe über die Systemoberfläche. Anwender, die vom Bild der Benutzeroberfläche auf ihren Inhalt schließen können, erfahren dies als globale visuelle Unterstützung (Makrotypografie, durchgängige Strukturen der Interaktionsmasken, Anordnung von Textteilen, Funktionselementen und Grafiken; Bartsch, 2001, S. 21). Es geht um die äußerliche Verdeutlichung von Semantik und inneren Strukturen. Dies kann z.B. erfolgen durch Visualisierungen, umrandete Textteile zur Zusammenfassung im Text, Verzeichnisse, Worterläuterungen oder Orientierungen für den Anwender, damit er weiß, an welcher Stelle im System er sich befindet. Wichtig dabei ist, dass das Erscheinungsbild dem Inhalt entspricht, dass sich „Augensinn“ und „Kognition“ ergänzen. Eine nur äußerlich suggerierte Klarheit ist ebenso störend wie eine fehlende äußere Strukturierung.

Den äußeren Gestaltungsmerkmalen eines Anwendungssystems kommt die Aufgabe zu, Informationen auffindbar zu machen und deren tiefergehende Verarbeitung zu fördern. Der Anwender kann bei hoher Zugänglichkeit (z.B. anhand der Überschriften, Buttons, Verzeichnisse) erkennen, „in welche Richtung“ das Verständnis gehen soll. Die sachliche Richtigkeit der Information wird vorausgesetzt. Es gilt natürlich, nur mögliche relevante Anschlusshandlungen auf der Benutzeroberfläche darzustellen bzw. zu berücksichtigen. So sollten mögliche Handlungen, die z.B. der Umgehung von Vorschriften dienen, nicht dargestellt werden. Das mentale Modell, das der Anwender vom dargestellten Sachverhalt hat, wird über die Benutzeroberfläche angesprochen und aktiviert. Dies ist für gebrauchstaugliche Systeme unerlässlich.

Für Anwendungssysteme im unternehmerischen Kontext wird unterstellt, dass diese nicht einfach intuitiv oder zufällig richtig und gut bedient werden können. Das heißt konkret, dass zusätzlich zu einer benutzeradäquat gestalteten Benutzeroberfläche der richtig geschulte Benutzer ein wesentlicher Bestandteil eines gebrauchstauglichen Systems ist. Das Vorhandensein eines mentalen Modells beim Benutzer ist unabdingbare Voraussetzung für die Anwendbarkeit. Verstehbarkeit und Anwendbarkeit eines Anwendungssystems gehen jedoch über den erfolgreichen Aufbau eines mentalen Modells hinaus. Mit Schulungen werden die mentalen Modelle der Anwen-

der aktualisiert und adaptiert. Der notwendige Aufbau der mentalen Modelle wird als Teil der systematischen Entwicklungsarbeit verstanden.

Im Rahmen der Stützung werden künftige Nutzer von Anwendungssystemen geschult. Schulungen für Benutzer sollten nicht nur auf die Nutzung und den Umgang mit dem System vorbereiten, sondern bei Bedarf auch auf organisatorische und soziale Änderungen eingehen. Nach Möglichkeit sollten Schulungen nicht nur vor der Einführung sondern über den gesamten Produktlebenszyklus institutionalisiert werden. Im Sinne einer Schachbrettmetapher kann dafür bei Bedarf flexibel auf geeignete Methoden bzw. Methodenkomponenten zurückgegriffen werden (vgl. Abschnitte 2.3.4.3, S. 57, 2.4.4, S. 70 u. 3.5.4, S. 171). Die Grundlagen für benutzerzentrierte Schulungen werden in der sprachbasierten Informatik durch ein systematisches, über den gesamten Lebenszyklus eines Anwendungssystems institutionalisiertes, Wissensmanagement geschaffen.

3.4.2 Exkurs in das Wissensmanagement zum Zweck der Stabilisierung

3.4.2.1 Überblick und Begriffsverständnis

Die Thematik des Wissensmanagement wurde in den letzten Jahren aus unterschiedlichen Gesichtspunkten behandelt, zunehmend standen dabei Fragen der Gestaltung und Implementierung von Wissensmanagementsystemen in Unternehmen im Vordergrund. Frei nach Davenport & Prusak (1999) kann gesagt werden, dass es bei diesen Aktivitäten vordergründig darum ging, vorhandenes Know-how in den Unternehmen in IT-Systemen verfügbar, wiederauffindbar und damit nutzbar zu machen. Wissensmanagement ist zu einem Trend avanciert, da es zunehmend wichtiger wird, die Ressource Wissen (nicht nur individuelles Wissen) in den Mittelpunkt unternehmerischen Denkens zu stellen.

Wissensmanagement lediglich aus Sicht von einzelnen Disziplinen wie z.B. der Betriebswirtschaft oder der Anwendungsentwicklung zu betrachten, reicht für den vorliegenden Kontext nicht aus. Für die vorliegenden Fragestellungen bedarf es einer grundlegenderen Sichtweise auf das Wissensmanagement, welche mit der in Abschnitt 2.2 (S. 22ff) ausgeführten Schemaentwicklung bereits eingeleitet wurde. Als Grundlage für die weiteren Ausführungen dient die Arbeitsdefinition von Wissen von Probst, Raub & Romhardt (2006, S. 46):

> Wissen bezeichnet die Gesamtheit der Kenntnisse und Fähigkeiten, die Individuen zur Lösung von Problemen einsetzen. Dies umfasst sowohl theoretische Erkenntnisse als auch praktische Alltagsregeln und Handlungsanweisungen. Wissen stützt sich auf Daten und Informationen, ist im Gegensatz zu diesen jedoch immer an Personen gebunden. Es wird von Individuen konstruiert und repräsentiert deren Erwartungen über Ursache-Wirkungs-Zusammenhänge in einem bestimmten Kontext.

Auf den ersten Blick könnte man meinen, die Definition beschränke sich auf individuelles Wissen. Doch auch Wissen, das als Software (= Schemata) in Form von Computerprogrammen oder Anwendungssystemen verfügbar ist, wurde ursächlich von Individuen konstruiert. Ergänzend dazu übernehmen wird von Davenport & Prusak (1999, S. 32) die Definition von organisationalem Wissen:

> In Organisationen ist Wissen häufig nicht nur in Dokumenten oder Speichern enthalten, sondern erfährt eine allmähliche Einbettung in organisatorische Routinen, Prozesse, Praktiken und Normen.

Bei der Definition von organisationalem Wissen bildet Wissen Prozesse, Praktiken und Normen, welche ebenfalls als Schemata zu verstehen sind. Bei ihrer Definition von individuellem Wissen stellen Davenport & Prusak dieses als eine fließende Mischung aus strukturierten Erfahrungen, Wertvorstellungen, Kontextinformationen und Fachkenntnissen dar. Sie charakterisieren Wissen als Strukturrahmen zur Beurteilung und Eingliederung neuer Erfahrungen und Informationen. Bei individuellem Wissen handelt es sich demnach um Schemata die a priori vorhanden sind. Aus Sicht der Psychologie handelt es sich dabei um mentale Schemata. Der Zusammenhang zwischen Wissen und Information, Schema und Ausprägung sowie Handeln und Verstehen wird in Abb. 30 verdeutlicht.

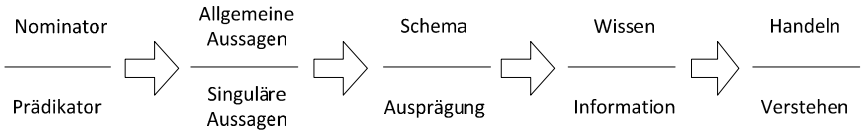

Abb. 30: Zusammenhänge Schema-Ausprägung sowie Wissen-Information

Wissen als Schema bietet Orientierung für das Handeln. Die Ausprägungen von Schemata, z.B. in Form von singulären Aussagen, transportieren Informationen und tragen zum Verstehen bei. Wissen ist demnach mächtiger und bedeutender als Information. An diesem Punkt werden der Zusammenhang und die Relevanz des Wissensmanagements für die Anwendungsentwicklung deutlich. Der Wissensmanagementprozess ist ein Schemabildungsprozess und die Anwendungsentwicklung ist

ihrerseits ein Schemabildungsprozess, da Anwendungssysteme zusammengefügte Schemata sind. Werden Schemata zusammengefügt, auch ohne dass von außen zusätzliches Wissen hinzukommt, kann es wieder neues Wissen geben. Die Auffassung von Wissen als Schemata begleitet den gesamten Entwicklungsprozess. Zur gezielten Steuerung des Wissens zum Zweck der Stützung der Nutzer ist es mitunter erforderlich, das relevante Wissen zur Wissensvermittlung nach verschiedenen Gesichtspunkten zu systematisieren und zu strukturieren. Hilfsmittel dafür sind Wissenskategorien und Wissensarchitekturen. Nachfolgend werden dazu ausgewählte Wissenskategorien und Wissensarchitekturen eingeführt.

3.4.2.2 Wissenskategorien - Wissensarchitektur

Für die Unterscheidung von Wissenskategorien (detailliert ausgeführt von Heinemann, 2006, S. 54-56) stellt die Differenzierung von implizitem und explizitem Wissen bzw. individuellem und kollektivem Wissen (z.B. Nonaka & Takeuchi, 1995, S. 59ff) einen gängigen Ansatz dar. Bei implizitem Wissen wird zusätzlich eine kognitive Dimension (mentale Modelle geprägt von Wertvorstellungen und Überzeugungen) und eine technologische (informelle, persönliche Fähigkeiten und Fertigkeiten im Sinne von „Know-how“) Dimension unterschieden (Nonaka & Takeuchi, 1995, S. 60). Die Unterscheidung von prozeduralem Wissen (Können, Know-how, handlungsorientiertes Wissen) und deklarativem Wissen (Faktenwissen, Information, Knowing-That) z.B. nach Fink (2000, S. 31) dient als Ansatz in Richtung „Könnensmanagement“ (Ability-Management). Anwendungsbezogen kann Wissen nach Bereichen und Funktionen eines Unternehmens (z.B. Wissen in Prozessen, Wissen in Personen, Wissen in Produkten und Dienstleistungen, Wissen in Beziehungen) eingeteilt werden (Thiesse, 2001, S. 1-2). Interessant erscheint zusätzlich die griechische Einteilung in episteme (universal), techne (technical), phronesis (practical) und mètis (cultural). Wesentlich für den vorliegenden Kontext ist dabei die Kategorie „episteme“. Sie bezeichnet dokumentierbare und überlieferbare Gesetzmäßigkeiten und Prinzipien, also (abstraktes) allgemeingültiges „Wissen über etwas“ (explizites, allgemeingültiges Wissen = Schemata).

Für die Stützung der Nutzer in der sprachbasierten Entwicklungsarbeit werden in Anlehnung an Mittelstraß (2001, S. 75) die Kategorien Weltwissen, Orientierungswissen sowie Sach- und Verfügungswissen herangezogen. Das Verfügungswissen ist das Wissen über Ursachen, Wirkungen und Mittel. Es ist elementar für die Rekonstruktionsarbeit. Das Orientierungswissen ist demgegenüber das Wissen über (gerechtfertigte) Zwecke und Ziele. Für den Fall der Anwendungsentwicklung handelt es sich dabei um Schemata, die Zweck, Ziel und Kontext des Anwendungssystems in der Anwenderorganisation repräsentieren. Diese Schemata sind wichtig zur Orientierung des Entwicklers hinsichtlich des Entwicklungsergebnisses und zur Orientierung des

Anwenders hinsichtlich der Verrichtung seiner Arbeit. Über das Orientierungswissen erfolgt auch eine mögliche Verbindung zur Unternehmensstrategie. Beide Wissenskategorien werden unter dem (expliziten) Weltwissen subsumiert, das in seiner Relevanz für ein Unternehmen, z.B. in Form einer Unternehmens-Enzyklopädie, mit Hilfe eines Repositoriums effizient verwaltet werden kann. Die erwähnten Wissenskategorien wurden bereits in Abschnitt 2.3.4 (S. 52ff) in Verbindung mit den Instrumenten zur Unternehmensmodellierung (Unternehmensrekonstruktion) eingeführt.

Eine weitere Sicht auf spezielle Wissenskategorien führt uns zu einer Wissensarchitektur. Eine Wissensarchitektur bildet einen Rahmen mit sehr freien Gestaltungsmöglichkeiten (im Sinne von Kunst) und beinhaltet gleichzeitig Elemente eines Vorgehensmodells für das Know-how-Engineering (Fink, 2000, S. 72 u. S. 75). Für die Anwendungsentwicklung, insbesondere für den Aufbau eines Repositoriums zur Verwaltung von Grammatik und *Terminologie* im Rahmen der sprachbasierten Informatik, ist die Orientierung an der Wissensarchitektur, wie sie Heinemann (2006, S. 166-176) vorschlägt (Abb. 31), zweckmäßig.

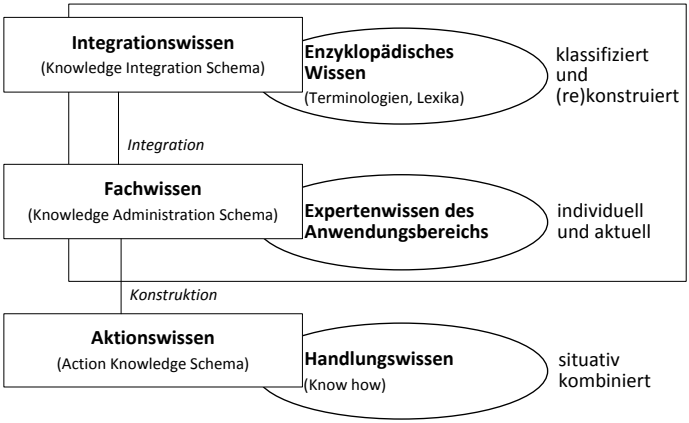

Abb. 31: Wissensarchitektur in Anlehnung an Heinemann

Heinemann beschreibt Sprachwissen (= Integrationswissen), Fachwissen und Aktionswissen als Bestandteile eines Repositoriums. Als Wissensarchitektur verstanden, lassen sich diese Wissenskategorien, eingeteilt in klassifiziertes und rekonstruiertes Wissen, in aktuelles und individuelles Wissen sowie in situativ kombiniertes Wissen, auf Basis der Methoden (sprachkritische Rekonstruktion) dem Multipfad-Vorgehensmodell zuordnen. Es handelt sich also um eine methodenbasierte Verbindung mit Rückschluss zum grundlegenden Verständnis von Wissen als Schemata. Eine (zweckorientierte) Lebenszyklusorientierung der skizzierten Wissensarchitektur ließe

sich über das Darmstädter Wissensmanagement-Konzept (kurz: DarWin) von Heinemann (2006, S. 137-149) herstellen.

DarWin ist ein sprachbasiertes Konzept des Wissensmanagements. Es fokussiert, wie es der Name schon sagt, auf *epistemische* Anwendungssysteme verstanden als Managementsysteme zur Unterstützung menschlicher Selbsttätigkeit auf Basis von Schemata (Heinemann, 2006, S. 134-137). Auf das DarWin-Konzept kann zurückgegriffen werden, wenn im Rahmen der Stützung der Nutzer bzw. der Systemnutzung eine Architektur für das „Könnens-Management" benötigt wird. Die generische Verbindung zu einem Prozessmodell des Wissensmanagements, das der sprachkritischen Entwicklungsarbeit gerecht wird, lässt sich zum Wissenslebenszyklus nach Ortner herstellen.

3.4.2.3 Prozessmodelle des Wissensmanagements

Meist ausgehend von Strukturmodellen (Wissensarchitekturen) oder in Verbindung damit, finden sich z.B. bei Probst, Raub & Romhardt (2006), Fink (2000), Nonaka & Takeuchi (1995) und anderen ablauforientierte Darstellungen unterschiedlichen Charakters. Fink (2000, S. 34-37) verweist auf einen evolutionstheoretischen Ansatz der Wissensentwicklung von Kwasnicki (1996), der sich in der Beschreibung der Wissensentwicklung jedoch auf die menschliche Wissensentwicklung beschränkt. Das Modell von Fink ist sowohl personenbezogen als auch organisationsbezogen anwendbar und sieht im Gegensatz zu anderen Modellen auch die Unterstützung von Wissensvernetzung vor. Wenngleich Anknüpfungspunkte zu identifizieren sind, stellt Fink keinen direkten Bezug zur Anwendungsentwicklung her. Nonaka & Takeuchi (1995, S. 73) schlagen eine Theorie der organisationalen Wissensbildung vor. Probst, Raub & Romhardt (2006, S. 59) verstehen Wissensmanagement als Integrationsauftrag. Sie bieten ein Modell aus sechs Kernprozessen an, welche ergänzt werden durch zwei pragmatische Bausteine, die Wissensziele und Wissensbewertung anwendungsorientiert einbeziehen.

Dem sprachbasierten Ansatz wird das Wissenslebenszyklusmodell von Ortner gerecht. Er beschreibt den Wissenslebenszyklus aufgabenbezogen (Abb. 32, S. 146) und weist darauf hin, dass Wissen auf das Handeln der Menschen stärker Einfluss nimmt als Information. Folgerichtig betont Ortner (2005, S. 166f) in seinen Ausführungen die Handlungsrelevanz des Wissens für Mitarbeiter in einem Unternehmen. Das Wissen gehört von Zeit zu Zeit überprüft und bei Bedarf erneuert, d.h. rekonstruiert. In diesem Fall wird das Wissen der Wiederverwendung zugeführt. Um Wissenssysteme nicht zu „Wissensfriedhöfen" (in Analogie zu „Datenfriedhöfen") werden zu lassen, ist im Rahmen der Überprüfung von Wissensbeständen festzustellen, welches Wissen nicht mehr aktiv ist und auch nicht mehr benötigt wird. Diese Forderung ist nicht trivial. Für Organisationen gelten eine Reihe von Aufbewahrungsvorschriften, denen

Wissenssysteme gerecht werden müssen. Je älter das Wissenszeitalter wird, desto brisanter wird diese Thematik werden. Der gelebte Wissenslebenszyklus unterstützt die Gebrauchstauglichkeit von Wissen und ist anzusprechen über die epistemologische Dimension von Usability. Dies betrifft sowohl das WAS (Wissen als Inhalt von Systemen) als auch das WIE (z.B. Wissen, wie die Gebrauchstauglichkeit systematisch beeinflusst werden kann).

Das Wissensmanagement steht dem Informationsmanagement sehr nahe. Information wird als Ausprägung von Wissen verstanden (Abb. 30, S. 142). Demzufolge kann Informationsmanagement als Wissensmanagement auf operativer Ebene bezeichnet werden. Dies trifft zumindest für den inhaltlichen Teil des Informationsmanagements zu. Das Informationsmanagement als Teilbereich der Wirtschaftsinformatik kann als jenes Aufgabengebiet verstanden werden, das die Anwendungssysteme aus Sicht der (sprachbasierten) Informatik mit der Betriebswirtschaftslehre und weiteren relevanten Disziplinen verbindet (Ortner, 2005, S. 154).

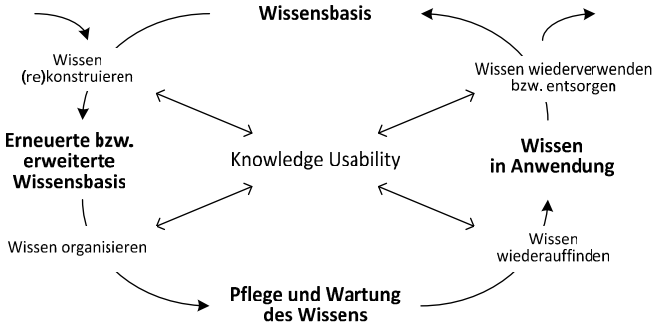

Abb. 32: Wissenslebenszyklus in Anlehnung an Ortner

Konsequenterweise ist das Informationsmanagement im Multipfad-Vorgehensmodell dem Vorgangstyp Gebrauch zugeordnet. Es ist somit Aufgabe der Wissensorganisation im Wissenslebenszyklus, das Wissen unter Verwendung geeigneter Methoden in eine Form zu bringen, die eine optimale Verwaltung und Bereitstellung des Wissens erlaubt. In diesem Zusammenhang ist auf Kriterien wie Bedarf, Verfügbarkeit, Kosten, Schnelligkeit u.a.m. zu achten. Nicht mehr aktuelles Wissen, bzw. Wissen das nicht mehr benötigt wird, gehört entsorgt, um eine „Vermüllung“ von Wissenssystemen zu verhindern (Heinemann, 2006, S. 58).

3.4.2.4 Methoden im Wissenslebenszyklus

Sprachbasierte Rekonstruktion

Im Zusammenhang mit dem von Ortner (2005, S. 166f) beschriebenen Wissenslebenszyklus sind neben der sprachbasierten Rekonstruktion und den ihr untergeordneten Vorgehensweisen keine Methoden beschrieben. Die Rekonstruktion ist vollumfänglich in das Multipfad-Vorgehensmodell integriert und in Abschnitt 3.2 (S. 104ff) beschrieben. Es handelt sich dabei um eine Methode, die das Wissensmanagement grundlegend unterstützt und das über den gesamten System- oder Wissenslebenszyklus.

GABEK®

GABEK® steht für GAnzheitliche BEwältigung von Komplexität. Die Methode wurde von Prof. Josef Zelger, Professor für Philosophie, und seinen Mitarbeitern an der Universität Innsbruck entwickelt. Die dazugehörige Software (WinRelan®) wurde 2002 von der European Knowledge Media Association in Schweden mit dem Prädikat „Exceptional Quality“ ausgezeichnet. GABEK® wurde als Methode zur Unterstützung der qualitativen Forschung entwickelt und wird mittlerweile zur Meinungsforschung, zur Wissensverarbeitung und zur Systemgestaltung eingesetzt. Basierend auf der Vernetzung von Erfahrungen vieler Personen wird eine ganzheitliche Darstellung komplexer gesellschaftlicher Situationen ermöglicht (Zelger & Buber, 2000). Die Methode erlaubt eine Darstellung von Begriffs- und Wissensnetzen einer Organisation ebenso wie deren Interpretation (Zelger, Raich & Schober, 2008).

GABEK® ist im Abschnitt Wissensmanagement als Methode eingereiht, weil es die Verarbeitung von Wissen unterstützen kann. Zelger (2000, S. 31-40) baut dabei auf die Erkenntnisse der menschlichen Wissensverarbeitung und unterscheidet die unbewusste Erfahrungsverarbeitung, welche durch parallele mentale Prozesse zustande kommt und die bewusste Wissensverarbeitung, welche als serielle Repräsentation selektierter Inhalte aufgefasst wird. Nach Zelger läuft die kognitive Wissensverarbeitung ständig ab und die empfangenen und gespeicherten verbalen Basisdaten werden unbewusst immer wieder neu zu „sprachlichen Gestalten“ (wie Zelger ein solches Konstrukt nennt) zusammengestellt, welche wiederum zu „sprachlichen Hypergestalten“ verbunden werden. Es werden „Gestaltenbäume“ gebildet. Nach der gleichen Systematik werden „sprachliche Gestalten“ in „Wirkungsnetzen“ und „Bewertungsprofilen“ miteinander vernetzt.

Was kann nun unter einer sprachlichen Gestalt verstanden werden? Naheliegend ist die Herstellung einer Verbindung zu Schemata. Die aufgenommenen Worte (Sprache) werden im Gehirn über (bewusste oder unbewusste) mentale Prozesse zu Worten hinzugefügt, die bereits vorhanden sind. Die bereits vorhandenen Worte (aber nicht

nur Worte) und deren Vernetzungen bilden das Wissen des Menschen, d.h. sie stellen jene Schemata dar, die dem Menschen Handlungsorientierung ermöglichen. Durch Einflüsse von außen, z.B. Worte und Sprachartefakte, die aufgenommen werden, werden nun diese Schemata verändert. Der Mensch lernt[21], seine Schemata werden erweitert und verfeinert. In der Folge wird natürlich der Gestaltenbaum erweitert und verfeinert. „Gestaltenbäume" nach Zelger sind also Schemata, und Schemata sind Wissen, und es wird unterstellt, dass eine solche Ordnung unbewusst erzeugt werden kann (Zelger, 2000, S. 44). Für Details und Erfahrungsberichte zur Methode GABEK sei hier auf die einschlägige Literatur (Zelger, 2000; Zelger et al., 2000; Zelger et al, 2008) und die Webseite (www.gabek.com) verwiesen.

Normalsprachliche Äußerungen (Aussagen) sind mehr als eine einfache Verknüpfung von lexikalischen Begriffen (im Sinne einer Parasprache nach Ortner, 2005, S. 248). Die grammatikalische Ordnung und Reihung der Worte steuert die Möglichkeiten des Ausdrucks auf der einen Seite und die des Verstehens auf der anderen Seite. Systemgestütztes sprachliches Verstehen ist nur möglich, wenn Erfahrungen (in Form von Aussagen erfasst) systematisiert werden. Dies könnte z.B. mit Hilfe von GABEK® passieren. Die Freiheitsgrade und die Flexibilität erhöhen sich mit wachsender Systematisierung (Schematisierung) der Erfahrung. Es liegt dann nahe, durch Systematisierung des Erfahrungswissens von Mitarbeitern und Kunden eines Unternehmens die Flexibilität des Unternehmens und die Vielfalt der möglichen Reaktionen zu erhöhen (Zelger, 2000, S. 79). In einem CRM-System sollte dies möglich sein. Zelger geht sogar so weit, dass er von einer bewussten Steuerung des Verhaltens spricht, die Steuerung wird beeinflusst durch die Begriffssystematik (nach GABEK®) im Hintergrund. Hier könnte ein Anknüpfungspunkt zur Akzeptanzforschung bzw. zur Beeinflussung von *Akzeptanz* in der benutzerzentrierten Entwicklungsarbeit sein, denn dabei geht es um nichts anderes als um Verhaltenssteuerung, z.B. auf Basis der Akzeptanzfaktoren des UTAUT-Modells (vgl. Abschnitt 2.4.6.3, S. 94ff) und in Verbindung mit der sprachbasierten Informatik auf Basis der Sprache, systematisiert in der logischen Propädeutik.

In der Erfassung und Steuerung von organisationalem Wissen als Analyse- und Darstellungsinstrument wurde GABEK® untersucht und es gibt Erfahrungsberichte. Es gibt auch Erfahrungsberichte, wonach GABEK® zur Systematisierung von Veränderungsprozessen in Unternehmen eingesetzt wurde (Schober, 2008, S. 139). Keine Erfahrungsberichte konnten in der Anwendung von GABEK® zur Unterstützung von Modellierungsarbeiten zur Unternehmensmodellierung gefunden werden. GABEK® unterstützt mit einer Systematik von Gestaltenbäumen und gestattet, normalsprachliche Äußerungen (Aussagen) so miteinander zu vernetzen, dass sich eine klare Orientierung über eine Meinungslandschaft anbietet. Aus den zahlreichen Anwendungs-

21 In diesem Fall ist es ein Lernen durch Verstärkung der Begriffsverknüpfung (Zelger, 2000, S. 78).

berichten (Zelger & Buber, 2000; Zelger et al., 2008) konnte jedoch kein Bericht über einen Methodeneinsatz in der Anwendungsentwicklung gefunden werden. Dennoch lassen die vorliegenden Beschreibungen (Zelger et al, 2008) vermuten, dass dieses Verfahren auch in der Unterstützung der sprachkritischen Entwicklungsarbeit wertvolle Dienste leisten könnte. Es könnte zudem Sinn machen, das zugehörige Werkzeug (WinRelan®) in Verbindung mit einem Repository entwicklungsbegleitend einzusetzen, was in der Anwendung jedoch mittels Einsatz von WinRelan® noch zu testen und zu evaluieren ist.

3.4.2.5 Zusammenfassende Bewertung

Wissen ist in der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik als Gegenstand der Arbeit zu sehen (WIE) und auch als deren Inhalt (WAS). In Analogie zum Wandel des Informationsmanagement zum Know-how-Management (in Anlehnung an Fink, 2000) erfolgt der Wandel der Anwendungsentwicklung zu einem Knowledge Engineering. Die Rekonstruktion als Methode für dieses Knowledge Engineering wurde in Abschnitt 3.2 (S. 104ff) ausführlich dargestellt. Als weitere mögliche Methodik in Verbindung mit der sprachbasierten Vorgehensweise erscheint der Ansatz von Zelger et al. (2008) interessant und soll bei einer Vertiefung der methodenbasierten Integration des Usability Engineering in die Anwendungsentwicklung erneut aufgegriffen werden. Methoden zur Identifikation von rekonstruktionsbedürftigem oder zu entsorgendem Wissen konnten nicht evaluiert werden. Ein Repository kann in diesem Zusammenhang jedoch sicherlich eine Schlüsselrolle einnehmen.

Das Wissen wird in der Rekonstruktion und der Modellierung systematisch rekonstruiert und organisiert. Zur Stützung der Nutzer wird dieses Wissen benötigt, um den Lernprozess der Nutzer zu unterstützen. Diese Qualifikation ist integraler Bestandteil der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik und begünstigt nicht zuletzt die Gebrauchstauglichkeit eines Anwendungssystems. Im Rahmen der Stützung der Nutzer für den Gebrauch eines Anwendungssystems ist es maßgebend, dass ein lebenszyklusorientiertes Wissensmanagement gepflegt wird. Die Einflussmöglichkeiten beschränken sich nicht auf die Gebrauchstauglichkeit, sondern gelten für nahezu alle Einflussfaktoren des UTAUT-Modells im Hinblick auf die bewusste „Schaffung“ von *Akzeptanz*. Bei dieser erweiterten Sicht auf die Anwendungsentwicklung wird klar, dass sich die Stützung des Nutzers nicht auf eine einmalige Schulung vor Systemeinführung beschränken kann, auch wenn dort sicher ein Schwerpunkt zu sehen ist. In der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik erstreckt sich die Stützung, mit mehr oder weniger Gewicht, über den gesamten Lebenszyklus des Anwendungssystems und soll sich seinerseits wieder am Wissenslebenszyklus orientieren. Damit das gelingen kann, ist der Wissenslebenszyklus nach Ortner (Abb. 32, S. 146) in Verbindung mit dem Multipfad-

Vorgehensmodell zu sehen. Die Kongruenzen wurden von Eller (2008b) in Verbindung mit der Gebrauchstauglichkeit von Wissen verdeutlicht.

3.4.3 Duale Verwendung der Modellierungsergebnisse aus organisationszentrischer Sicht

Die Rückführung der Entwicklungsergebnisse in Form von Anwendungssoftware in den Gebrauch ist die häufigste Verwendung der Modellierungsergebnisse. Die organisationszentrische Sichtweise in der sprachbasierten Informatik (vgl. Abb. 12, S. 41) eröffnet eine weitere Verwendungsmöglichkeit in Richtung Human Ressource Management. Die organisationszentrische Sicht berücksichtigt neben den umfangreichen Tätigkeiten der Anwendungsentwicklung zusätzlich organisations- und projektspezifische Aufgaben, wie z.B. eine Verbesserung und Optimierung von Arbeitsabläufen oder eine Wandlung der Systemlandschaft in Richtung Serviceorientierung. Es handelt sich dabei um Aktivitäten, die, wenn sie in ein Entwicklungsprojekt integriert werden, zu zusätzlichen Synergien aus Sicht der Anwenderorganisation führen können.

Die organisationszentrische Vorgehensweise stellt die Anwenderorganisation in den Mittelpunkt. Die zugehörigen Arbeiten sollten weitestgehend auf Seite der Anwenderorganisation vorgenommen werden. Diese Verlagerung von Entwicklungsarbeiten in die Anwenderorganisation kommt der Integration des Usability Engineering insofern entgegen, als die Entwicklungsarbeiten „näher“ beim Anwender stattfinden und die benutzerorientierte Arbeit dadurch wesentlich erleichtert wird.

Die Analysearbeiten zu Beginn eines Entwicklungsprojektes bilden den ersten Schwerpunkt im organisationszentrischen Vorgehen. Dabei geht es u.a. um die Umsetzung des Prinzips „Applications follow processes“ mit einer Orchestrierung von Menschen, Abläufen und Systemen im Unternehmen innerhalb des Rahmens von PROCEM®. Ausgangspunkt für die Orchestrierung ist die statische Aufbauorganisation einer Organisation. Die Erhebung der Aufbauorganisation zielt bereits auf die Nutzung des künftigen Anwendungssystems ab, z.B. indem sämtliche relevanten Stakeholder und deren Rollen identifiziert werden. Kernpunkt und Zentrum der Analysearbeiten ist jedoch die Ablauforganisation und damit verbunden die Prozessrekonstruktion. Rekonstruktion ist hier zu verstehen als eine „Nach-Konstruktion“ von bereits bestehenden Arbeitsabläufen bei gleichzeitiger Verbesserung dieser. Mit Verbesserung sind Harmonisierungen von Abläufen über mehrere Organisationeinheiten ebenso gemeint wie Optimierungen von Arbeitsabläufen. Aus Sicht des Menschen gibt es drei Möglichkeiten für eine Optimierung der Arbeitsabläufe (Ghani et al., 2007):

- Reduktion der Arbeitsbelastung durch Automatisierung.
- Unterstützung des Menschen bei der Verrichtung der Arbeit, z.B. durch interaktive Systeme.
- Verbesserung der Qualifikation der Mitarbeiter.

In der organisationszentrischen Vorgehensweise werden alle Optimierungspotenziale identifiziert und erfasst, also auch jene, die keinen Systemzusammenhang erwarten lassen. Das Vorgehen basiert auf dem Konzept der Rekonstruktion von Wissen (Ortner, 2005; Ghani et al., 2007; vgl. Abschnitt 3.2, S. 104ff). Die dargestellten Abläufe umfassen den gesamten Arbeitsbereich eines Benutzers. Es sind manuelle Arbeitsschritte gleichermaßen skizziert wie automatisierte Arbeitsschritte und solche, die später mit Software unterstützt werden sollten. Die Arbeitsschritte werden prozessorientiert oder zweckorientiert gruppiert. Damit verfügen wir über modulare Beschreibungen von Tätigkeiten, die im Unternehmen zu verrichten sind. Ortner (2008b) spricht auch von „Arbeit als Produkt".

Die Rekonstruktion führt in der Regel zu Aussagen über Prozesse, Arbeitsschritte, Ressourcen, Partizipanten an Prozessen und Arbeitsschritten sowie über Daten und Datenflüsse. Die Typisierung der Aussagen und die Grenzen zwischen den Ergebnistypen sind nicht immer scharf zu ziehen. Adäquate Darstellungsmethoden (vgl. Abschnitt 3.3, S. 134ff) erlauben es, mit vergleichsweise wenigen Sprachelementen komplizierte Arbeitsabläufe in einer Form darzustellen, die auch für Modellierungslaien noch gut verständlich ist. Die Kommunikation mit den Benutzern wird auf Basis solcher Diagramme nicht gefährdet. Eine Alternative dazu wäre die Verwendung von Werkzeugen wie BPMN to Text (TECHNUM, 2009).

Die Beschreibungen aus der organisationszentrischen Vorgehensweise werden zur Spezifikation von Anwendungssoftware verwendet oder um den Einsatz von Services in die Wege zu leiten. Arbeit als Produkt kann nicht nur automatisiert werden (= Services), sondern auch z.B. als (teil-)manuelle Dienstleistung angeboten werden. Gleichzeitig fließen diese Ergebnisse in den Bereich des Human Capital Managements ein. Eine Stellenbeschreibung als Rekonstruktionsergebnis repräsentiert die Darstellung von Know-how, welches die Perspektiven von Mitarbeitern (Mensch), Aufgaben (Organisation) und Technik (Anwendungssysteme) umfasst. Sie kann somit als integratives Element zwischen diesen Bereichen aufgefasst werden und es erfolgt ein Wandel in der Organisation hin zum Wissens- bzw. Könnens-Management (Ability-Management). Ein Teil dieses Wissens ist für die Stützung der Nutzer relevant. So kann für jeden Mitarbeiter der individuelle Know-how-Kern (als Sollzustand) herausgearbeitet werden. Darauf aufbauend werden weitere Maßnahmen zur Gestaltung der Mikro-Organisation und zur gezielten Mitarbeiterqualifizierung im Sinne des Aspekts des Lernens nach dem DarWin-Konzept von Heinemann (2006) gesetzt. Zur

detaillierten Durchführung der weiteren personenbezogenen Maßnahmen hinsichtlich Qualifikation kann z.B. auf das Vorgehensmodell von Fink (2000, S. 31) als konkreter Ansatz in Richtung Könnens-Management zurückgegriffen werden. Damit ist die Rückkoppelung der Rekonstruktionsarbeit zur Aufbauorganisation gegeben. Diese ist Voraussetzung für eine prozessorientierte, elastische Aufbauorganisation im Sinne von Hamel & Välikangas (2003).

3.4.4 Integration des Usability Engineering im Rahmen der Stützung der Nutzer

Eine weitere Gelegenheit zur direkten und systematischen Einflussnahme auf Usability und Akzeptanz ist im Rahmen der Stützung der Nutzer für den Gebrauch eines Systems gegeben. Die Einflussnahme ist ergänzend zur Rekonstruktion oder unabhängig von ihr möglich. Die Grundlagen einer Integration des Usability Engineering im Rahmen der Stützung der Nutzer sind jene der semantischen Integration, wie sie auch für die Integration in der Rekonstruktion auf der Sprachebene des Anwendungsbereichs gelten (vgl. Abschnitt 3.2.5, S. 132f). Damit der Übergang von der Sprachebene vom Standpunkt der Informatik auf die Sprachebene vom Standpunkt des Anwendungsbereichs gelingen kann, ist ein Wissensmanagement wie in Abschnitt 3.4.2 (S. 141ff) beschrieben Voraussetzung. Das Wissensmanagement, technisch unterstützt durch ein Repository, gewährleistet eine nahtlose semantische Integration sowohl des Usability Engineering als auch des Acceptance Engineering. Mayhew betonte bereits 1999 die daraus zu erwartende Produktivität (1999a, S. 431 und Synopsis).

Wesentlich für die Stützung der Nutzer ist eine Förderung der Aufmerksamkeit für die (Arbeits)Aufgaben, die einem Anwendungssystem zugrunde liegen. Im Kontext der Aufgabenerledigung ist auch die Herstellung der Balance zwischen den Herausforderungen für den Benutzer durch das System und den Fähigkeiten des Benutzers zu bewältigen. Weiters gilt es, die gegebene Verbindung von Kognition und Emotion zur Kenntnis zu nehmen und den Benutzer nicht als rein kognitiv ausgerichteten, nutzenmaximierenden Informationsverarbeiter zu verstehen. Emotionale Prozesse und Gefühlszustände im Rahmen der Nutzung (z.B. Verwirrung, Zweifel, Zuversicht, Enttäuschung, Klarheit, Optimismus, Besorgnis, Unsicherheit) dürfen nicht ignoriert werden. Es sind also im Rahmen der Stützung auch vermehrt emotionale und soziale Aspekte zu berücksichtigen. Die Beeinflussung der emotionalen Aspekte begründen eine Änderung der Emotional Usability. Diese kann indirekt und ggf. im Vorfeld bereits durch Einflussnahme über die physiologische und die epistemologische Dimension von Usability gesteuert werden (vgl. Abschnitt 2.4.2, S. 62ff).

Die erwähnten Gefühls- und Verhaltenskomponenten stehen nicht klar in Verbindung mit dem Begriff der Usability. Sie stehen jedoch in durchaus in Verbindung mit den

Einflussfaktoren für Akzeptanz (in Anlehnung an Venkatesh et al., 2003). Grundsätzlich ist Sensibilität dafür angebracht, dass Akzeptanz in den verschiedenen Entwicklungsphasen unterschiedliche Gestalt annehmen kann und ein einmal erreichter Akzeptanzlevel angemessener Aufmerksamkeit bedarf, um nicht wieder zu sinken. Weder eine Einschränkung des Akzeptanzbegriffs auf Kauf- und Nutzungsakt (Kollmann, 2000, S. 68) noch die Beschränkung auf den Entwicklungsprozess allein ist für die Stützung sinnvoll. Für die Entwicklung von Anwendungssystemen besteht der Anspruch die Akzeptanzbetrachtung auf den gesamten Produktlebenszyklus auszuweiten. Die *Akzeptanz* ist als dynamisches Konstrukt mit unterschiedlicher Intensität und Dynamik über den gesamten Lebenszyklus des Anwendungssystems zu beobachten (Monitoring), damit bei Bedarf gehandelt werden kann. Es wird dabei auf die Akzeptanzfaktoren aus dem UTAUT-Modell zurückgegriffen. Das *Mapping* der Phaseneinteilung von Kollmann (2004, S. 74 u. S. 143) ist im Rahmen der strukturellen Integration ebenfalls in Betracht zu ziehen, auch wenn seine Phaseneinteilung nicht auf Entwicklungsprozesse oder eine Lebenszyklusbetrachtung ausgerichtet ist.

Die systematisch-methodische Grundlage zur Beeinflussung von *Akzeptanz*, damit ist auch die Beeinflussung von Usability eingeschlossen, in einer frühen Phase der Entwicklungsarbeit ist Voraussetzung, um mit der Messung des Einflusses verschiedener Akzeptanzfaktoren (z.B. nach Venkatesh et al., 2003) zu beginnen und in den verschiedenen Phasen der Entwicklungsarbeit fortzuführen. Das gilt auch für die gezielte Analyse der gemessenen Ergebnisse zur weiteren Verwendung in der Praxis. Die Messung der Wirkung einer Einflussnahme auf Akzeptanz und Gebrauchstauglichkeit ist nicht Gegenstand dieser Arbeit und bleibt weiterführenden Studien vorbehalten.

Für die Benutzer entfällt in der Regel die Möglichkeit, sich für ein Anwendungssystem zu entscheiden. Die Bereichsleiter bzw. die Geschäftsleitung entscheidet und die Benutzer werden unmittelbar mit der Produktnutzung konfrontiert. Spätestens bei der Stützung der Nutzer geht es jedoch um die Bildung einer Produkteinstellung. Diese Einstellung des Nutzers muss sich dabei nicht nur auf das Anwendungssystem beziehen. Es werden auch anwendungsbereichsspezifische Wahrnehmungen, wie z.B. eine wahrnehmbare kollektive Ablehnung einer Gruppe von Mitarbeitern, mit einfließen (Dethloff, 2004, S. 219).

Lernprozesse im Rahmen der Stützung nehmen Einfluss auf *Akzeptanz*, Duldung, Indifferenz, Ambivalenz, Reaktanz, Frustration oder Nicht-Akzeptanz. Durch Vorgehensweisen der sprachbasierten Informatik mit nahtlos integrierten Aktivitäten zum Usability Engineering können solche Lernprozesse bereits im Vorfeld der Stabilisierung, d.h. ab den frühen Entwicklungsphasen, begünstigt werden. Zudem ist im Rahmen der Stützung insbesondere die Berücksichtigung unterschiedlicher Typen von Benutzern im Hinblick auf ihr Innovationsbedürfnis und ihre Innovationsneigung nö-

tig. Dies können Entdecker, Ablehner, Optimisten, Überforderte (hohes Durchschnittsalter und geringes Bildungsniveau), Pragmatiker oder Indifferente sein. Es empfiehlt sich, die Benutzer vor den Schulungen in dieser Hinsicht zu kategorisieren. Ein weiterer Aspekt sind die Befürchtungen von Anwendern, bei Nichtübernahme „auf der Strecke zu bleiben“ bzw. „abgehängt zu werden“. Diese Befürchtungen verstärken sich mit zunehmender tatsächlicher Aussetzung von Innovationsübernahmen (Häufigkeit). Der wahrgenommene Rückstand auf das soziale System (Anwenderbereich) wächst dadurch und führt auf Dauer zu resignativem Verhalten bei Anwendern (Dethloff, 2004, S. 218 ff).

3.5 Ergänzende, strukturelle Integration des Usability Engineering

3.5.1 Integrationsbelange im Überblick

In umfangreichen Untersuchungen und Workshops im Zeitraum von 1997 bis 2004 (z.B. van Harmelen & Wilson, 1997; van Harmelen, et al., 1997; Artim, et al., 1998; Seffah & Hayne, 1999; Gulliksen, Lantz & Boivie, 1998; Nunes & e Cunha, 2000; Gulliksen, Lantz & Boivie, 2001; Kazman, Bass & Bosch, 2003; John, Bass, Kazman & Chen, 2004) wurden Lücken im Software Engineering im Hinblick auf die Mensch-Computer-Interaktion identifiziert und die Bedeutung erkannt, sich damit zu befassen. In diesen Arbeiten kristallisierten sich vielfältige Integrationsbelange heraus. Einige davon haben Seffah et al. (2005, S. 39-40) zusammengefasst:

1) Ausweitung des Gegenstands des Software Engineering auf die bessere Beschreibung und Präzisierung von Benutzerschnittstellen, vergleichbar mit der Anreicherung von Use Cases mit Aufgabenbeschreibungen (Cockburn, 1997; Constantine & Lockwood, 1999; Rosson, 1999).
2) Weiterentwicklung von Notationen und Modellen in der objektorientierten Softwareentwicklung (Nunes & e Cunha, 2000; Artim et al., 1998; Kruchten, 1999; da Silva & Paton, 2001).
3) Mögliche Erweiterungen von Methoden der benutzerzentrierten Entwicklung zur Sammlung von Anforderungen durch Beobachtung und Interviews, Ableiten eines Entwurfs aus Szenarien, Anwendungsfällen und Aufgabenbeschreibungen (Rosson, 1999; Paternò, 2001; Benyon & Macaulay, 2002) und die Verwendung von *Personas* (Cooper & Reimann, 2000) als Möglichkeiten Benutzer zu verstehen und zu modellieren.
4) Neue Methodologien zur Entwicklung interaktiver Systeme, wie sie z.B. Nielsen (1993 u. 1995), Mayhew (1999b) und Roberts, Berry, Isensee & Mullaly (1998) vorgeschlagen haben, ebenso wie Ansätze, die bereits be-

stehende Methodologien ergänzen (z.B. Constantine & Lockwood, 1999; Kruchten, 1999).

Zwei dieser von Seffah et al. (2005) dokumentierten Integrationsbelange werden in der vorliegenden strukturellen Integration verfolgt. Zum einen ist das eine Erweiterung des Multipfad-Vorgehensmodells als Ansatz zur Ergänzung bestehender Methodologien im Hinblick auf die benutzerorientierte Entwicklungsarbeit. Diese Ergänzung erfolgt auf konzeptioneller Ebene, im vorliegenden Fall jedoch ohne Berücksichtigung der technischen Aspekte (Tab. 1, S. 7). Zum anderen wird die Verbindung zu einer praktischen Integration auf externer Ebene angestrebt. Diese soll durch Erweiterung von Methoden der benutzerzentrierten Entwicklung, im Sinne von Punkt drei der angeführten Integrationsbelange erreicht werden. Über die semantische Integration in der Rekonstruktion und der Stützung der Nutzer ist für beide verfolgten Integrationsbelange die Rückkoppelung zur internen Ebene sichergestellt. Die explizite Forderung nach einer semantischen Integration fehlt bei Seffah et al.

Vom heutigen Stand der Technik, der betrieblichen Organisationslehre und unserer Kenntnis vom arbeitenden Menschen aus gesehen, sollten sich die Entwicklung, das Management und damit verbunden der Betrieb von Anwendungssystemen (z.B. ERP-Systemen) als ein dynamischer Prozess aus den Unternehmen selbst heraus vollziehen. Hamel & Välikangas (2003) brachten dies sehr treffend zum Ausdruck, als sie ein solchermaßen funktionierendes Unternehmen „The Resilient Enterprise“, also das „elastische oder spannkräftige Unternehmen“, nannten. Das Streben nach Elastizität, „The Quest for Resilience“, wie es von Hamel & Välikangas (2003) für den strategischen Bereich gefordert wird, soll für die benutzerorientierte Entwicklungsarbeit mit der systematischen Integration des Usability Engineering über mehrere Ebenen (vgl. Tab. 1, S. 7) bewerkstelligt werden. Die Integration über alle Ebenen soll einen elastisch gestaltbaren Entwicklungsprozess ermöglichen.

Die Softwareentwicklung kann nicht alle Anforderungen im Laufe des Entwicklungsprozesses vorwegnehmen bzw. berücksichtigen. Auch die Frequenz bzw. Häufigkeit von notwendigen Änderungen kann oft nicht reduziert werden. Ein starrer Prozess kann der Vielfalt an Aufgaben, wie sie für die Anwendungsentwicklung gegeben ist, nicht gerecht werden. Der geforderte elastische Prozess soll biegsam und zugleich spannkräftig sein im Sinne von anpassbar (adaptiv) an neue Aufgaben, an verschiedene Menschen und Organisationen, und auch an neue technologische Herausforderungen. Der Entwicklungsprozess muss für unterschiedliche Projekte elastisch gestaltet werden können und gleichzeitig innerhalb eines vorgegebenen Rahmens (z.B. ProCEM®) systematisch, nachvollziehbar und bei Bedarf auch wiederholbar sein. Die geforderte Elastizität innerhalb eines durchdachten Rahmens dient nicht nur der Anpassung von Vorgehensmodell und Methoden an eine spezifische Projektsituation.

Sie soll es ermöglichen, situativ Konzepte zu integrieren, die derzeit noch außerhalb der Grenzen der Softwareentwicklung liegen, wie z.B. das *End User Development* (Liebermann, Paterno & Wulf, 2006).

Die Einführung von Anwendungssystemen in Unternehmen ist immer mit einem Veränderungsprozess verbunden. In Anbetracht der vielen möglichen Spielarten von Entwicklungsprojekten über den gesamten Systemlebenszyklus soll das Integrationskonzept auf kein bestimmtes Veränderungsmodell (von Rosenstiel, 2007, S. 455, mit Verweis auf Reiß, 1997) abgestellt werden. Sollte in der Praxis die Unterlegung eines solchen Modells von Nutzen sein, so hängt die Wahl der Methode der Veränderung in starkem Maße von den impliziten Menschen- und Organisationsbildern der Entscheider (Topmanagement oder hinzugezogene Berater) ab. Erfolgreiche Veränderungen müssen immer von der Spitze einer Organisation getragen werden. Das obere Management muss hierbei die Führerschaft übernehmen und die entsprechenden Ressourcen bereitstellen. Außerdem müssen die Veränderungen von den Mitarbeitern mit-getragen werden, die hierzu wiederum das notwendige Verständnis für den Prozess und die angestrebten Veränderungsziele besitzen müssen. Da Veränderungsprozesse generell mit Widerständen versehen sind, kann bei „doppelten Veränderungen“ (im Sinne von Battle & Lockheed, 2005; vgl. Abschnitt 3.5.2, S. 156ff) mit doppelten Schwierigkeiten gerechnet werden. Diese Tatsache unterstreicht die Forderung nach einer Integration des Usability Engineering und des-Acceptance Engineering in den Entwicklungsprozess und zwar derart, dass die Aktivitäten aus diesen Engineering Bereichen für die Anwender und die Anwenderorganisation im Projekt als Einheit erscheinen.

Wesentliche Aktivitäten im Hinblick auf die Erreichung von *Akzeptanz* und Gebrauchstauglichkeit passieren nicht auf der Herstellerseite (Auftragnehmerseite), sondern sind eng an Aktivitäten auf der Auftraggeberseite gebunden (Abb. 13, S. 42). Damit ist eine Erweiterung des Multipfad-Vorgehensmodells zu fordern. Diese Forderung geht mit einem Integrationsgesichtspunkt von Seffah et al. (2005) konform.

3.5.2 Bestehende Integrationsansätze

Die in Abschnitt 2.4.3 (S. 66ff) erwähnten oder eingeführten Konzepte und Modelle verfügen punktuell bereits über Ansätze zur strukturellen Integration von Usability Engineering in die Anwendungsentwicklung. Diese Integrationsansätze äußern sich meist in Teilbereichen eines Konzepts bzw. in Details zu hinterlegten Vorgehensweisen. Als Beispiel kann hier das Konzept des Agile Usage Centered Software Life-Cycle (AUCSL) von Gundelsweiler et al. (2004) angeführt werden. Weitere Beispiele sind XP+UE+XU unter dem Titel „eXtreme Usability“ von Holzinger & Slany (2006) und CRUISER (A Cross Discipline User Interface and Software Engineering LifeCycle) von

Memmel, Gundelsweiler & Reiterer (2007b). Alle genannten Konzepte unterliegen dem agilen Entwicklungsparadigma.

Pyla, Pérez-Quinones, Arthur & Hartson (2005) schlagen demgegenüber ein ereignisgesteuertes Framework zur Integration von Usability Engineering und Software Engineering LifeCycles vor. Auch die von Gulliksen, Göransson, Boivie, Persson, Blomkvist & Cajander (2005, S. 26-29) vorgeschlagenen zwölf Prinzipien zur Einführung eines benutzerzentrierten Entwicklungsprozesses sollten hier erwähnt sein. Zu guter letzt soll das Konzept von Battle & Lockheed genannt und kurz skizziert werden.

Battle & Lockheed (2005) gehen in ihrem Ansatz von drei Phasen auf der Zeitachse der Entwicklung aus, der frühen, der mittleren und der späten Phase (Abb. 33, S. 157, in Anlehnung an Battle & Lockheed, 2005, S. 304). Dieser Art finden sie eine „salomonische" Lösung zur ansonsten heterogenen Benennung von Entwicklungsphasen in unterschiedlichen Phasenmodellen.

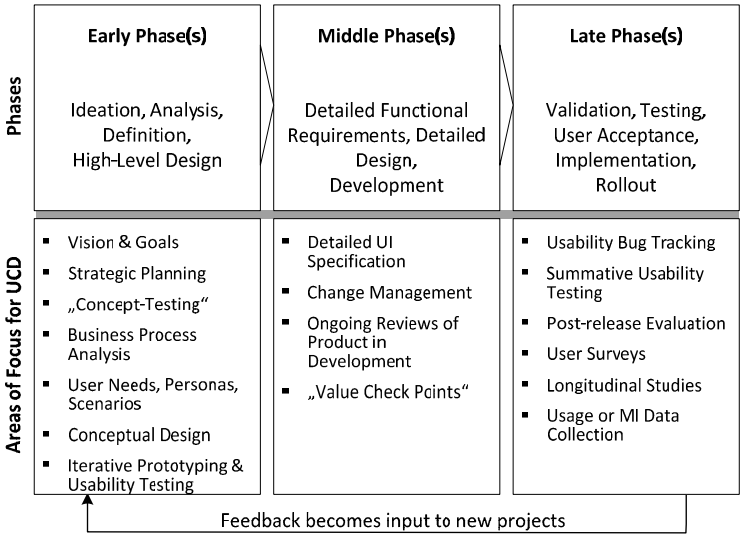

Abb. 33: UCD-Fokus in den Entwicklungsphasen nach Battle & Lockheed

In den einzelnen Phasen weisen sie im Zusammenhang mit den aufgelisteten und zugeordneten Methoden auf die jeweils spezifischen Herausforderungen hin. Erwähnenswert beim Konzept von Battle & Lockheed ist der Hinweis auf die Erkenntnis, dass ein Veränderungsprozess an sich bereits eine große Herausforderung für Organisationen darstellt. Zusätzlich noch einen UCD-Prozess einzuführen erscheint den meisten Organisationen zu schwierig. Aus ihren Ausführungen lässt sich schließen,

dass die Einführung neuer Systeme in Verbindung mit der Einführung eines User Centered Design von den Anwenderorganisationen derzeit als „doppelte Veränderung“ wahrgenommen wird.

Um diesen generellen Herausforderungen zu begegnen schlagen Battle & Lockheed vor, auf verschiedenen Ebenen und in kleinen Schritten und Aktivitäten eine bessere Benutzerfokussierung zu unterstützen, z.B. auch mit der Verwendung einer „organisationseigenen Sprache und *Terminologie,* und zwar in der Form „Adopt the organization’s own language/terminology if possible rather than forcing new terms and work practices“ (Battle & Lockheed, 2005).

3.5.3 Zusammenführung ausgewählter Prozessmodelle

3.5.3.1 Einleitung

Die Zusammenführung von Anwendungsentwicklung und Usability Engineering auf konzeptioneller Ebene erfolgt in Form eines *Mapping*. Als Integrationsgegenstand auf dieser Ebene werden vorerst ausgewählte Prozessmodelle der beteiligten Disziplinen herangezogen. Die relevanten Prozessmodelle wurden bereits in den Abschnitten 2.3.3 (S. 41ff), 2.4.3 (S. 66ff) und 2.4.6 (S. 85ff) eingeführt. Das Multipfad-Vorgehensmodell nach Ortner (2005; Abschnitt 2.3.3, S. 41ff) wird als sprachbasiertes, ganzheitlich ausgerichtetes Vorgehensmodell der Anwendungsentwicklung als führendes Modell zur Integration herangezogen. Die Grundlegung des Modells in der sprachbasierten Informatik ist ausschlaggebend für die Auswahl. Der Gestaltungsrahmen für den Usability Engineering Prozess ist ausdrücklich nicht als Prozessmodell definiert (DATech, 2008), orientiert sich aber an grundlegenden Phasen der Entwicklungsarbeit und differenziert Aufgaben und Verantwortungen zwischen Auftraggeber- und Auftragnehmerorganisation (vgl. Abschnitt 2.4.3, S. 66ff). Das Modell ist zwar nicht sprachbasiert ausgerichtet, es kann aber als ganzheitlicher Ansatz bezeichnet werden, der zudem ausreichend offen ist, um Gestaltungsspielraum für die Integration zu geben. Allein durch diese Charakteristika erscheint er tauglich, eine konstruktive Zusammenführung der Modelle zu gestatten und wesentliche Aspekte des Usability Engineering einbringen zu können. Als relevant erachtete Prozess- bzw. Faktorenmodelle aus der Akzeptanzforschung (UTAUT-Modell und Akzeptanzmodell nach Kollmann; vgl. Abschnitt 2.4.6, S. 85ff) runden die Modellauswahl für das Prozessmapping ab.

Der Schwerpunkt der Zusammenführung liegt auf den frühen Entwicklungsphasen und bei der Anwenderorganisation. Zur anschaulichen Gestaltung des Prozess-Mapping werden die Modelle begleitend in grafischen Darstellungen angenähert und zweckorientiert vereint. Die *Terminologie* der Benennung von Tätigkeiten und Vor-

gehensweisen kann im Usability Engineering eine andere sein als in der Anwendungssystementwicklung. Eine eventuell erforderliche Klärung von Benennungen erfolgt begleitend und wird im Glossar dokumentiert. Die grafischen Darstellungen betreffen den gesamten Lebenszyklus und bieten damit eine Basis für Integrationsaktivitäten, die über die frühen Entwicklungsphasen hinausgehen.

3.5.3.2 Annäherung des Gestaltungsrahmens für den Usability Engineering Prozess an das Multipfad-Vorgehensmodell

Für die Integration des Usability Engineering in die Anwendungsentwicklung wurde das Multipfad-Vorgehensmodell als führendes Modell ausgewählt. Dies erfolgte einerseits aufgrund der Ausführungen und Bewertungen des Modellvergleichs von Hildenbrand, Behm, Rashid & Geisser (2006) und zudem unter Beachtung der Maßgabe, dass sowohl das sprachkritische Paradigma als auch die ganzheitliche Entwicklungsarbeit in diesem Modell bereits sehr ausgeprägt sind. Eine durchgängige Werkzeugunterstützung liegt (noch) nicht vor, was sowohl als Vorteil wie auch als Nachteil gewertet werden kann. Ein Vorteil ist, dass das Modell offen ist für den Einsatz unterschiedlicher Tools (z.B. Open-Source-Produkte zur Unterstützung der Modellierung und zur Dokumentation).

Im Zuge der anwendungsorientierten Beschreibung der einzelnen Phasen bzw. Vorgangstypen des Multipfad-Vorgehensmodells wurde bereits Integrationspotenzial deutlich. Im Besonderen betrifft dies die Aufgaben auf Seiten der Anwenderorganisation und die Übergänge von der Anwenderorganisation zur Herstellerorganisation zu Beginn der Entwicklung und im Zuge der Einführung eines Anwendungssystems in umgekehrter Richtung.

Die Notwendigkeit der Benutzerbeteiligung und wichtige Aspekte dazu sind in der Beschreibung der Phasen des Multipfad-Vorgehensmodells erwähnt. Das Modell orientiert sich an den Verhältnissen in der Praxis was die Problembeschreibung von Seiten der Anwenderorganisation betrifft. Diese Problembeschreibungen sind oft sehr knapp und wenig präzise. Der bisher verfolgte Ansatz, diese Defizite mit Aufgaben in den Phasen der Voruntersuchung und der Stabilisierung - beide sind überwiegend der Herstellerorganisation zugeordnet - auszugleichen, erscheint nicht ausreichend und wurde aus Sicht der sprachbasierten Informatik wiederholt unterstrichen (vgl. Abschnitte 3.2, S. 112ff u. 3.4, S. 139ff). Zudem entsteht in der Beschreibung der Eindruck, dass die Organisationsmodellierung erst im Zuge der Stabilisierung erfolgen soll. Dies würde einem in der Praxis üblichen Vorgehen entsprechen, das Anbieter von Standardsoftware einsetzen. Die Organisationsmodellierung bestünde dann darin, die Prozesse des Unternehmens (ein Stück weit) an die Prozesse der Standardsoftware anzupassen. Durch die Hinterlegung einer Schachbrettmetapher (vgl. Abschnitt 2.3.3.1, S. 41ff) kann diese Argumentation entkräftet werden.

Die Forderung von Anwenderorganisationen an Hersteller nach besseren und kostengünstigeren Lösungen ist bei genauerer Betrachtung gleichzeitig eine Forderung der Anwenderorganisationen an sich selbst. Diese Forderung impliziert für die Anwenderorganisation, die Organisationsmodellierung und Benutzerbeteiligung weitgehend selbst in die Hand zu nehmen und zu forcieren. Dazu ist wiederum eine Erweiterung des Multipfad-Vorgehensmodells erforderlich.

Ein weiterer Integrationsaspekt ist die „baukastenorientierte" Entwicklung von Anwendungssystemen. Diese wird durch das Multipfad-Vorgehensmodell unterstützt. Ortner (2005, S. 141) skizziert diese Baukastenorientierung sprachbasiert und stellt grundlegend auch die Übergänge zur Anwenderorganisation dar. Mit der entstehenden Transparenz sollte jegliche Art von Schnittstellen (zwischen Mensch und Maschine und umgekehrt) besser adressiert werden können und sie soll auch der laufenden Optimierung von Abläufen im Unternehmen dienen. Eine baukastenorientierte Entwicklung unterliegt gewissen Regeln. Die Regeln sind aufzustellen, zu administrieren und verfügbar zu machen (z.B. in einer SOA-Governance). Von den Beteiligten (Rollen) sind die Regeln situativ anzuwenden. Die Regeln begrenzen Möglichkeiten und Dynamik des Baukastensystems. Dementsprechend funktioniert die Baukastenorientierung vergleichbar einem Schachspiel. Die Schachbrettmetapher (vgl. Abschnitt 2.3.3.1, S. 41ff) fungiert als dynamisches Element des Baukastensystems.

Das Multipfad-Vorgehensmodell (vgl. Abschnitt 2.3.3, S. 41ff) und der Gestaltungsrahmen für den Usability Engineering Prozess (vgl. Abschnitt 2.4.3, S. 66ff) werden angenähert. Im Zuge dessen gilt es, die einzelnen Phasen der Modelle unter Berücksichtigung der vorgenommenen Abgrenzungen des Integrationsvorhabens abzugleichen. Zur Verbesserung der Transparenz dieses Schritts wird die Annäherung grafisch unterstützt (Abb. 34, S. 161).

Der Entwicklungsprozess auf Seiten der Herstellerorganisation umfasst alle wesentlichen Phasen der Entwicklungsarbeit und macht die Verfolgung der (Haupt)Pfade deutlich. Das Multipfad-Vorgehensmodell zielt nicht auf die Entwicklung von bestimmten Systemtypen auf Basis bestimmter Architekturen ab. Es kann für jeden Anwendungssystemtyp herangezogen werden. Für jeden Vorgangstyp werden konkrete Methoden empfohlen und durch entsprechende Beschreibung konkretisiert (Ortner, 2005).

Die Abbildung des Entwicklungsprozesses geschieht im Gestaltungsrahmen für den Usability Engineering Prozess in groben Zügen. Ließe sich von der Verteilung der Phasen auf die Wichtigkeit der Aufgaben schließen, würde bereits beim Phasenvergleich deutlich, dass die Arbeiten auf Seiten der Auftraggeberorganisation im Usability Engineering ungleich ausgeprägter sind, als dies im Multipfad-Vorgehensmodell der Fall

ist. Die explizite Erweiterung bzw. Verfeinerung des Multipfad-Vorgehensmodells bietet sich in dieser Hinsicht erneut an.

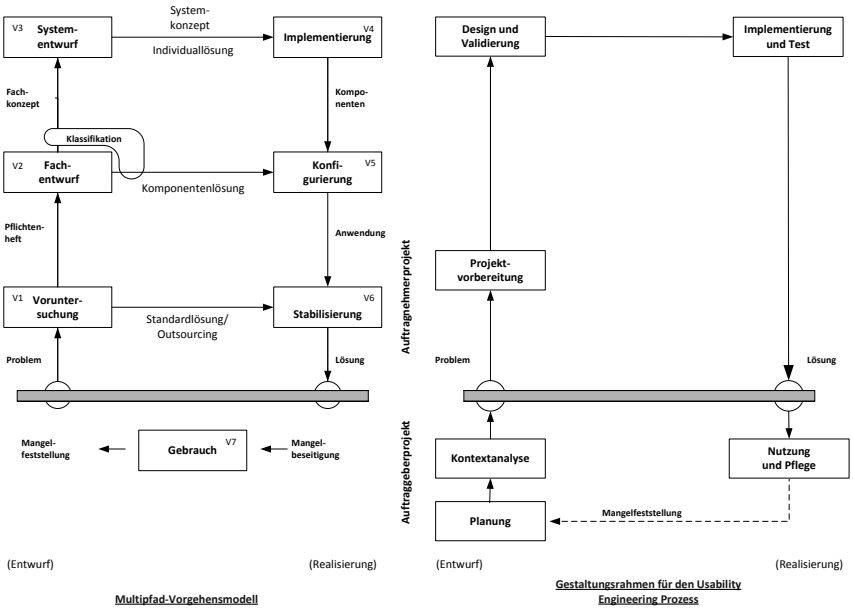

Abb. 34: Annäherung des UE-Gestaltungsrahmens an das Multipfad-Vorgehensmodell

Hersteller von Standardsoftware verfügen zum Zeitpunkt der Herstellung (noch) nicht über Kenntnisse der tatsächlichen Einsatzumgebung und sind natürlich bemüht, durch entsprechende Kontakte, diesen fehlenden Bezug zu kompensieren. Da der Bezug nicht ausreichend herzustellen ist, werden „potentielle“ Abläufe und Gegebenheiten als „Standard“ in den IT-Lösungen implementiert, was aus Sicht der Anwender einer hypothetischen Entwicklung gleichkommt. Hier ist eine Lücke zwischen Hersteller und Anwender immanent. Diese trägt u.a. wesentlich zum Scheitern vieler Projekte bei bzw. kann Projekte erheblich verzögern und verteuern. Diese Lücke gilt es im Zuge der Modellerweiterung genauer zu betrachten und womöglich zu schließen.

Die Erweiterung des Multipfad-Vorgehensmodells auf Seiten der Anwenderorganisation wirkt sich insbesondere auf die Gestaltung der Vorgangstypen Voruntersuchung und Stabilisierung aus, wenngleich es nicht möglich und auch nicht zielführend ist, scharfe Grenzen zu ziehen. Die methodische Unterstützung der Anwenderorganisation, die im Zuge einer solchen Erweiterung notwendig ist, hat den Zwecken und Maßgaben der Unternehmensmodellierung und der Verbesserung von *Akzeptanz* und

Gebrauchstauglichkeit zu folgen. Durch die Verlagerung von Teilen der Entwicklungsarbeit auf die Seite der Auftraggeberorganisation kann die Entwicklung auf Seiten der Herstellerorganisation an Qualität und Potenzial gewinnen, dies betrifft insbesondere den Übergang zwischen den beteiligten Organisationen im Entwicklungsprozess. Im Zuge einer möglichen Modellerweiterung ist jedoch darauf zu achten, dass sich die „Baukastenorientierung“ auch in der Anwenderorganisation fortsetzt und die Übergänge aus dieser Sicht stimmig sind.

Für das Multipfad-Vorgehensmodell dient die Schachbrettmetapher zur Verdeutlichung der darin gegebenen Flexibilität in der Anwendung des Modells. Analog dazu sprechen die Autoren des Gestaltungsrahmens für den Usability Engineering Prozess von Prozessbausteinen, deren Anordnung nicht unbedingt dem vorgegebenen Muster folgen muss. Beide Modelle stellen also eine bei Bedarf variable Anordnung von Vorgangstypen bzw. Bausteinen zur Verfügung. Es kann aus diesem Blickwinkel davon ausgegangen werden, dass bei der Zusammenführung der Modelle keine Einbußen für die geforderte Elastizität des erweiterten Multipfad-Vorgehensmodells entstehen.

Der Gestaltungsrahmen für den Usability Engineering Prozess berücksichtigt Teile des Projektmanagements als Prozessbausteine. Im Multipfad-Vorgehensmodell stellt das Projektmanagement eine Querschnittfunktion dar, welche in das Management der Anwendung eingebettet ist (vgl. Abschnitt 2.3.3, S. 41ff). Projektmanagementaufgaben als Phasen in das Multipfad-Vorgehensmodell zu integrieren erscheint nicht zielführend. Vielmehr sollten Initialaufgaben und auch begleitende Aufgaben des Usability Engineering zusätzlich im Projektmanagement der Anwendungsentwicklung Platz finden. Es handelt sich dabei speziell um die Sensibilisierung für Usability im Vorfeld und zu Beginn eines Entwicklungsprojekts und die Verankerung von Usability in der Unternehmenskultur. Aufgabe des Projektmanagements ist es außerdem, besonders darauf zu achten, dass projektspezifische Festlegungen im Hinblick auf Usability in den Vertragsunterlagen festgelegt werden. Auch geplante Usability Aktivitäten sind dort zu vereinbaren. Dies betrifft Aktivitäten auf Seiten der Anwenderorganisation ebenso wie auf Seiten der Herstellerorganisation, was zu einer klaren Rollenverteilung führt. Die Wahrnehmung der genannten Aufgaben aus dem Usability Engineering durch das Projektmanagement fordert eine adäquate Qualifikation des Projektmanagers.

Die Rollen der Anwenderorganisation und der Herstellerorganisation werden in beiden relevanten Modellen explizit angesprochen. Im Gestaltungsrahmen für Usability Engineering wird dieser Sicht ungleich mehr Aufmerksamkeit entgegengebracht, als dies im Multipfad-Vorgehensmodell der Fall ist. Die Forcierung von nutzerzentrierten Vorgehensweisen bedarf jedoch einer adäquaten Verankerung der „Nutzerseite“ und damit der Anwenderorganisation im Vorgehensmodell. Mit einer Erweiterung des

Multipfad-Vorgehensmodells in Richtung Anwenderorganisation können die zunehmenden Aufgaben der Anwenderorganisation, wie sie sich sowohl in der Rekonstruktion als auch in der Stützung der Nutzer ergeben, adäquat verankert werden.

Der Gestaltungsrahmen für den Usability Engineering Prozess sieht bereits vor Ausschreibung und Auftragsvergabe der Entwicklungsarbeiten umfangreiche Analyseaufgaben auf Seiten des Auftraggebers vor. Damit soll erreicht werden, dass Nutzungs- und Benutzeranforderungen frühzeitig klar sind und hinreichend dokumentiert dem Auftragnehmer mitgeteilt werden können. Im weiteren Verlauf sollte anhand dieser Aufzeichnungen das System abgenommen werden können, d.h. u.a. das System wird anhand dieser *Nutzungsanforderungen* validiert. Die aktuellen Forderungen aus dem Bereich der Anwendungsentwicklung gehen sogar so weit, dass davon gesprochen wird, die Unternehmensmodellierung bzw. Teile davon in die Fachbereiche (d.h. Anwendungsbereiche auf Seiten der Anwenderorganisation) zu verlagern.

Das Multipfad-Vorgehensmodell sieht eine Benutzerbeteiligung in frühen Entwicklungsphasen vor, d.h. von den Voruntersuchungen bis hinein in den Fachentwurf. Eine Verlagerung der Analyse- und Modellierungsarbeiten in die Anwenderorganisation ist im Modell (noch) nicht verankert. Dass es ansatzweise eine solche Verlagerung bereits gibt und sich diese noch weiter entwickeln wird, machen Anwenderorganisationen deutlich, wenn sie den möglichen Mehrfachnutzen von Modellierungsarbeiten für sich reklamieren (vgl. z.B. Abschnitt 2.3.2, S. 37ff). Unternehmen sehen zunehmend Potenzial für sich, die Modellierung und deren Ergebnisse mehrfach nutzen zu können, z.B. um ihre Ablauforganisation zu verbessern, zu optimieren oder zu harmonisieren, als Unterlagen für die Einführung neuer Mitarbeiter oder als Komponenten von Stellenbeschreibungen. Anwenderorganisationen sind daher immer weniger bereit, große Geldsummen für Analysearbeiten durch Herstellerorganisationen zu bezahlen und deren Ergebnisse nicht ausreichend beeinflussen und zudem nicht weiter nutzen zu können. Was Unternehmen mitunter daran hindert, diese Analysearbeiten selbst durchzuführen, ist die erforderliche Qualifikation von Mitarbeitern. Auch wenn die dazu erforderlichen Qualifikationen (vorerst) intern nicht verfügbar sind, so gibt es zunehmend die Möglichkeit und auch das Angebot solche Qualifikationen temporär zuzukaufen, z.B. durch Inanspruchnahme eines freiberuflich tätigen *Enterprise Engineers*.

Das Multipfad-Vorgehensmodell wird um Vorgangstypen erweitert, die Vorarbeiten zu Entwicklungsprojekten auf Seiten der Auftraggeberorganisation vorsehen (Abb. 34, S. 161). In Verbindung mit den neuen Vorgangstypen sind auch adäquate Methoden zur Bewältigung dieser Aufgaben im Sinne des Auftraggebers vorzusehen. Vor den eigentlichen Modellierungsarbeiten gilt es, einen Gesamtrahmen (Ist-Situation) zu erfassen um das Entwicklungsvorhaben grob abgrenzen zu können (Voranalyse –

V0.1). Es handelt sich dabei um eine Ist-Analyse mit Fokus auf das vorab bereits definierte Vorhaben (Projektplanung). Diese Voranalyse bildet den Vorgangstyp V0.1 im erweiterten Multipfad-Vorgehensmodell.

Bei Bedarf können Detailanalysen (V0.2) erforderlich sein. Der Bedarf ergibt sich entweder aus dem Entwicklungsprojekt selbst oder es besteht Änderungsbedarf in der Organisation, der in Verbindung mit einem neuen Anwendungssystem gelöst werden sollte. Dies können z.B. organisationale Veränderungen wie Prozessoptimierungen oder Prozessharmonisierungen über mehrere Teilorganisationen sein. Die Analysearbeiten sollten durch spezielle Erhebungs- und Dokumentationsmethoden unterstützt sein (vgl. Abschnitt 3.5.4, S. 171ff). Für die Detailanalysen wird ein eigener Vorgangstyp definiert. Eine klare Abgrenzung zwischen Vor- und Detailanalysen ist nicht immer möglich.

Die Ergebnisse der Vor- und Detailanalysen münden in eine *Anforderungsspezifikation* der Anwenderorganisation (V0.3). Die Erstellung dieser *Anforderungsspezifikation* wird begleitend zu den Analysearbeiten begonnen. Abhängig von der Komplexität eines Entwicklungsvorhabens kann es erforderlich sein, die Anforderungsspezifikation, z.B. in Ausprägung eines Lastenheftes, in mehreren Iterationen fertigzustellen. Dabei geht es einerseits um die Sicherstellung der Vollständigkeit der Anforderungen, andererseits um die Gewährleistung, dass die dokumentierten Anforderungen auch das repräsentieren, was die effektive und effiziente Unterstützung der Arbeitsprozesse ermöglicht und gleichzeitig auch abbildet, was durch Bedarf und Bedürfnisse gefordert ist. Letzteres trägt zur Zufriedenheit der Anwender bei und ist im Sinne einer Geltungssicherung ein früher Ansatz für die Akzeptanzsicherung.

Ist die *Anforderungsspezifikation* geklärt und von Anwendern und Verantwortlichen genehmigt, kann auf dieser Basis Kontakt mit Anbietern aufgenommen werden. Es gibt unterschiedliche Möglichkeiten der Kontaktaufnahme (z.B. Ausschreibung, gezielte Einladung von wenigen Anbietern, interne Vergabe an die Entwicklungsabteilung oder gesetzlich geregelte Vergabeverfahren). Im Multipfad-Vorgehensmodell ist hierfür ein separater Vorgangstyp vorgesehen (V0.4). Dieser Vorgangstyp leitet die Übergabe der Entwicklungsarbeit an die Herstellerorganisation ein. Das soll nicht heißen, dass Benutzer in späteren Vorgangstypen nicht mehr integriert sind, es soll lediglich bedeuten, dass der „Lead“ des Projektes und die Hauptverantwortung für den Fortgang wechseln. Jene Aufgaben, die im Gestaltungsrahmen für den Usability Engineering Prozess (kurz: DATech Gestaltungsrahmen) in der Phase Kontextanalyse verankert sind, werden in den neu eingeführten frühen Phasen des nunmehr erweiterten Multipfad-Vorgehens-modells auf Seiten des Auftraggebers wahrgenommen und methodisch verankert.

In den Phasen Voranalyse, Detailanalyse, Anforderungsspezifikation und Ausschreibung liegt die Hauptverantwortung bei der Auftraggeberorganisation. In den folgenden Phasen, nach dem klassischen Multipfad-Vorgehensmodell sind das die Phasen V1-V6, liegt die Hauptverantwortung bei der Herstellerorganisation. Die Phasen Design und Validierung sowie Implementierung und Test im DATech Gestaltungsrahmen stehen parallel zu den Phasen V1-V6 des Multipfad-Vorgehensmodells. Letztere sind aus Sicht der Anwendungsentwicklung jedoch wesentlich umfassender und detaillierter als dies die zwei Phasen des DATech Gestaltungsrahmens andeuten. Das Multipfad-Vorgehens-modell wird mit seiner Gestaltung auch neuen Technologien wie z.B. der Serviceorientierung gerecht und die Entwicklung kann auf den Grundlagen der sprachbasierten Informatik innerhalb verschiedener Theorien realisiert werden. Die Aufgaben im Gestaltungsrahmen fokussieren auf die Förderung der Gebrauchstauglichkeit von Systemen. Diese finden im Rahmen der Integration auf Methodenebene (vgl. Abschnitt 3.5.4, S. 171ff) Berücksichtigung. Inhaltsstandards sind im Vorgangstyp Fachentwurf von elementarer Bedeutung und haben integrativen Charakter. Aus dem Bereich des Usability Engineering können z.B. konkrete Normen aus der DIN EN ISO 9241-Reihe im Fachentwurf verankert werden. Mit dieser Zuordnung werden wir der nahtlosen Verankerung von Usability Engineering im Entwicklungsprozess besser gerecht als mit einer Aufgabenbeschreibung auf Ebene der Vorgangstypen.

Nachdem ein Anwendungssystem getestet und für die Anwenderorganisation parametrisiert ist, wie das der Vorgangstyp Stabilisierung aus technischer Sicht vorsieht, wird das System in der Anwenderorganisation eingeführt (roll-out). Keines der beiden betrachteten Modelle sieht (bisher) für die Systemeinführung einen eigenen Vorgangstyp vor. Beim Multipfad-Vorgehensmodell ist die Einführung auf die Phasen Stabilisierung und Gebrauch verteilt. Der DATech Gestaltungsrahmen bildet die Einführung als Teil der Phase Nutzung und Pflege ab.

Spätestens in der Einführungsphase (V7.0) spielen die Benutzer wieder eine der Hauptrollen. Aufgaben und Anforderungen an die Verantwortlichen von Hersteller- und Anwenderorganisation sind vielschichtig und komplex. Viele Dinge laufen parallel ab. Zu den fachlichen Belangen kommen spätestens in dieser Phase auch technische Aspekte das System betreffend. Die tägliche Arbeitsbewältigung der Benutzer muss gewährleistet sein, Schulungen sind vorab bzw. parallel zur Einführung durchzuführen. Meist muss aber auch noch begleitend eine „Feuerwehr“ von Seiten des Herstellers bereit sein, um rasch eingreifen zu können, wenn Schwierigkeiten auftauchen. Diesen Schwierigkeiten gilt es entgegenzuwirken. Gleichzeitig ist abzuschätzen, ob möglicherweise eine Anpassung des Systems zur endgültigen Fehlerbehebung erforderlich ist. Erste Schwächen des Systems werden deutlich. Die Benutzer sollten angehalten werden, diese zu dokumentieren. Ein Feedbacksystem für Benutzer ist einzurichten und laufend auszuwerten. Aus Vorfällen im Zuge der Einführung können erste

Vertragsstreitigkeiten entstehen, was die Situation zusätzlich verschlechtern kann. Im erweiterten Multipfad-Vorgehensmodell wird für die Systemeinführung ein eigener Vorgangstyp eingeführt. Benutzerzentrierte Aufgaben, die auf der Auftraggeberseite durchzuführen sind und bisher dem Vorgangstyp Stabilisierung zugeordnet waren, werden künftig dem Vorgangstyp Einführung zugeordnet. Dies sind insbesondere Aufgaben zur Stützung der Nutzer (vgl. Abschnitt 3.4, S. 139ff). Abb. 35 (S. 166) verdeutlicht die Ergebnisse der Zusammenführung in einem erweiterten Multipfad-Vorgehensmodell.

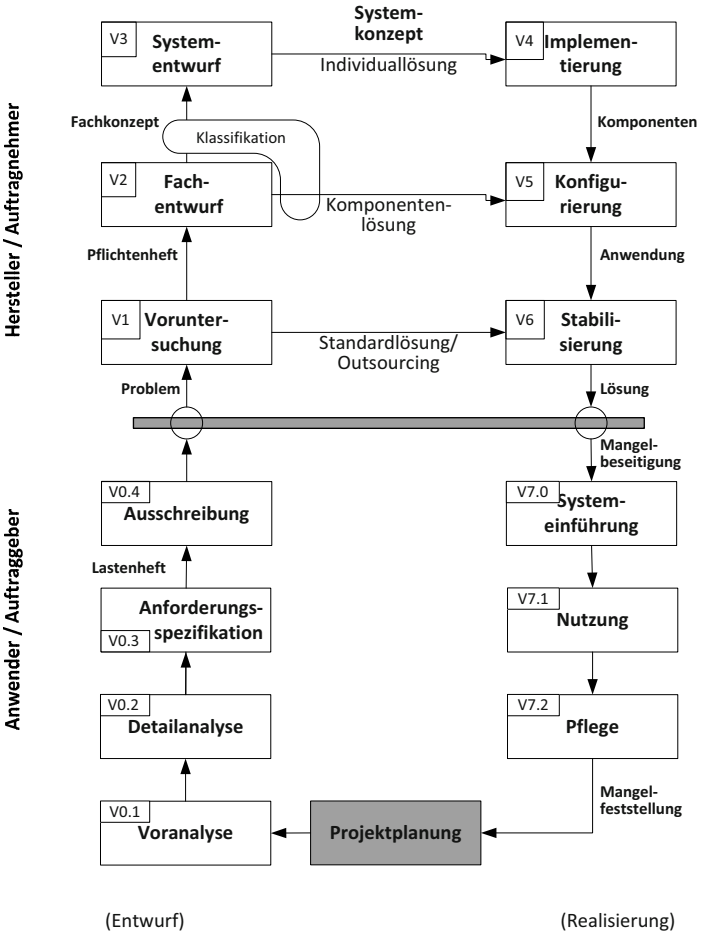

Abb. 35: Erweitertes Multipfad-Vorgehensmodell

Auf die Systemeinführung folgen die Nutzung (V7.1) und die Pflege (V7.2) eines Anwendungssystems. Während dieser Phasen werden Erkenntnisse gewonnen, die der Vorbereitung einer nächsten Entwicklungsstufe des Systems dienen. Die Erkenntnisse sollten systematisch gesammelt werden. Zudem sind Schulungen über die gesamte Nutzungsdauer eines Anwendungssystems nicht zu vernachlässigen. Benutzer nutzen oft nur jene Teile eines Systems, die für ihre Arbeit unmittelbar relevant sind, wichtige Zusammenhänge zu anderen Systemteilen unterliegen einer „Vergessenskurve". Zur Gewährleistung einer optimalen Systemnutzung sind begleitende Schulungen für Mitarbeiter unbedingt erforderlich. Auch die laufende Adaptierung von Handbüchern und sonstigen Benutzerhilfen fällt in diese Phasen. Neue Mitarbeiter sollten eine umfassende Systemschulung von „neutraler" Seite erhalten, damit sich Fehlnutzungen eines Systems, wie sie durch Schulung durch „erfahrene" Mitarbeiter möglich sind, nicht fortpflanzen können. Die Phase Gebrauch im Multipfad-Vorgehensmodell wird umbenannt in „Nutzung". Eine separate Phase für die Pflege wird ergänzt.

Der Übergang von der Nutzung zu einer neuen Entwicklungsstufe des Systems oder zu einem neuen Anwendungssystem wird mit einem „Pseudovorgangstyp" visualisiert. Dieser Vorgangstyp in Gestalt der Projektplanung repräsentiert die bedeutende Rolle des Projektmanagements im Übergang. Auf Basis des nunmehr erweiterten Multipfad-Vorgehensmodells erfolgt die konzeptionelle Integration von Prozessaspekten bzw. Einflussfaktoren aus der Akzeptanzforschung.

3.5.3.3 Akzeptanzmodell und Akzeptanzfaktoren im Multipfad-Vorgehensmodell

Im Rahmen der Anwendungsentwicklung völlig außer Acht gelassen, aber für die Akzeptanz von Anwendungssystemen relevant, ist die Betrachtung von Computern (Technik) als sogenanntes „*evokatorisches Objekt*" durch den Benutzer. Dies sei hier herausgegriffen als ein Aspekt der Wirklichkeitserfahrung im Umgang mit Technik im Allgemeinen und Computern im Besonderen. Ein *evokatorisches Objekt* ist ein Gegenstand, der Gedanken und Assoziationen hervorruft. Der Computer (das Gerät an sich, aber auch das Gerät in Verbindung mit der Software) als ein solches Objekt kann bei Benutzern starke emotionale Reaktionen hervorrufen wie Gefühle, Leidenschaften, Frustrationen aber auch Widerstände, Wünsche und Phantasien (Leithäuser, 1994, S. 76-77). Der Computer ist in solchen Fällen mehr als ein (lebloser) Teil eines Beziehungssystems. Evokationen dieser Art können verschiedenste Auswirkungen für den konkreten Benutzer am Computer haben. Diese Auswirkungen, aber auch die Ursachen für solche Evokationen fallen in den Bereich des Usability Engineering, insbesondere unter die „Emotional Usability" (vgl. Abschnitt 2.4.2, S. 62f) und spielen zweifellos eine Rolle für die Akzeptanz von Systemen (vgl. Akzeptanzfaktoren im UTAUT-Modell; Abb. 24, S. 96).

Das eingeführte UTAUT-Modell als führendes Technologieakzeptanzmodell (vgl. Abschnitt 2.4.6.3, S. 94ff) unterscheidet ablauforientiert zwei Phasen. Die erste Phase befindet sich vor der Nutzung (geplante Nutzung), die zweite Phase repräsentiert die (tatsächliche) Nutzung. Alle Einflussfaktoren des UTAUT-Modells beziehen sich auf diese Phasen. Beide Phasen sind implizit auch im Multipfad-Vorgehensmodell vorhanden, auch wenn dort der Übergang zur Nutzung nicht so klar abgegrenzt wird, wie dies beim UTAUT-Modell der Fall ist. Möglichkeiten zur Verfeinerung des Multipfad-Vorgehensmodells können bei seiner Gegenüberstellung mit dem UTAUT-Modell nicht identifiziert werden. Die systematische Berücksichtigung von Einflussfaktoren hinsichtlich der *Akzeptanz* auf Ebene der Anwender - dieser Ebene entsprechen die Untersuchungen zum UTAUT-Modell – wird dadurch nicht eingeschränkt. Die Einflussnahme erfolgt über Methoden, sodass die systematische Berücksichtigung von Einflussfaktoren zur Verbesserung von Akzeptanz vordergründig über die Integration von Methoden stattfinden kann. Die damit verbundenen Aktivitäten können wiederum den einzelnen Phasen des erweiterten Multipfad-Vorgehensmodells zugeordnet werden.

Ähnlich verhält es sich mit einem Akzeptanzmodell völlig anderen Charakters. Kollmann versuchte in seinem Akzeptanzmodell (Abb. 25, S. 98) dem dynamischen Charakter von Akzeptanz gerecht zu werden. Er unterscheidet eine Einstellungsphase, eine Handlungsphase (Übernahme) und eine Nutzungsphase für Produkte. Zusätzlich gliedert er die möglichen Einflussfaktoren nach Mensch (akzeptiererbezogen), Organisation (organisationsbezogen) und Technik (produktbezogen). Kollmann versteht *Akzeptanz* als dynamisches Konstrukt und skizziert rudimentär den Akzeptanzprozess (Abb. 25, S. 98). Eine Annäherung der Elemente des Akzeptanzprozesses nach Kollmann (2006, S. 266 u. 1998, S. 135) an das Multipfad-Vorgehensmodell zeigt Abb. 36. Die Prozesselemente wurden von Kollmann im Kontext von „Kaufvorgängen“ gestaltet und beziehen sich auf den Menschen in seiner Rolle als Käufer und Anwender. Bei Einführung eines neuen Anwendungssystems befindet sich der Anwender in einer ähnlichen Rolle. Auswahl und Kaufentscheidung fallen jedoch in der Regel weg.

Die Einbeziehung von Phasen bzw. Vorgangstypen des Akzeptanzmodells von Kollmann in das Multipfad-Vorgehensmodell bietet sich ebenso wenig an, wie das für das UTAUT-Modell der Fall ist. Kollmann ordnet seinem Akzeptanzprozess sieben Gruppen von Aufgaben zur Akzeptanzförderung zu. Diese Aufgaben gilt es, wenn möglich auch unter Berücksichtigung von Aufgaben zum UTAUT-Modell, den einzelnen Phasen des erweiterten Multipfad-Vorgehensmodells zuzuordnen.

Kollmann beginnt mit der Bewusstseinsbildung und fährt damit fort, Interesse zu schaffen. Im Hinblick auf die Anwendungsentwicklung bedeutet das, Bewusstsein und Interesse für den Veränderungsprozess an sich und für ein neues Anwendungssystem

zu schaffen. Beides sind Aufgaben, die im Multipfad-Vorgehensmodell vor seiner Erweiterung nicht explizit verankert waren. Mit der Integration des Gestaltungsrahmens für den Usability Engineering Prozess wurden diese Aufgaben in Ausprägung einer Sensibilisierung für Usability der Planungsphase jedoch verankert und dem Projektmanagement zugeordnet (vgl. Abschnitt 3.5.3.2, S. 159ff). Als Schnittstelle hierzu ist im erweiterten Multipfad-Vorgehensmodell der Pseudo-Vorgangstyp Projektplanung zu verstehen. Weder die Bewusstseinsbildung noch die Schaffung bzw. die Erhaltung des Interesses kann in einem Vorgangstyp adäquat abgebildet werden. Eine systematische Berücksichtigung kann über die Integration von Methoden stattfinden.

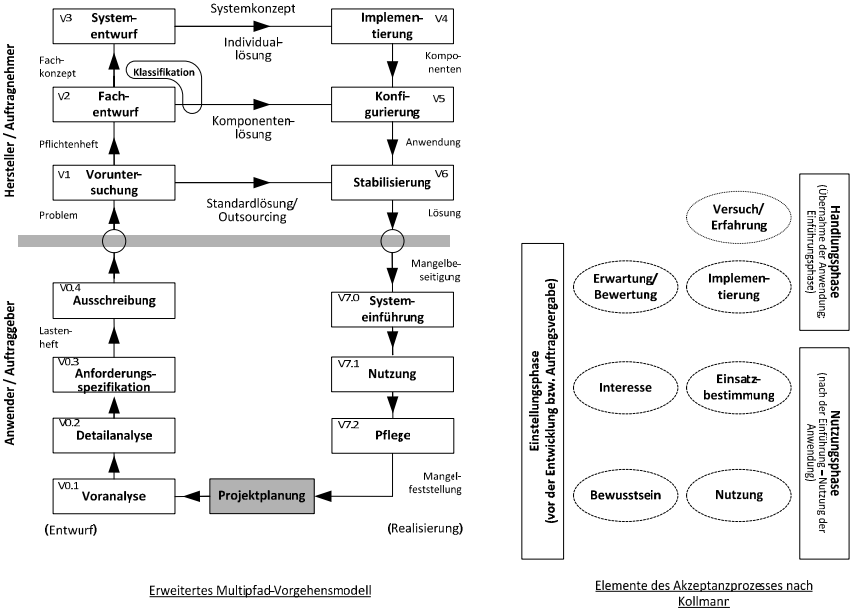

Abb. 36: Annäherung des Akzeptanzprozesses nach Kollmann an das erweiterte Multipfad-Vorgehensmodell

In einem weiteren Aufgabenpaket zur Schaffung von Einstellungsakzeptanz geht es um die Erwartungshaltung der Anwender und die Bewertung dieser. Erwartungen im Rahmen von Technologieinnovationen unterteilt das UTAUT-Modell in Leistungserwartungen an das System und Erwartungen zum Arbeitsaufwand (vgl. Tab. 7, S. 195). In diesen Gruppen von Einflussgrößen sind Faktoren zusammengefasst, die sich direkt auf die Einstellung der Anwender auswirken. Zusätzlich gibt es noch indirekt wirksame Faktoren auf die Einstellung zur Technologienutzung (vgl. Tab. 11, S. 199). Die Aufgaben dazu stehen wiederum in engem Zusammenhang mit denen des Usability

Engineering, sind aber teilweise auch bereits explizit im Multipfad-Vorgehensmodell verankert, wie z.B. die Sicherstellung der Zweckmäßigkeit, die Komplexitätsreduktion oder die Maßgabe der Sicherstellung einer einfachen Nutzung (ease of use). Als zusätzliche Aufgaben sind u.a. die Förderung der Motivation (intrinsische und extrinsische Motivation), Herstellung eines Job-Fit für den Benutzer sowie die Beachtung von Einflussfaktoren, insbesondere die emotionale Dimension des Usability Engineering betreffend (vgl. Abschnitt 5.3, S. 194ff), in den Vorgangstypen (V0 bis V7) zu verankern. Der Schwerpunkt der Aufmerksamkeit liegt bei diesen Aufgaben in den frühen Entwicklungsphasen und bei der Einführung sowie den dort relevanten Methoden. Aus Sicht der Anwendungsentwicklung nach der sprachbasierten Informatik (Abb. 26, S. 101) sind dies die Rekonstruktion und die Stützung der Nutzer.

Kollmann fährt im Zuge der Produktübernahme fort mit Testmöglichkeiten und dem Einfluss von Erfahrungen auf das Anwenderverhalten. Testmöglichkeiten sind sowohl im Usability Engineering als auch in der Anwendungsentwicklung verankert. Die Umsetzung erfolgt über Methoden mit ausreichender Benutzerbeteiligung. Als Implementierung bezeichnet Kollmann die Übernahme eines Produkts. Die Verwendung des Begriffs steht in keinem Zusammenhang mit der Implementierung im Sinne der Anwendungsentwicklung und des Usability Engineering. Der Begriff kann im übertragenen Sinn als Einführung eines Systems verstanden werden. In Verbindung mit Einsatzbestimmungen zielt Kollmann hier auf eine günstige Beeinflussung der Nutzung bereits vor der tatsächlichen Nutzung ab. Sowohl im Multipfad-Vorgehensmodell als auch im Gestaltungsrahmen der DATech sind Aufgaben hinsichtlich der Nutzungsvorbereitung verankert. Die Darstellung der Unterstützung der Beeinflussung des Nutzungsverhaltens ist jedoch wenig ausgeprägt. Dies kann im Rahmen einer Methodenintegration für diese Phasen konkretisiert werden. Kollmann versucht mit seiner Gruppierung von Einflussgrößen diese, bezogen auf den Akzeptanzprozess, implizit zu bündeln. Er schließt sein Modell jedoch nicht im Sinne einer Lebenszyklusorientierung ab.

Eine Entwicklungsphase, wie sie in den Prozessmodellen der Anwendungsentwicklung und des Usability Engineering ausgeprägt ist, kommt bei beiden besprochenen Akzeptanzmodellen (vgl. Abschnitte 2.4.6.3 u. 2.4.6.4, S. 94ff) nicht vor. Die gezielte Einflussnahme auf *Akzeptanz* von Anwendungssystemen innerhalb der Entwicklungsarbeiten kann dadurch über diese Modelle nicht transparent gemacht werden. Hinzu kommt, dass die Einflussfaktoren (noch) nicht mit Methoden zu deren Beeinflussung in Verbindung gebracht werden. Beide Modelle sind Teil des Eco-Systems und fokussieren auf den Anwender. Hersteller und Auftraggeber sind nicht angesprochen. Demzufolge unterbleibt eine Rollenverteilung hinsichtlich der Möglichkeiten, wer wann wie in der Lage ist, welche Aspekte von Akzeptanz zu beeinflussen.

Da das Modell von Kollmann auf den Kaufprozess abzielt und eine Einbeziehung der (Produkt-)Entwicklung fehlt, ist es nur bedingt auf die Anwendungssystementwicklung übertragbar. Die Ergänzung bzw. Erweiterung von Vorgangstypen im Multipfad-Vorgehensmodell anhand des Modells von Kollmann erscheint nicht zweckmäßig. Trotzdem bringt das Modell von Kollmann es wichtige Impulse für das *Acceptance Engineering*. Wertvoll ist vor allem die Erkenntnis, *Akzeptanz* als dynamisches Konstrukt, welches individuell bezogen auf den Anwender Zustandsänderungen im Laufe des Entwicklungsprozesses erfahren kann, zu verstehen.

Die Einflussfaktoren des UTAUT-Modells stehen nicht im Widerspruch zum Modell von Kollmann. Die Zuordnung von Aufgaben des Acceptance Engineering wurde anhand der Gruppierung von Kollmann vorgenommen (Abb. 36, S. 169). Der Bezug zu möglichen Einflussfaktoren im Kontext von Aufgaben erfolgte über das UTAUT-Modell, da diese Einflussfaktoren im Kontext von Technologie-Innovationen untersucht wurden und zusätzliche Einflüsse von Erfahrung, Geschlecht und Alter berücksichtigt werden. Welche Aktivitäten zu welchen Zustandsänderungen führen können, ist bis dato nur ansatzweise untersucht (z.B. von Venkatesh et al., 2003). Inwieweit eine derzeit noch nicht definierte Akzeptanzmessung die Verfolgung dieser Zustandsänderungen in Zukunft ermöglichen wird, bleibt offen. Beispielsweise könnte mit einer „kumulierten Akzeptanzkurve“ (erreichter Akzeptanzlevel der betroffenen Benutzer in einer Organisation im Sinne einer Diffusionskurve) der Projektfortschritt aus Benutzersicht transparent gemacht werden. Auch die Gestaltung eines Akzeptanzprofils und die Darstellung der Entwicklung dieses Profils im Zeitverlauf wäre ein Ansatz für ein Acceptance-Controlling. Die konzeptionelle Gestaltung und auch die Durchführung einschlägiger empirischer Untersuchungen gehen über das vorliegende Integrationsvorhaben weit hinaus und sind weiterführenden Forschungs- und Entwicklungsprojekten vorbehalten.

3.5.4 Integration über Methoden

3.5.4.1 Einleitung und Übersicht

Im Übergang zur Integration auf externer Ebene (Tab. 1, S. 7) dienen Methoden in Abstimmung mit den behandelten Prozessmodellen als Integrationsmedien zur überwiegend praktischen Integration. Gleiche oder scheinbar äquivalente Methoden kommen aus unterschiedlichen Disziplinen und stellen sich in unterschiedlichem Anwendungskontext anders dar. Den Anwendern von Methoden fällt es schwer im Vorfeld des Methodeneinsatzes abzuschätzen, welche Methode oder welche Kombination von Methoden in welcher Variante und Intensität für die vorliegende Problemstellung zielführend ist. Hinzu kommt, dass sich die Ansprüche an Methoden und Methodenbeschreibung ändern. Dies ist einerseits abhängig vom Anwendungszweck

und Anwendungskontext, und andererseits davon bestimmt, wer eine Methode einsetzt, ob dies eine Hersteller- oder eine Anwenderorganisation ist, und z.B. auch davon, welchen Auftrag und welche Qualifikation die anwendende Person hat.

Voraussetzung für eine gelungene Zusammenführung von Methoden aus den unterschiedlichen beteiligten Disziplinen ist die Schaffung einer gemeinsamen Basis für deren Beschreibung. Die Grundlagen hierfür wurden bereits in den Abschnitten 2.3.4 (S. 52ff) bzw. 2.4.4 (S. 70ff) geschaffen. Zur Umsetzung wird ein Ordnungsrahmen vorgeschlagen (Abb. 37, S. 177). Er unterstützt eine anwendungsorientierte und systematische Aspektierung von Methoden. Der Rahmen ist modular und damit erweiterbar, d.h. es können jederzeit weitere Beschreibungselemente integriert werden. Allerdings sollte darauf geachtet werden, dass neu einzuführende Aspekte orthogonal zu allen vorhandenen Aspekten stehen. Ist das nicht der Fall, müssen bestehende Aspekte um die entsprechende Funktionalität bzw. Ausdruckskraft erweitert werden (Jablonski et al., 1999). Zudem sind die Methoden an den zugrunde gelegten Entwicklungsparadigmen (vgl. Abschnitt 2.1, S. 15ff) auszurichten und die Arbeitsprinzipien (vgl. Abschnitt 1.2, S. 4ff) sind zu beachten.

Das theoretische Modell des Ordnungsrahmens fordert weitgehende Unabhängigkeit der *Aspekte* untereinander (= *Orthogonalität*). Bei der anwendungsbezogenen Beschreibung einzelner Aspekte können jedoch Redundanzen entstehen. Dies widerspricht der Forderung nach *Orthogonalität* nur bedingt und wird damit begründet, dass eine Methodenbeschreibung ihrerseits den Anforderungen der Gebrauchstauglichkeit genügen soll. Redundanzen sind zulässig, um dem Methodenanwender die Orientierung zu erleichtern. Sie werden idealerweise durch Querverweise transparent gemacht. Die Unabhängigkeit der Aspekte untereinander ist zudem Voraussetzung für die geordnete Erweiterung des Modells.

Zur Gestaltung gebrauchstauglicher Methodenbeschreibungen bietet sich die Vernetzung von Methodenkomponenten mit Hilfe eines Repository an. Über dieses Werkzeug kann auch die gemeinsame Fachsprache eingebunden werden. Damit greift die Integration über Methoden auf dieselben Grundlagen zurück wie die Anwendungsentwicklung und bedient sich der gleichen Theorien (z.B. des Wissensmanagements, vgl. Abschnitt 3.4.2, S. 141ff).

3.5.4.2 Methoden im erweiterten Multipfad-Vorgehensmodell

Methoden der Anwendungsentwicklung und des Usability Engineering können den Phasen bzw. Vorgangstypen des Multipfad-Vorgehensmodells zugeordnet werden. Spezifische Methoden zur Umsetzung des *Acceptance Engineering* im Sinne einer systematischen Beeinflussung von Akzeptanzfaktoren konnten nicht identifiziert werden. Durch das Zusammenführen der Prozessmodelle kam es zu einer Erweiterung

des Multipfad-Vorgehensmodells (vgl. Abschnitt 3.5.3, S. 158ff). Eine wesentliche Änderung besteht in der Verlagerung von Analyseaufgaben auf die Auftraggeberseite. Was das für die Methodenzuordnung in den frühen Entwicklungsphasen bedeutet, verdeutlicht Tab. 6 (S. 174).

Die angeführten Methoden des Usability Engineering beziehen sich überwiegend auf den eingeführten Gestaltungsrahmen für den Usability Engineering Prozess der DATech (2008). Methoden der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik wurden für die frühen Phasen eingehend beschrieben und punktuell auf relevante weitere Methoden verwiesen (vgl. Abschnitt 3.2, S. 104ff). Eine systematische, dem Ordnungsrahmen entsprechende, und zugleich anwendungsorientierte Beschreibung dieser Methoden existiert bis dato noch nicht. Die Konkretisierung (= Ausprägung) einer Methode macht oft erst in einem bestimmten Anwendungskontext Sinn. Es handelt sich dann um eine reale, unternehmensspezifische Ausgestaltung einer Methode.

Sowohl der DATech Gestaltungsrahmen als auch damit verbundene Normen (z.B. DIN EN ISO 9241-110, 2006; DIN EN ISO 13407, 1999) beinhalten zahlreiche Prinzipien und Grundsätze zur nutzungsorientierten Gestaltung von Systemen und zu einem benutzerzentrierten Vorgehen. Relevante Prinzipien, Regeln und Grundsätze können in einem Regelwerk für die Entwicklungsarbeit (z.B. einer SOA-Governance) im Unternehmen verankert sein. Für die Anwender von Methoden sind die Gesamtzusammenhänge solcher Regelwerke oft nicht einfach herzustellen. Relevante Aussagen zu einem Problem sind oft nicht rasch genug auffindbar. International anerkannte Normen sind in der Regel nicht weit genug spezifiziert, um daraus einen konkreten Handlungsrahmen für ein Entwicklungsprojekt ableiten zu können. Für die Integration über Methoden bedeutet das, dass im Detail ausreichend Gestaltungsraum und -freiheit gegeben ist und Detailbeschreibungen einer Methode organisationsspezifisch zu schaffen sind.

Systematische Beschreibungen von Methoden liegen im Gestaltungsrahmen für das Usability Engineering nicht vor, es handelt sich vielmehr um Aufgabenbeschreibungen die durch Erfahrungswerte angereichert sind. Die dazu festgehaltenen best practices bieten nützliche Hinweise zur Aufgabenerfüllung, ermöglichen aber keinen direkten Methodenvergleich, sodass z.B. daraus eine systematische Gegenüberstellung von Vorgehensweisen und erforderlichen Ressourcen von Methoden zur Unterstützung der Analysearbeit abgeleitet werden könnten. Auch die Auflistung von erwarteten Ergebnissen und zugehörigen Maßnahmen kann diese Lücke nicht kompensieren. Die Beschreibungen sind als Handlungsanleitung für einen Entwickler ungeeignet.

Vorgangstyp bzw. Phase	**Methoden der Anwendungsentwicklung**	**Methoden des Usability Engineering**
Projektplanung	Methoden des Projektmanagements (als Querschnittfunktion)	Methoden der Kommunikation zur Sensibilisierung für Usability
Voranalyse	• Interview (z.B. Wedekind, 1976; Schienmann, 2002) • Dokumentenanalyse (s.o.) • Aussagensammlung (z.B. Apel, 1976; Ortner, 2005)	• Kosten-/Nutzenanalyse (z.B. Sen, 2000; Hirschmeier, 2005) • Lead-User-Interview (z.B. DATech, 2008) • Dokumentenanalyse (z.B. Lehner et al., 1991; DATech, 2008)
Detailanalyse	• Interview (s.o.) • Dokumentenanalyse (s.o.) • Aussagensammlung (s.o.) • Teilnehmende Beobachtung (z.B. Lüders, 2006) • Sprachkritische Rekonstruktion (z.B. Ortner, 1993 u. 2005; Schienmann, 1997; Lehmann, 1999) • Begriffsklärung (z.B. Lorenzen, 1985; Ortner, 1997 u. 2005; Schienmann, 1997; Lehmann 1999)	• Dokumentenanalyse (s.o.) • Teilnehmende Beobachtung (z.B. DATech, 2008) • Ethnografische Beobachtung (z.B. Sommerville & Kotonya, 1998) • Thinking aloud (z.B. Holzinger, 2004) • Personas (z.B. usability.gov, 2008) • Claims Analysis (z.B. Seffah et al., 2005; DATech, 2008; Usability first, 2008) • Aufgabenanalyse (z.B. DATech, 2008) • Szenariotechniken (z.B. Rosson & Carroll, 2002) • Use Cases (z.B. Fowler & Scott, 2000; Interface consult, 2008) • Walkthrough-Verfahren (z.B. Nielsen, 1993)
Anforderungs-spezifikation	• Begriffsklärung (s.o.) • Aussagensammlung (s.o.) • Workshops Erstellung eines Lastenhefts	• Fokusgruppen (z.B. usability.gov, 2008) • Claims Analysis (s.o.) • Workshops
Ausschreibung	Keine spezifischen Methoden: Kennenlernen des künftigen Partners Vertragsgestaltung	Keine spezifischen Methoden: Vertragsgestaltung
Voruntersuchung	• Bedingungsmatrix (z.B. Franke, 1975; Wedekind & Ortner, 1980; Ortner, 2005) Erstellung eines Pflichtenhefts	Keine spezifischen Methoden: Planung von Usability Aktivitäten projektspezifische Festlegungen Rollenverteilung
Fachentwurf	• Dokumentenanalyse (s.o.) • Sprachkritische Rekonstruktion (s.o.) • Aussagensammlung (s.o.) • Begriffsklärung (s.o.) • Repository (z.B. Ortner, 2005) • Normierung und Klassifizierung von Aussagen (z.B. Ortner, 1997; Schienmann, 2002)	• Teilnehmende Beobachtung (s.o.) • Ethnografische Beobachtung (s.o.) • Aufgabenanalyse (s.o.) • Style Guidelines (z.B. Nielsen, 1993; Mayhew, 1999a; Sarodnick & Brau, 2006) • Usability Prototyping (Pomberger et al., 1991; Sarodnick & Brau, 2006) • Heuristiken (z.B. Nielsen, 1993) • Expertenleitfäden (z.B. Sarodnick & Brau, 2006; Mayhew, 1999a) • GOMS-Modell Benutzerbefragung (z.B. DATech, 2008)

Tab. 6: Methoden im erweiterten Multipfad-Vorgehensmodell

Im Usability Engineering sind Test- und Messmethoden sehr ausgeprägt. Diese überwiegende Test- und Messorientierung wird dem technischen Bereich des Usability Engineering zugeordnet und findet in diesem Rahmen keine Berücksichtigung. Dennoch können Test- und Messmethoden auch in der vorgeschlagenen Weise zusammengeführt werden.

In einer systematischen, integrierten Methodenbeschreibung für den Zweck der Unterstützung einer benutzerorientierten Entwicklungsarbeit überwiegen Elemente der Anwendungsentwicklung und des Usability Engineering. Diese können jederzeit um Aspekte aus der Psychologie, insbesondere der Kognitionspsychologie und der Sozialpsychologie, der Gehirnforschung und anderen Disziplinen, soweit diese für relevant gehalten werden, ergänzt werden. Die Einflussfaktoren des UTAUT-Modells sollten aufgegriffen werden und bei der systematischen Adressierung von Systemeigenschaften über die Usability Dimensionen dazu dienen, gewollte Beeinflussungen klarer zu definieren.

Anhand von systematischen Beschreibungen lassen sich Parallelitäten bzw. Kongruenzen unterschiedlicher Methoden leichter bzw. überhaupt erst identifizieren. Methoden werden vergleichbar, Parallelitäten können dazu genutzt werden, Methoden zusammenzuführen. Wesentliche Einflussfaktoren (z.B. aus dem Acceptance Engineering oder aus dem Wissensmanagement) können systematisch adressiert werden. Auch Systemeigenschaften in den unterschiedlichen Usability-Dimensionen lassen sich damit gezielt adressieren. Die Methoden der Anwendungssystementwicklung stellen das führende Medium für eine Methodenintegration dar. Die Schaffung eines Ordnungsrahmens ist Grundlage für die Erarbeitung einer einheitlichen, systematischen und integrierten Methodenbeschreibung über mindestens zwei Disziplinen. Dieser Rahmen kann als zweckneutrale Grundlage für weitere Integrationsvorhaben herangezogen werden.

3.5.4.3 Schaffung eines Ordnungsrahmens zur integrierten Methodenbeschreibung

Eine Methode wurde klar definiert als Sprache in Verbindung mit einer Vorgehensweise (vgl. Abschnitt 2.3.4.1, S. 52ff). Zudem wurde in Abschnitt 2.2.4 (S. 26ff) ein allgemeines Datenmodell für Methoden vorgestellt (Abb. 5, S. 27). Liegt eine systematische Vernetzung von Aktivitäts- und Resultatstypen von Methoden vor, sprechen wir von einem Vorgehensmodell. All diese Definitionen reichen aus, um in den folgenden Ausführungen auf eine klare theoretische Grundlage zurückgreifen zu können. Anwendungsbezogen können diese Ausführungen Grundlage sein für die Modellierung von Aktivitäten (z.B. Methoden) zum Zweck der Automatisierung. Für die Auswahl von Methoden bzw. den Methodeneinsatz im Sinne einer zweckorientierten Unterstützung eines Entwicklerteams reichen diese Grundlagen jedoch nicht aus.

Allein die Vielzahl der in der Literatur beschriebenen Methoden in unterschiedlicher Qualität und Granularität, und dies noch auf die Bedürfnisse der unterschiedlichen Disziplinen ausgerichtet, würde Schwierigkeiten in der Auffindung und in der Auswahl der Methoden mit sich bringen. Die gleiche Methode nimmt für unterschiedliche Anwendungszwecke mit unterschiedlichem Anwendungskontext jeweils eine andere Gestalt an. Hinzu kommt noch, dass die Ansprüche an eine Methode und damit auch an eine Methodenbeschreibung variieren. Dies ist abhängig davon, wer die Methode einsetzen wird (z.B. Anwenderorganisation, Analytiker, Enterprise Engineer, Entwickler, Herstellerorganisation). Die Ansprüche an Methoden unterscheiden sich also hinsichtlich des Zeitaufwands, der beteiligten Rollen und Personen, der einzusetzenden Mittel, der benötigten Infrastruktur (Raum, Medien), der Projektgröße und auch hinsichtlich der Ergebniserwartung, um nur einige zu nennen. Um die gegebene Komplexität beherrschen zu können greifen wir auf das Prinzip der Aspektorientierung zurück.

Die Aspektorientierung ist grundsätzlich nicht neu. In der Konstruktionslehre des Maschinenbaus (z.B. Roth, 2000, mit Bezug auf Hubka, 1973) werden die Stadien der Konstruktion über den Zeitablauf gegliedert (Phasen). Innerhalb dieser werden sukzessive die relevanten *Aspekte* des zu konstruierenden Systems erschlossen (z.B. Baustruktur, Gestalt, Einbezug von Eigenschaften, Funktionen usw.). Auch im Bereich von technisch orientierten Unternehmensarchitekturen findet eine aspektorientierte Betrachtung statt. So kann die zwei-dimensionale Zuordnung von sechs verschiedenen Perspektiven (Was, Wie, Wer, Wo, Wann, Warum) zu den verschiedenen Akteuren des betrachteten Objekts (Sessions, 2007), im Zachman Framework (Zachman, 1987 u. 2008) auch als Aspektierung aufgefasst, erfolgen.

In der Entwicklung von Workflowsystemen betrachten Jablonski et al. (1999) die Modellierungsbereiche als Aspekte (z.B. Funktionsaspekt, Verhaltensaspekt, Informationsaspekt, Operationsaspekt, Organisationsaspekt, Technologieaspekt) zur Gliederung der Modellierungsarbeit. Aspekte sind also Modellierungsbereiche, aus denen schrittweise ein Gesamtbild (Schema) z.B. einer Workflow-Management-Anwendung entsteht. Es wird gefordert, dass *Aspekte* (weitgehend) orthogonal im Hinblick auf den sie umgebenden Kontext (andere Aspekte) zu bestimmen sind (Jablonski et al., 1999, S. 485). Ortner (2008a) greift ebenfalls auf eine Aspektorientierung zurück, wenn er z.B. die Arbeitsorganisation im Rahmen der informatischen Organisationslehre beschreibt. Er differenziert dabei zwischen statischer und dynamischer Betrachtung. Mit der Architektur (Bestand, statisch) und Aspekten wie z.B. Arbeitsmittel, Arbeitseingabe, Arbeitsausgabe, Arbeitsleister, Arbeitsplatz und Arbeitsgegenstand wird der Arbeitsaufbau beschrieben. Mit Prozessen (Bewegung, dynamisch) und Aspekten wie Arbeitsvorgang, Arbeitsverfahren, Arbeitsfolge wird der Arbeitsablauf beschrieben.

Im übertragenen Sinn werden Aspekte im Ordnungsrahmen für die Modellierung und Beschreibung von Methoden verwendet. Durch Methoden wird Arbeit organisiert und unterstützt. Die Auswahl von Aspekten zur Methodenbeschreibung orientiert sich daher an den Aspekten zur Arbeitsorganisation nach Ortner (2008a) und an den Aspekten für Workflow-Management-Systeme, wie sie von Jablonski et al. (1999) vorgeschlagen werden. Die statischen und dynamischen Beschreibungskategorien werden erweitert um anwendungsorientierte Angaben zur Methode, wie z.B. Zweckeignung der Methode, Erfahrungswerte (Vorteile, Risiken), Qualitätsaspekt, normativer Aspekt. Während die statischen und dynamischen Aspekte weitgehend orthogonal zueinander stehen, trifft dies für Erfahrungswerte und die Beschreibungselemente in der Methodenübersicht nicht zu. Die letztgenannten Kategorien dienen überwiegend der Orientierung für Methodenanwender, während die Beschreibung der strukturellen und prozessualen Aspekte so weit detailliert werden kann, dass eine Methode nach dieser Beschreibung auch automatisiert werden könnte. Die entstandene Grobstruktur (Abb. 37, S. 177) kann jeder (anwendungsbezogenen) Methodenbeschreibung zugrunde gelegt werden.

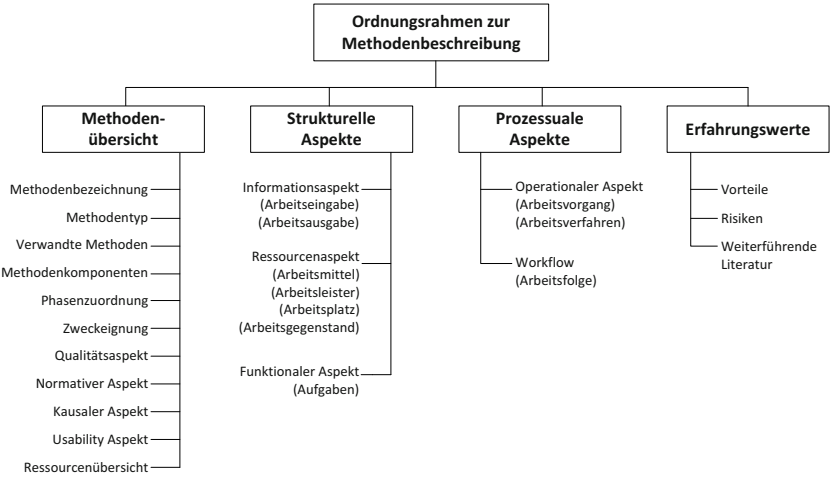

Abb. 37: Ordnungsrahmen zur Methodenbeschreibung

Eine Erweiterung des Ordnungsrahmens zur Methodenbeschreibung ist jederzeit möglich, die Verfeinerung und Konkretisierung der strukturellen und prozessualen Aspekte sind anwendungsspezifisch unbedingt erforderlich. Sowohl die anwendungsbegleitende Ergänzung der Erfahrungswerte als auch die Pflege und Aktualisierung der Inhalte über den gesamten Wissenslebenszyklus (Ortner, 2005, S. 166f) stellt eine Komponente des Wissensmanagements in der Entwicklung von Anwendungssyste-

men dar (z.B. Davenport & Prusak, 1999; Fink, 2000; Heinemann, 2006). Im Anhang (Abschnitt 5.5, S. 206ff) sind die einzelnen Aspekte zum Ordnungsrahmen genauer spezifiziert. Die systematische Vernetzung von Methodenbeschreibungen und Gebrauchssprache ist z.B. über ein Repository möglich.

Die Gestaltung von Arbeit und Arbeitsabläufen ist Sache der Anwenderorganisation. Die Organisationen sind zunehmend gefordert, Methoden, insbesondere Erhebungsmethoden zur Rekonstruktion von Prozessen, ihren Anforderungen entsprechend auszugestalten und zu formulieren. Dazu sind die Methoden aus Tab. 6 (S. 174) auf Basis des eingeführten Ordnungsrahmens in einer anwendungsorientierten, organisationsbezogenen Beschreibung zusammenzuführen. Die Beschreibung beschränkt sich nicht auf die Erweiterung um Methodenkomponenten des Usability Engineering. Anwendungsspezifisch und erfahrungsbasiert sollte Wert auf die empirisch fundierte Beschreibung der Beeinflussung von Akzeptanz gelegt werden. Bei der Methodenzusammenführung ist außerdem darauf zu achten, dass Methodenkomponenten, die aus relevanten Disziplinen zur sprachkritischen Rekonstruktion hinzukommen, ebenfalls sprachsensibel ausgestaltet werden. Eine permanente Förderung der *Sprachkompetenz* und *Sprachperformanz* der Menschen im Anwendungsbereich rundet die Integration ab.

4 Fazit und Ausblick

Trotz der vielen Vorteile und der Wirtschaftlichkeit, die das Usability Engineering verspricht und die in zahlreichen Projekten nachgewiesen wurden (Nielsen, 1993, S. 3), wird dem Usability Engineering noch immer ein relativ geringer Stellenwert eingeräumt. Viele Auftraggeber scheuen die zusätzlichen Kosten zu Projektbeginn, die mit dem Usability Engineering verbunden sind. Je früher im Entwicklungszyklus Usability Engineering bzw. Acceptance Engineering stattfinden, desto geringer sind die damit verbundenen Kosten (Mai, 1994). Donahue et al. (1999) führen etwa aus, dass für jeden in Usability investierten Euro zwischen 10 und 100 Euro in der Entwicklung, bei den Tests und bei nachträglichen Änderungen eingespart werden können. Bei der Entwicklungszeit können 30 % bis 60 % der Zeit für Entwicklungsaufgaben eingespart werden, wenn Usability Experten bereits von Projektbeginn an mit einbezogen werden. Ist ein System gebrauchstauglich gestaltet, hat das geringere Schulungskosten zur Folge. Sarodnick & Brau (2006, S. 86-87, mit Anlehnung an Kalbach, 2003) listen Argumente zur Rechtfertigung von zusätzlichen Kosten des Usability Engineering in der Anfangsphase des Entwicklungsprozesses auf hinsichtlich Erfolgssicherheit, Zeitersparnis, Kostenersparnis, Steigerung der Umsätze, Entwicklung neuer Ideen sowie Stressreduktion und Spaß. Diese Fakten lassen sich mit dem Konzept des dynamischen LifeCycle Costing monetär untermauern (Britzelmaier & Eller, 2004).

Für Organisationen die Anwendungssysteme einsetzen, und das betrifft nahezu alle Organisationen in Wirtschaft und Verwaltung, verspricht der vorliegende Ansatz großen Nutzen. Mit der Verlagerung von benutzernahen Teilen der Entwicklungsarbeit in die Anwenderorganisation zeigen sich nennenswerte Nutzenpotenziale. Diese liegen bei der internen Prozessoptimierung oder Prozessharmonisierung mit verbundener Kostensenkung ebenso wie bei der Sicherung von Know-how innerhalb des Unternehmens oder in der Veränderung der Verhandlungsposition zur Herstellerorganisation bzw. zu Anbietern bei Anschaffung von Anwendungssystemen.

Die Notwendigkeit der Integration des Usability Engineering in die Anwendungsentwicklung gilt als längst erkannt (vgl. Abschnitt 1.1, S. 1ff). Das Usability Engineering ist eine Disziplin ohne klare Konturen. Es kommen darin Ansätze der Wirtschaftsinformatik ebenso zum Tragen, wie beispielsweise solche des Qualitätsmanagement, des Requirements Engineering oder der Psychologie. Der vorliegende Ansatz unterscheidet sich grundlegend von gängigen Publikationen zum Usability Engineering, er greift eine bisher unbearbeitete Nische des Usability Engineering auf. Die Integration über die sprachbasierte Informatik eröffnet einen wissensbasierten Zugang zur Steuerung von Akzeptanz und Gebrauchstauglichkeit von Anwendungssystemen. Die Veran-

kerung im erweiterten Multipfad-Vorgehensmodell betont die zunehmende Verlagerung dieser Entwicklungsarbeiten in die Anwenderorganisation und kommt der Integration des Usability Engineering insofern entgegen, als die Entwicklungsarbeiten „näher" beim Anwender stattfinden und die benutzerorientierte Arbeit dadurch wesentlich erleichtert wird. Die Integration erfolgt auf drei Ebenen, auf interner, auf konzeptioneller und auf externer Ebene (vgl. Tab. 1, S. 7), letztere entspricht dem Anwendungsbereich. Die sprachbasierte Informatik ermöglicht dabei eine nahtlose Integration, wie sie Mayhew bereits 1999 forderte (Mayhew, 1999a; vgl. Abschnitt 1.1., S. 5f).

Die sprachbasierte Informatik ist fundiert im Erlanger Konstruktivismus (vgl. Abschnitt 1.4, S. 8ff). Das Verständnis von Sprache als System geht auf die logische Propädeutik (vgl. Kamlah & Lorenzen, 1975; Lorenzen, 2000) zurück. Vor allem Wedekind und Ortner adaptierten diese Erkenntnisse für die Anwendungsentwicklung. Die resultierende, sprachbasierte Entwicklung von Anwendungssystemen hebt die zu bewältigenden Sprachunterschiede innerhalb des Entwicklungsprozesses und zwischen den am Entwicklungsprozess Beteiligten hervor. Die systematische Bewältigung dieser Sprachunterschiede ist in Abschnitt 3 (S. 99ff) anhand von Sprachebenen verdeutlicht, beginnend im Anwendungsbereich, über eine geklärte Anwendungsfachsprache und Entwicklerfachsprachen bis hin zum Gebrauch, der wiederum im Anwendungsbereich liegt. Die Bewältigung der Übergänge zwischen den Sprachebenen passiert in der sprachbasierten Anwendungsentwicklung vor allem im Rahmen der Schemaentwicklung und bei der Stützung der Nutzer. Die Grundlagen und Modelle der sprachbasierten Informatik sind geeignet für eine systematische Integration des Usability Engineering über alle drei Ebenen (vgl. Tab. 1, S. 7).

Auf der internen Ebene, mit einer Orientierung in den Formal- und Sprachwissenschaften und der logischen Propädeutik als Grundlage, ist die grundlegende Integration verankert. Sie ist eine systematische Zusammenführung von „etwas" über Sprache. Diese sprachbasierte Zusammenführung erfolgt über die systematische Handhabung der *Bedeutung* von Begriffen und wird demzufolge auch als semantische Integration bezeichnet.

Bei der Betrachtung der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik werden Grundlagen, Potenzial und Lösungsschritte zur Umsetzung dieser semantischen Integration deutlich (vgl. Abschnitt 3, S. 99ff). Dies gilt in Bezug auf die nutzerorientierte Entwicklerarbeit insbesondere für die Rekonstruktion und die Stützung der Nutzer in der Rückführung der Anwendungssoftware in den Gebrauch (vgl. Abb. 26, S. 101). Dieser Zusammenhang wurde bis dato nicht erkannt oder völlig vernachlässigt. Erst die Betrachtung der Anwendungsentwicklung aus Sicht der sprachbasierten Informatik macht die Bedeutung dieses Zusammenhangs klar. Die semanti-

sche Integration ist eine notwendige Voraussetzung dafür, dass eine strukturelle und praktische Integration gelingen kann. Sie ist Grundlage für alle weiteren Aspekte der Integration.

Die Integration des Usability Engineering - und in Erweiterung dazu des Acceptance Engineering - auf konzeptioneller Ebene kann technischer oder struktureller Natur sein. Die Betrachtung des technischen Aspekts wurde für den vorliegenden Rahmen von vornherein ausgeklammert. Für eine strukturelle Integration auf Basis von Modellen bestehen bereits Ansätze (vgl. Abschnitt 3.5.2, S. 156ff). Diese Ansätze sind jedoch nicht so grundlegend verankert, wie der vorliegende. Die strukturelle Integration wurde hier vordergründig mit dem Mapping von Prozessmodellen der sprachbasierten Entwicklung von Anwendungssystemen, des Usability Engineering und Ansätzen des Acceptance Engineering bewerkstelligt. Es ist die Zusammenführung von Theorien anhand von Modellen. Theorien stützen sich auf Grundlagen. Die Sprachbasierung auf interner Ebene stützt die Zusammenführung auf konzeptioneller Ebene.

Ähnlich verhält es sich mit der modellbasierten Zusammenführung von Methoden relevanter Disziplinen (vgl. Abschnitt 3.5.4, S. 171ff). Zur integrierten Beschreibung von Methoden wird ein aspektorientierter und interdisziplinärer Ordnungsrahmen vorgeschlagen. Der Rahmen zur Methodenbeschreibung ist konzeptioneller Natur. Eine integrierte Beschreibung im vorgeschlagenen Rahmen ist Grundlage für die praktische Integration. Über den Methodeneinsatz ist der unmittelbare Anwendungsbezug herstellbar. Ist im Methodeneinsatz in der Praxis die Präsenz unterschiedlicher Disziplinen nicht oder kaum mehr spürbar, erscheinen die Aktivitäten aus den beteiligten Engineering Bereichen für alle Projektbeteiligten als Einheit. Die Integration aus praktischer Sicht ist dann im Sinne von Mayhew (1999a) als nahtlos zu bezeichnen und damit gelungen.

Methoden sind Konstrukte, die aus Vorgehensweisen in Verbindung mit Sprache bestehen (vgl. Abschnitt 2.3.4.1, S. 52ff). Über die eingesetzte Sprache können Methoden konstruktivistisch fundiert werden, z.B. auf Basis der logischen Propädeutik. Dieser Zusammenhang verdeutlicht die Rückkoppelung der Entwicklungsarbeit über Methoden von der externen Ebene, wo die praktische Integration stattfindet, auf die interne Ebene zur sprachbasierten Integration (vgl. Tab. 1, S. 7).

Die erste Kernfrage zur vorliegenden Problemstellung lautete: „Wie kann das Usability-Engineering innerhalb der Anwendungssystementwicklung systematisch unterstützt werden?“ Dieser Frage wurde mit der Erschließung der praktischen, strukturellen und semantischen Integration gemäß Tab. 1 (S. 7) auf den Grund gegangen. Neu in der Darstellung des Integrationsgeschehens ist die Rolle der semantischen Integration in der Rekonstruktion und in der Stützung der Nutzer. Durch ihre Grundlegung in

der logischen Propädeutik ist sie gleichermaßen systematisch wie praktisch. Die semantische Integration ist insofern als sehr anspruchsvoll zu bewerten, als sie von einem Entwickler bzw. einem Enterprise Engineer zusätzlich zur Fach- und Sozialkompetenz auch Sprachkompetenz und Sprachperformanz in einem hohen Ausmaß verlangt. Die konsequente Umsetzung einer Begriffsrekonstruktion und -klärung ist für die semantische Integration elementar. Eine Werkzeugunterstützung empfiehlt sich bereits für relativ kleine Projekte. Dadurch kann die nachhaltige Nutzung der Entwicklungsergebnisse begünstigt werden.

Die zweite Kernfrage zielte auf die strukturelle Integration ab: „Wie kann ein Vorgehensmodell (Prozessmodell) derart (flexibel) gestaltet werden, dass diese Integration hinreichend institutionalisiert ist und dennoch situativ adaptiert werden kann?“ Einen Lösungsansatz dazu bietet die gezeigte strukturelle Integration des Usability Engineering unter Berücksichtigung von Akzeptanzfaktoren in das Multipfad-Vorgehensmodell. Sie ergänzt die semantische Integration sinnvoll. Das rekonstruierte, erweiterte Multipfad-Vorgehensmodell ermöglicht eine bessere Unterstützung einer nutzerorientierten Entwicklungsarbeit während des gesamten Produktlebenszyklus. Gleichzeitig bietet es einen hinreichenden Rahmen für die grundlegende semantische Integration durch die Institutionalisierung der Rekonstruktion, der Modellierung und der Stützung der Nutzer. Zudem wird das erweiterte Multipfad-Vorgehensmodell (Abb. 35, S. 166) der zunehmenden Verlagerung von Entwicklungsarbeiten in die Anwenderorganisation gerecht und trägt damit der Einbindung von Usability Engineering und Acceptance Engineering Rechnung.

Das Beschreiten von verschiedenen Entwicklungspfaden im Sinne der eingeführten Schachbrettmetapher (vgl. Abschnitt 2.3.3, S. 41ff) ermöglicht eine situativ flexible Adaptierung des Vorgehens während des Entwicklungsprozesses. Übersicht und Transparenz werden durch die gegebene Strukturierung über das erweiterte Multipfad-Vorgehensmodell gewährleistet. Ein Vorgehensmodell ermöglicht nur mittelbar eine Steuerung von Produktakzeptanz und Gebrauchstauglichkeit. Über die zusätzlich vorgeschlagene systematische Zusammenführung von Methoden und deren kontextspezifische Ausrichtung ist eine solche Steuerung jedoch vorstellbar. Systematisches Akzeptanz-Controlling wäre eine denkbare Konsequenz daraus. Dies ist zu prüfen und durch weiterführende Studien zu untermauern.

Die nachhaltige Verbesserung der Entwicklungsarbeit ist jedoch nur dann möglich, wenn diese über adäquate Strukturen in das gesamte Unternehmensgeschehen eingegliedert wird. Als adäquate Struktur kann eine Unternehmensarchitektur in Verbindung mit relevanten Teilarchitekturen (z.B. Technologiearchitektur, Aufgaben- und Prozessarchitektur)und Regelsystemen (z.B. SOA-Governance und Corporate Governance) gelten.

Der vorgestellte Entwurf für die systematische Integration des Usability Engineering in die Entwicklung von Anwendungssystemen baut auf die Grundlagen der sprachbasierten Informatik. Der durchgeführte konstruktive Aufbau der Integration von den Grundlagen (interne Ebene – Integration über Sprache) über Theorien (konzeptionelle Ebene – Integration über Modelle) bis hin zur Praxis (externe Ebene - Umsetzung der Integration über integrierte Methoden) bringt Klarheit für den Entwickler und die nutzerorientierte Vorgehensweise. Damit sind Fundament und Modelle für die Verbesserung des Entwicklungsprozesses geschaffen, dies gilt insbesondere für die Steuerung von Produktakzeptanz und Gebrauchstauglichkeit. Eine durchgängige empirische Überprüfung des vorliegenden Entwurfs und weiterer Annahmen, die im Rahmen der Ausführungen zu treffen waren, ist noch offen und bleibt weiterführenden Studien in der Praxis der Entwicklungsarbeit vorbehalten. Idealerweise werden solche Studien über mehrere Jahre durch Experten aus den relevanten Disziplinen begleitet. Ein denkbarer Rahmen ist ein interdisziplinäres Forschungsprojekt. Im Rahmen eines solchen Projekts kann auch das Potenzial der systematischen Sprachbasierung, wie sie hier für die Anwendungsentwicklung aufgezeigt und konstruktivistisch fundiert wurde, als Grundlage für die Wirtschaftsinformatik generell, untersucht werden.

5 Anhang

5.1 Sprechakte und deren Taxonomie

> „Man kann für eine große Klasse von Fällen der Benützung des Wortes ‚Bedeutung' – wenn auch nicht für alle Fälle seiner Benützung – dieses Wort so erklären: Die Bedeutung eines Wortes ist sein Gebrauch in der Sprache."
>
> (Wittgenstein, 2001, S. 711)

Die Sprachakttheorie geht auf Austin (1979) zurück. In seiner Sprechhandlungstheorie (nicht Sprachhandlungstheorie) ging er davon aus, dass der Sprecher mit jeder Äußerung (zusätzlich zu dem, was sie jeweils bedeutet) eine spezifische Handlung ausführt („to do things"). Searle hat die Abhandlungen von Austin weiterentwickelt. Die Sprechakttheorie von Searle stellt den ersten systematischen Ansatz zur Beschreibung sprachlicher Handlungen dar. Wittgenstein ist nie so weit gegangen, die vielfältigen Ähnlichkeiten und Unterschiede zwischen den einzelnen Sprachspielen durch ein Klassifizierungsmodell oder durch ein einheitliches Beschreibungsmodell zu erfassen. Dies hätte wohl seinem Denkstil widersprochen (Hindelang, 2000, S. 84).

Die grundlegenden Handlungsarten, wie sie Searle postulierte, sind nachfolgend aufgelistet (Searle, 1976, Hindelang, 2000, S. 46-50; Ernst, 2002, S. 102):

Repräsentativa sind solche Sprechakte, durch die der Sprecher zu erkennen gibt, was er glaubt, dass in der Welt der Fall ist. Repräsentativa verpflichten den Sprecher zur Wahrheit der ausgedrückten *Proposition*. Repräsentativa sind z.B. behaupten, informieren, berichten, beschreiben, feststellen, klassifizieren, taxieren, vorhersagen.

Direktiva sind solche Sprechakte, durch die der Sprecher zu erkennen gibt, was er will, dass der andere tun soll. Direktiva stellen demnach Versuche des Sprechers dar, den Adressaten dazu zu bringen, etwas zu tun. Direktiva sind z.B. befehlen, bitten, anordnen, verbieten, raten, empfehlen, vorschlagen, fragen, erlauben.

Kommissiva sind solche Sprechakte, durch die der Sprecher zu erkennen gibt, was er selbst vor hat zu tun. Kommissiva verpflichten den Sprecher zu einer zukünftigen Handlung. Kommissiva sind z.B. vereinbaren, versprechen, anbieten, ausmachen, drohen, garantieren, schwören, sich verabreden.

Expressiva sind Sprechakte, durch die der Sprecher zu erkennen gibt, wie ihm zumute ist. Expressiva drücken demnach einen Zustand des Sprechers aus.

Expressiva sind z.B. danken, gratulieren, grüßen, verfluchen, jemandem etwas wünschen, entschuldigen, klagen.

Deklarativa sind solche Sprechakte, durch die der Sprecher zu erkennen gibt, was in einem bestimmten institutionellen Rahmen der Fall sein soll. Deklarativa bewirken eine sofortige Veränderung am Zustand der Dinge und neigen dazu, von komplexen außersprachlichen Ereignissen abzuhängen. Deklarativa sind z.B. ernennen, nominieren, verhaften, definieren, begnadigen, entlassen, etwas als etwas abkürzen, taufen, den Krieg erklären, kündigen.

Grundlegende Ausführungen zur Sprechakttheorie und eine detaillierte Interpretation dieser fünf Klassen von Sprechakten finden sich z.B. bei Hindelang (2000), Schrodt (1999) und Henne (1975). Bei Hindelang (2000, S. 86-97) ist auch eine Zusammenfassung der Beschreibung illokutionärer Akte durch Searle zu finden, und zwar tat Searle dies in Form von Regeln und Bedingungen.

Auch wenn die Erkenntnisinteressen der „Urväter“ des Sprachhandelns unterschiedlich sind und sicher gewisse Einschränkungen zu machen sind, so kann doch davon ausgegangen werden, dass die Ursprünge der Sprechakttheorie auf Wittgenstein (1889 – 1951) zurückgehen. Er führt bereits aus, dass „sprechen“ auch „handeln“ ist, was er mit dem Wort „Sprachspiel“ zum Ausdruck bringt. *Sprachspiele* sind durch Verben benannt. Diese waren Ausgangspunkt für die Sprechakttheorie von Austin (1911 – 1960) und später Searle (geb. 1932), welche auf den zwei folgenden Überlegungen basiert (Ernst, 2002, S. 91):

> Die Bedeutung sprachlicher Zeichen „haftet“ nicht absolut an den Ausdrücken, wie dies das *Saussure'sche* Zeichenmodell nahe legt, sondern folgt aus ihrem Gebrauch, ist also relational. Sprechen bedeutet Handeln.

5.2 Bausteinbeschreibung für den Usability Engineering Prozess

Die nachfolgende Beschreibung der Phasen ist zum überwiegenden Teil eine Darstellung der Essenzen dessen, was in DATech (2008) beschrieben ist. Auch wenn ein Bezug zur Anwendungssystementwicklung unterstellt wird, so sind in dieser Beschreibung keine entwicklungsspezifischen Aspekte zu finden. Die Beschreibung des Gestaltungsrahmens beschränkt sich auf Aktivitäten des Usability Engineering. Es werden zu jeder Phase Methoden genannt.

Planung

Die Phase der Planung wird untergliedert in:

- Impuls für Anwendungsentwicklung
- Projektvorbereitung
- Sensibilisierung für Usability (Gebrauchstauglichkeit)

Impulse für die Anwendungssystementwicklung sind auch die ersten Grundlagen für eine Projektplanung. Die Impulse kommen in der Regel aus den Anwendungsbereichen oder können auch durch Technologiesprünge oder strategische Um- oder Neuorientierung des Unternehmens verursacht sein. Prinzipiell sprechen wir von der Feststellung einer Mangelsituation. Die Projektvorbereitung und auch weitere Tätigkeiten, die üblicherweise dem Projektmanagement zuzuordnen sind, finden sich in diesem Modell in einzelnen Phasen bzw. deren untergeordneten Tätigkeitsbereichen. Da das Projektmanagement in der Systementwicklung gemeinhin als Querschnittfunktion wahrgenommen wird, die weit über Fachspezifika hinausgeht, wird dies auch für das Usability Engineering unterstellt. Wir merken jedoch vor, dass es so etwas wie eine grundsätzliche Forderung nach begleitendem Usability Engineering gibt und dessen Notwendigkeit nicht (mehr) in Frage zu stellen ist.

Eine weitere Aufgabe in der Planungsphase ist die Sensibilisierung für Usability (Gebrauchstauglichkeit). Auch wenn die Notwendigkeit grundsätzlich nicht in Frage gestellt wird, ist eine Sensibilisierung bereits vor dem Projektbeginn essentiell. Um Usability Engineering sinnvoll in ein Projekt integrieren zu können, ist die Einbeziehung und Unterstützung der Beteiligten notwendig. Zu den Beteiligten zählen auf jeden Fall die Anwenderorganisation (Auftraggeber), die Herstellerorganisation (Auftragnehmer) und die Benutzer. Möglicherweise noch Zulieferer und weitere Stakeholder (Anspruchsgruppen der Unternehmung) der Auftraggeberorganisation. Um dies zu erreichen, muss den Beteiligten ein grundlegendes Wissen über Sinn und Zweck von Usability und Usability-Maßnahmen vermittelt werden (Beispielsweise Reduzierung des Nutzungsaufwands, Steigerung der Produktivität).

Kontextanalyse

Die Phase der Kontextanalyse wird untergliedert in:

- Kosten-/Nutzenanalyse
- Ist-Analyse
- Analyse des Nutzungskontextes (Benutzer/Arbeitsaufgaben/Arbeitsumgebung)
- Spezifikation von Nutzungsanforderungen

- Lead-User-Interview
- Ausschreibung mit Usability als geforderte Produktqualität

Der Nutzungskontext eines Produkts besteht gemäß DIN EN ISO 9241-11 (1999) aus den Benutzern, ihren Arbeitsaufgaben, den relevanten Arbeitsmitteln sowie der physischen, organisatorischen und sozialen Arbeitsumgebung. All diese Faktoren können die Gebrauchstauglichkeit eines Systems beeinflussen und sollten daher vor der Festlegung von Nutzungsanforderungen analysiert werden.

Deswegen werden während der Kontextanalyse Informationen des Nutzungskontexts gesammelt und analysiert. Dazu zählt die Analyse der Einsatzumgebung und der Einsatzbedingungen des Systems. Außerdem sind folgende Informationen wichtig: die Arbeitsstruktur, die Arbeitsaufgaben und -Prozesse sowie die Nutzer und Rollenverteilungen. Nutzungsanforderungen werden formuliert und soweit möglich präzisiert und validiert. Die Kontextanalyse sollte in Anlehnung an DIN EN ISO 9241-110 (2006) durchgeführt werden. Die Informationen und Auswertungen der Kontextanalyse können u.a. durch Kontextszenarien und Use-Case Szenarien dargestellt werden.

Die Hauptverantwortung für diese Tätigkeiten liegt beim Auftraggeber bzw. Anwenderunternehmen.

Projektvorbereitung

Die Phase der Projektvorbereitung wird untergliedert in:

- Marketingunterstützung
- Angebot und Vertrag
- Planung von Usability-Aktivitäten
- projektspezifische Festlegungen
- Rollenverteilung

Es wird besonderer Wert darauf gelegt, dass bereits zu Beginn des Herstellerprojektes dem Thema „Gebrauchstauglichkeit“ größte Aufmerksamkeit gewidmet wird. Um notwendige Usability-Aktivitäten entsprechend abzusichern (sowohl finanziell als auch im Hinblick auf personelle Ressourcen) wird empfohlen, Gebrauchstauglichkeit in der Ausschreibung explizit zu fordern. Dies umso mehr als Aktivitäten des Usability Engineering in den Anfangsphasen der Entwicklung Mehrkosten verursachen können. Eine Amortisation dieser Kosten macht sich meist erst nach der Einführungsphase bemerkbar. Kosten und Nutzen liegen zeitlich oft weit auseinander (abhängig von der Projektdauer). Es ist Sache des Auftraggebers, diese Aktivitäten einzufordern und auch das entsprechende Budget dafür zur Verfügung zu stellen. Eine gesetzliche

Grundlage für Usability-Aktivitäten bildet z.B. die Bildschirmarbeitsplatzverordnung in Deutschland. DIN EN ISO 9001 (2000) zertifizierte Organisationen sind dazu verpflichtet.

Für die erfolgreiche Umsetzung von Usability-Aktivitäten ist neben dem Usability Experten auf Seiten des Anwenderunternehmens (Enterprise Engineer) der Usability Agent (= Usability Experte auf Seiten des Herstellers) wichtig. Die Prüfung der Möglichkeiten und Kompetenzen (Usability Expertise) des Hersteller ist essentiell und muss vor Auftragsvergabe erfolgen. Sie soll unbedingt Bestandteil des Vertrages sein. Ist die Vertragsgestaltung zwischen Anwenderorganisation und Herstellerorganisation geklärt, sind bei Bedarf Marketingaktivitäten zu unterstützen (z.B. für off-the-shelf-Produkte). Ein Kernthema ist zweifelsohne die Planung von Usability-Aktivitäten.

Hierbei geht es darum, den Gestaltungsrahmen für Usability Engineering Prozesse auf das konkret vorliegende Entwicklungsprojekt anzuwenden. Dabei muss entschieden werden, welche Aktivitäten, Methoden und Hilfsmittel im Projekt einzuplanen sind, und zwar ab der Phase Design und Validierung bis zur Einführung des Produkts mit anschließender Nutzung und Pflege. Des Weiteren sollte hier die spezifische Rollenverteilung und Verteilung der Verantwortlichkeiten zwischen Auftragnehmer und Auftraggeber im Projekt erfolgen. Unterschiedliche Gewichtung der Phasen bzw. Projektbausteine, z.B. abhängig davon, ob es sich um eine Neuentwicklung oder eine Weiterentwicklung handelt, ist möglich. Die Anwendung des Gestaltungsrahmens erfolgt in zwei Stufen:

1) Entwicklungsrelevante Projektbausteine werden aus dem Gestaltungsrahmen ausgewählt (Gestaltung des konkreten Prozesses).
2) Zur Durchführung geeignete Methoden und Hilfsmittel werden den Bausteinen zugeordnet.

Die Bausteine des Modells werden also bedarfsgerecht zusammengestellt. Für die hinterlegten und grob beschriebenen Aufgaben gilt es, geeignete Methoden auszuwählen. Methoden sind im DATech Gestaltungsrahmen partiell erwähnt. Ausgewählte Methoden sind grob beschrieben. Zuordnung und Beschreibung reichen jedoch nicht aus, Anwender der Methoden konkret zu unterstützen.

Die Verantwortung für alle Prozessschritte im Auftraggeber-Projekt wird dem Auftraggeber zugeordnet. Dem Auftragnehmer wird nach DATech (2008, S. 24) lediglich eine Beteiligung zugedacht. In diesem Punkt sind grafische Darstellung und Verantwortungszuordnung nicht stimmig.

Design und Validierung

Die Phase des Designs und der Validierung wird untergliedert in:

- Analyse des Nutzungskontextes (Benutzer/Arbeitsaufgaben/Arbeitsumgebung)
- Aufgabendesign
- Interaktionsdesign
- Usability Prototyping (Nutzungskonzept/Entwurf/Evaluierung)
- Oberflächendesign (Informationsdarstellung)
- Claims-Analysis
- Validierung der Systemanforderungen
- Benutzerdokumentation

In dieser Phase werden die Ergebnisse von Aktivitäten der vorausgegangenen Phasen aufgearbeitet. Es gilt, dem Herstellerunternehmen das gesammelte entwicklungsrelevante Wissen zu vermitteln und ein einheitliches gemeinsames Verständnis der Nutzungsanforderungen und der Systemanforderungen zwischen Anwender- und Herstellerorganisation herzustellen. Dabei ist auch in dieser Phase die Beteiligung der Nutzer aus den betroffenen Bereichen entscheidend.

Es kann in dieser Phase der dialektische Problemlösungsprozess in Zusammenarbeit mit dem Herstellerunternehmen nochmals durchgeführt werden (Analyse des Nutzungskontexts). Der Hersteller kann unter Umständen zu einer Erweiterung der Produktidee durch technologische Gestaltungsoptionen beitragen. Da sich das Anwenderunternehmen im Vorhinein mit seinen Anforderungen und denen seiner Nutzer auseinandergesetzt hat, kann verhindert werden, dass durch das Herstellerunternehmen nicht notwendige bzw. überflüssige Funktionalitäten und Komponenten empfohlen und in der Folge implementiert werden. In diesem Rahmen sind auch Forderungen (= Claims) von Seiten der Stakeholder zu klären und bei Bedarf in validierbare Anforderungen zu überführen.

Die aufgelisteten Aktivitäten dieser Phase unterstützen die Erarbeitung des Systementwurfs. Betont wird dabei die Validierung von Anforderungen und Zwischenergebnissen. Dies kommt nicht zuletzt durch die Benennung der Phase zum Ausdruck.

Nachdem die Analysen des Nutzungskontexts aus Sicht des Herstellers abgeschlossen sind, gilt es, ein detailliertes Aufgabendesign sowie ein detailliertes Interaktionsdesign zu erstellen. Hierbei sind nicht nur die technisch-fachlichen Aspekte relevant, sondern es steht die benutzer- und nutzungsfreundliche Komponente der Gestaltung

im Vordergrund. Bereits in dieser Phase ist Wert auf die begleitende Erstellung einer Benutzerdokumentation zu legen.

Für Phase Design und Validierung trägt das Herstellerunternehmen die Hauptverantwortung.

Implementierung und Test

Die Phase Implementierung und Test wird untergliedert in:

- Design-Regelwerk fortschreiben
- Unterstützung des Designs und Begleitung der Implementierung
- Entwicklungsbegleitende Usability-Tests
- Abnahmetest

In der Implementierungsphase werden die entwickelten Konzepte und Entwürfe, auch jene der Benutzungsoberfläche, umgesetzt und getestet. Des Weiteren findet die Systemintegration (Zusammenführung der Entwurfsteile) statt. Ausgehend von den Entwurfsvorschlägen aus Usability-Sicht durchläuft das System während der Realisierung zyklisch die Aktivitäten Design und Validierung sowie Implementierung und Test (DATech, 2008, S. 38). Das heißt, wenn z.B. Probleme bei der Implementierung erkannt werden, können Aktivitäten des Entwurfs wiederholt werden, um Vorschläge für die Problemlösung zu finden.

Der gesamte Implementierungsprozess sollte von Ansprechpartnern des Usability-Engineering (z.B. Usability-Experte, Usability-Agent) begleitet werden. Dadurch sollen einerseits die entwicklungsbegleitenden Evaluationen sichergestellt und unterstützt werden, andererseits gilt es, bei Entscheidungen, die Auswirkungen auf die Gebrauchstauglichkeit haben, den Bezug zum vorangegangenen Konsensfindungsprozess zu gewährleisten. Beispielsweise sollten Dialogentwürfe vom Usability-Engineer oder Usability-Agent bewertet werden.

In dieser Phase sollten die Nutzer für die Validierung der Ergebnisse gegen die Nutzungsanforderung und somit für die Abnahme der Systemlösung herangezogen werden. Sobald Nutzungsprobleme aufgedeckt werden, sollten diese bearbeitet und beseitigt werden. Ansonsten kommen diese spätestens in der Nutzungsphase zum Vorschein. Im letzteren Fall müssen diese im Rahmen der vertraglichen Vereinbarungen bearbeitet werden. Änderungen im Nachhinein bedeuten jedoch in der Regel wesentlich höhere Kosten. Des Weiteren können Grafik-Designer in dieser Phase hilfreich sein (DATech, 2008). Bei diesen Aktivitäten sind die Anwenderorganisation und die Herstellerorganisation gleichermaßen gefordert. Das Herstellerunternehmen trägt die Hauptverantwortung.

Nutzung und Pflege

Die Phase Nutzung und Pflege wird untergliedert in:

- Einführungsprojekt
- Konfiguration des Produkts
- Benutzerschulung
- Feedback von Benutzern auswerten
- Pflegeprozess
- Vorbereitung des nächsten Releases

Die in der Gliederung angeführten Teilaspekte zur Phase der Nutzung und Pflege beinhalten auch den Übergang der entwickelten Lösung vom Hersteller- zum Anwenderunternehmen. Dieser Übergang lässt sich meist nicht klar abgrenzen.

Zu Beginn dieser Phase ist die Einführung der entwickelten Lösung vorgesehen. Üblicherweise umfassen Einführungsprojekte die Einsatzanalyse, die Prüfung verfügbarer Hardware, die Installationen, das *Customizing* (Konfiguration), die Festlegung der Zugriffsrechte, die Anpassung der Arbeitsplätze, die Altdatenübernahme, bei Bedarf einen Wartungsvertrag, Schulungen und die Anpassung von Bedienungsanleitungen, Einsatzvorbereitung und Einrichtung des Supports (DATech, 2008). Das Ziel des Einführungsprojekts ist es, die erste Nutzungsphase mit einem neuen System zeitlich befristet zu begleiten. Es gilt, anfängliche Nutzungsprobleme im Nutzungskontext zu erkennen und nach Möglichkeit zu beheben. Dabei ist die aktive Kommunikation mit den Nutzern entscheidend. Zwar liegt die Verantwortung zur Aufdeckung von Nutzungsproblemen bei der Anwenderorganisation, allerdings ist für die Beseitigung von Problemen eine kooperative Zusammenarbeit mit der Herstellerorganisation und deren Zulieferern wichtig.

Trotz der Berücksichtigung von ergonomischen Anforderungen und der sorgfältigen Entwicklung von *Nutzungsanforderungen* und ihrer Validierung werden erfahrungsgemäß viele Nutzungsprobleme erst nach Validierung und Nutzung klar. Deswegen ist es wichtig in der Nutzungsphase, insbesondere in der Einführungsphase, für alle auftretenden Probleme (Fehler und Mängel) kompetente und verfügbare Ansprechpartner, sowohl bei der Anwender- wie auch bei der Herstellerorganisation, verfügbar zu haben.

Schulungen für Benutzer sollten nicht nur auf die Nutzung und den Umgang mit dem System vorbereiten, sondern bei Bedarf auch auf organisatorische und soziale Änderungen eingehen. Dementsprechend sind Schulungen bereits vor der Einführung durchzuführen. Nach Möglichkeit sollten Schulungen über den gesamten Produktle-

benszyklus institutionalisiert werden. Die Aktivität würde somit besser zur Benutzerbeteiligung als Querschnittfunktion zugeordnet.

In der ersten Nutzungsphase sollten alle Rückmeldungen zweckmäßig aufbereitet werden, damit ihre Bearbeitung im Zuge der Anpassung des Systems erleichtert wird. Es sollte beispielsweise im Rahmen von Schulungen auf Probleme bei der Einarbeitung der Nutzer in das neue System eingegangen werden. Über Beobachtungen, Befragungen und Veranstaltungen können Probleme kurzfristig behoben und in den Einführungsprozess aktiv integriert werden.

Der Pflegeprozess läuft anfänglich parallel zum Einführungsprojekt, zieht sich allerdings über den gesamten Nutzungszeitraum des Systems. Die Leistungen gehen über jene eines üblichen Wartungsvertrags hinaus.

Für die Phase der Nutzung und Pflege trägt die Anwenderorganisation die Hauptverantwortung.

Benutzerbeteiligung und Qualitätsmanagement als Querschnittfunktionen

Benutzerbeteiligung und Qualitätsmanagement werden in dem von der DATech entworfenen Gestaltungsrahmen als Querschnittfunktionen deklariert. Für die Querschnittfunktionen wurden in der Übersicht keine konkreten Aktivitäten vorgeschlagen. Das Qualitätsmanagement als wichtige Querschnittfunktion und seine Beziehung zum Usability Engineering ist generell sehr eng (DATech 2008). Durch die Hervorhebung der Benutzerbeteiligung als Querschnittfunktion wird ihre Bedeutung unterstrichen.

Nachfolgend sind in Anlehnung an DATech (2008) Anlässe für die Zusammenarbeit mit Benutzern aufgelistet, die in jedem Projekt gegeben sind:

- **Einbeziehung der Benutzer bei der Erstellung des Nutzungskonzepts** – Grundlage der Diskussionen mit den Benutzern ist das von den Usability-Requirements-Ingenieuren entworfene Nutzungskonzept. Die Gespräche über das Nutzungskonzept sollten so früh wie möglich durchgeführt werden. Dabei sollten die Benutzer dazu angeregt werden, Wünsche und Erwartungen bezüglich der Nutzung des zu entwickelnden Systems zu äußern.
- **Validierung von Nutzungsanforderungen durch die Benutzer** – Es gilt bereits die gesammelten Nutzungsanforderungen darauf zu prüfen, ob sie hinreichend genau das ausdrücken, was später das Anwendungssystem leisten sollte. Falls bei der Validierung festgestellt wird, dass man eine Nutzungsanforderung noch nicht hinreichend geklärt bzw. noch keine hinrei-

chend genaue Einigung über ein gemeinsames Verständnis erreicht hat, muss dies nachgeholt werden.

- **Prototypen gemeinsam durcharbeiten** – Prototypische Lösungsvorschläge und Interaktionsentwürfe sollten mit den Benutzern durchgesprochen werden, z.B. mit Anwendung eines Walkthrough-Verfahrens.
- **Iterative Usability-Evaluierungen** – Durch die Bewertung und iterative Verbesserung von Prototypen werden noch offene Fragen identifiziert oder Entwurfsentscheidungen vorbereitet. Dabei sollten alle wichtigen Abläufe des Systems, meist handelt es sich um Kernprozesse der Arbeitsbereiche, mit Benutzern aus allen relevanten Zielgruppen am Dialogsystem durchgearbeitet und hinsichtlich Effizienz und Zufriedenstellung bewertet werden.
- **Iterative Usability-Tests am integrierten System** – Übergreifende und zusammenhängende Aufgaben und Abläufe sollten am Gesamtsystem mit den Benutzern bewertet werden.

Als unerlässlich für den Erfolg des Usability-Engineering wird die kontinuierliche Beteiligung der künftigen Benutzer des zu entwickelnden Systems im Analyse- und Gestaltungsprozess angesehen. Der Erfolg der Benutzerbeteiligung hängt davon ab, dass geeignete Benutzer ausgewählt und auf ihre Rolle vorbereitet werden (O'Neill, 2000). Als wichtige Methode zur Benutzerbeteiligung werden regelmäßige, moderierte Ergonomie-Workshops erwähnt. Hierbei können wesentliche Prozessschritte und Arbeiten, die zusammen mit den Benutzern durchzuführen sind, in Angriff genommen werden. Auch sogenannte Fokus-Gruppen können hierzu eingerichtet werden. Gruppen-Workshops erleichtern die Herbeiführung von Kompromissentscheidungen (DATech, 2008).

Grad und Entwicklung der Benutzerzentriertheit können wiederum mittels Modellen des Qualitätsmanagements (z.B. CMMI oder UMM) gemessen werden. Der Level an Benutzerzentriertheit einer Organisation gibt dann Auskunft darüber, wie innerhalb der Organisation Belange der Benutzerzentrierung am besten umgesetzt werden. Die Modelle implizieren, dass eine Verbesserung des Usability Engineering Prozesses nur kontinuierlich und nicht ad hoc möglich ist (Earthy, 1998).

5.3 Einflussgrößen für Systemakzeptanz im UTAUT-Modell

Alle nachstehend angeführten Einflussgrößen für das UTAUT-Modell, die erläuternden Definitionen sowie eine mögliche Messskala zu der jeweiligen Größe entspringen den Untersuchungen von Venkatesh et al. (2003, S. 448-456). Es handelt sich

dabei um grundlegende Konstrukte zu den Modellbausteinen des UTAUT-Modells. Die jeweils relevanten zusätzlichen Gewichtungen sind im Anschluss an jede Tabelle angegeben.

Erwartungen zum Arbeitsaufwand		
Einflussgrößen	**Definition**	**Messskala**
Wahrgenommene einfache Nutzung	Repräsentiert jenen Grad an Anstrengungsfreiheit, den eine Person mit der Nutzung des Systems zu erreichen glaubt.	- Zu lernen, wie das System zu bedienen ist, wird einfach für mich sein. - Ich werde es einfach finden, mit dem System zu machen, was ich machen will. - Meine Interaktion mit dem System wird klar und verständlich sein. - Ich werde das System flexibel in der Interaktion finden. - Es wird einfach für mich sein, ein fachkundiger, gewandter Anwender des Systems zu werden. - Ich werde das System einfach in der Nutzung finden.
Komplexität	Repräsentiert jenen Grad, mit dem das System als relativ schwierig zu verstehen und zu nutzen wahrgenommen wird.	- Das System zu nutzen nimmt zu viel Zeit in Anspruch, welche zu Lasten meiner laufenden Verpflichtungen geht. - Mit dem System zu arbeiten ist kompliziert, es ist schwierig zu verstehen, was da passiert. - Die Nutzung des Systems bringt viel Zeitaufwand für mechanische Arbeit mit sich (z.B. Dateninput). - Es beansprucht zu viel Zeit zu lernen, wie das System zu nutzen ist, damit diese Anstrengungen gerechtfertigt sind.
Einfache Nutzung	Repräsentiert jenen Grad, mit dem die Nutzung einer Innovation als schwierig zu nutzen wahrgenommen wird.	- Meine Interaktion mit dem System ist klar und verständlich. - Ich glaube, dass es leicht ist, mit dem System zu machen was ich machen will. - Allgemein glaube ich, dass das System einfach zu benutzen ist. - Es ist einfach für mich, die Bedienung des Systems zu erlernen.

Tab. 7: Erwartungen zum Arbeitsaufwand im UTAUT-Modell

Die Erwartungen zum Arbeitsaufwand gilt es zusätzlich mit den Faktoren Geschlecht, Alter und Erfahrung zu gewichten.

Erleichternde Bedingungen		
Einflussgrößen	**Definition**	**Beschreibung/Skalierung**
Wahrgenommene Steuerung, Kontrolle des Verhaltens	Reflektiert die Sichtweise von internen und externen Bedingungen für das Verhalten und spannt den Bogen von erleichternden Bedingungen Hilfsmittel und Technologie betreffend bis zum Selbstvertrauen.	- Ich habe die Kontrolle über die Systemnutzung. - Ich habe die notwendigen Ressourcen (Hilfsmittel) zur Nutzung des Systems. - Ich verfüge über das Wissen, das zur Systemnutzung erforderlich ist. - Wenn Hilfsmittel, Einsatzmöglichkeiten und Wissen zur Systemnutzung gegeben sind, ist es für mich einfach, das System zu nutzen. - Das System ist nicht mit anderen Systemen, die ich nutze, kompatibel.
Erleichternde Bedingungen	Objektive Faktoren im Anwendungsbereich, bei denen die Beobachter sich einig sind, dass sie die Tätigkeiten erleichtern, den Computersupport eingeschlossen.	- Bei der Auswahl des Systems war für mich Beratung verfügbar. - Spezielle Schulungen das System betreffend waren für mich abrufbar. - Eine spezielle Person (oder eine Gruppe von Personen) steht für die Unterstützung bei Systemschwierigkeiten zur Verfügung.
Kompatibilität	Jener Grad, mit dem die Innovation wahrgenommen wird als konsistent mit existierenden Werten, Bedürfnissen und Erfahrungen von potenziellen Nutzern.	- Die Nutzung des Systems passt mit allen Perspektiven meiner Arbeit zusammen. - Ich denke die Nutzung des Systems passt gut zu der Art zu arbeiten, wie ich es mag. - Die Nutzung des Systems passt zu meinem Arbeitsstil.

Tab. 8: Erleichternde Bedingungen im UTAUT-Modell

Die erleichternden Bedingungen gilt es zusätzlich mit den Faktoren Alter und Erfahrung zu gewichten.

Leistungserwartungen an das System		
Einflussgrößen	**Definition**	**Beschreibung/Skalierung**
Wahrgenommene Zweckmäßigkeit	Ist jener Grad an Verbesserung der beruflichen Leistung, den eine Person durch die Nutzung eines bestimmten Systems subjektiv glaubt erreichen zu können.	Die Nutzung des Systems… … in meiner Arbeit befähigt mich, meine Aufgaben schneller zu erledigen. … verbessert meine berufliche Leistungsfähigkeit. … in meiner Arbeit wird meine Produktivität steigern. … wird meine Effektivität am Arbeitsplatz erhöhen. … macht es mir leichter meine Arbeit zu machen. - Ich werde das System als wertvoll für meine Arbeit wahrnehmen.

Leistungserwartungen an das System		
Einflussgrößen	Definition	**Beschreibung/Skalierung**
Extrinsische Motivation	Repräsentiert die Sichtweise der Nutzer, wie diese eine Leistung wahrnehmen wollen; wahrgenommene Leistung ergibt sich entweder aus der Tätigkeit selbst oder liegt z.B. in einer Verbesserung der Arbeit oder einer anderen Wirkung.	Die Skalierung ist die gleiche wie für die wahrgenommene Zweckmäßigkeit
Job-Fit	Diese Größe zeigt, wie die Möglichkeiten eines Systems die individuelle berufliche Leistungsfähigkeit fördern bzw. steigern können.	Die Benutzung des Systems… … wird keinen Effekt auf meine berufliche Leistungsfähigkeit haben (bzw. in umgekehrten Sinn, dass ein Effekt erzielbar sein wird). … kann eine Verringerung der Zeit bewirken, die ich für wichtige berufliche Verantwortlichkeiten benötige. … kann die Qualität meines beruflichen Outputs signifikant steigern. … kann die Effektivität der beruflichen Aufgabenerfüllung steigern. … kann die Quantität des Outputs bei gleichen Anstrengungen steigern. - Der umfassende Rahmen, mit dem das System die berufliche Tätigkeit unterstützen kann, betrifft alle Aufgaben (für diesen Punkt wird eine andere Skalierung verwendet).
Relativer Vorteil	Gibt jenen Grad an Vorteilhaftigkeit an, mit dem die Benutzung des Systems als besser wahrgenommen wird, in Relation zur Benutzung der Vorgängerversion.	Die Benutzung des Systems… … ermöglicht es mir, Aufgaben schneller durchzuführen. … verbessert die Qualität der Arbeit, die ich mache. … erleichtert es mir, meine Arbeit zu machen. … steigert meine Effektivität am Arbeitsplatz. … steigert meine Produktivität.
Ergebniserwartungen	Ergebniserwartungen in Beziehung zu den Konsequenzen des Verhaltens. Basierend auf empirischen Nachweisen werden diese in Leistungserwartungen (beruflich) und persönliche Erwartungen (persönliche Ziele) unterteilt.	Wenn ich das System nutze… … werde ich meine Effektivität bei der Arbeit steigern. … werde ich weniger Zeit mit Routineaufgaben verbringen. … werde ich die Qualität meiner Arbeitsergebnisse verbessern. … werde ich bei gleichem Einsatz meinen Output steigern. … werden mich meine Mitarbeiter als kompetent wahrnehmen. … werde ich meine Chancen für einen beruflichen Aufstieg fördern. … werde ich meine Chancen, eine Lohnerhöhung zu erhalten, steigern.

Tab. 9: Leistungserwartungen an das System im UTAUT-Modell

Die Leistungserwartungen an das System sind zusätzlich mit den Faktoren Geschlecht und Alter zu gewichten.

Soziale Einflüsse		
Einflussgrößen	**Definition**	**Beschreibung/Skalierung**
Subjektive Normen	Die Wahrnehmung einer Person, dass die meisten Menschen, die wichtig für sie sind, denken, sie sollte - oder sie sollte eben nicht - die Leistung in Frage stellen.	- Menschen, die mein Verhalten beeinflussen, meinen, dass ich das System nutzen sollte. - Menschen, die für mich wichtig sind, meinen, dass ich das System nutzen sollte.
Soziale Faktoren	Die Verinnerlichung (Internalisierung) des Menschen im Vergleich zur subjektiven Kultur der Referenzgruppe und spezifische zwischenmenschliche Übereinkünfte, zu denen einzelne Menschen mit anderen in speziellen sozialen Situationen gekommen sind.	- Ich nutze das System nach Maßgabe der Arbeitskollegen, die das System nutzen. - Die Führungskräfte dieses Unternehmens waren sehr behilflich bei der Nutzung des Systems. - Mein Vorgesetzter unterstützt mich sehr was die Nutzung des Systems für meine Arbeit anbelangt. - Generell hat die ganze Organisation die Nutzung des Systems unterstützt.
Image	Repräsentiert jenen Grad, mit dem die Nutzung der Innovation als Faktor, der das Image des Einzelnen in diesem sozialen System verbessert, wahrgenommen wird.	- Menschen in meinem Unternehmen, die das System nutzen, genießen mehr Ansehen als jene, die das nicht tun. - Menschen in meinem Unternehmen, die das System nutzen, finden große Beachtung. - Das System zu haben, ist ein Statussymbol für mein Unternehmen.

Tab. 10: Soziale Einflüsse im UTAUT-Modell

Die sozialen Einflüsse sind zusätzlich mit den Faktoren Geschlecht, Alter, Erfahrung und der Freiwilligkeit der Benutzung zu gewichten. Keine der aufgelisteten Einflussgrößen ist signifikant bei der freiwilligen Nutzung von Systemen. Erst wenn die Nutzung verpflichtend ist, werden diese Einflussgrößen signifikant.

Einstellung zur Technologienutzung – nicht direkt wirksame Faktoren		
Einflussgrößen	**Definition**	**Beschreibung/Skalierung**
Einstellung zum Verhalten	Die positiven oder negativen Gefühle eines Menschen, die darauf abzielen, ein Ziel-Verhalten (angestrebtes Verhalten) zu erreichen.	- Das System zu nutzen ist eine schlechte/gute Idee. - Das System zu nutzen ist eine unkluge/sinnvolle Idee. - Ich mag die Vorstellung der Systemnutzung (nicht). - Das System zu nutzen ist angenehm/unangenehm (widerlich).
Intrinsische Motivation	Objektive Faktoren im Anwendungsbereich, bei denen die Beobachter sich einig sind, dass sie die Tätigkeiten erleichtern, den Computersupport eingeschlossen.	- Ich empfinde die Nutzung des Systems als angenehm. - Die momentane Art, das System zu nutzen, ist ansprechend. - Ich habe Spaß bei der Systemnutzung.
Emotionen in Bezug auf die Benutzung	Gefühle wie Spaß, Begeisterung, Freude oder Bedrücktheit, Abscheu, Missmut oder Hass, die ein Mensch mit einer Tätigkeit in Verbindung bringt.	- Das System macht die Arbeit interessanter. - Mit dem System zu arbeiten macht Spaß. - Das System ist gut für manche Arbeiten, aber nicht für die Arbeit, die ich machen will (umgekehrte Skalierung).
Emotionen allgemein	Die Vorlieben eines Menschen im Hinblick auf das Verhalten.	- Ich arbeite gern mit dem System. - Ich freue mich auf jene Aspekte meiner Arbeit, die eine Systemnutzung verlangen. - Das System zu nutzen ist für mich frustrierend (umgekehrte Skalierung). - Wenn ich einmal mit der Nutzung des Systems begonnen habe, ist es schwer, wieder damit aufzuhören. - Wenn ich das System nutze, wird mir schnell langweilig (umgekehrte Skalierung).

Tab. 11: Einstellung zur Technologienutzung im UTAUT-Modell

Venkatesh et al. weisen in diesem Kontext darauf hin, dass diese Einflussfaktoren nicht direkt wirken.

5.4 Erhebungsmethoden

Primär dialogische Erhebungsmethoden

Interview

Interviews dienen dazu, Menschen über Sachverhalte eines Aufgabengebietes zu befragen und damit Ansichten und Meinungen zu erfahren. Ein Interview kommt in der Anwendung in unterschiedlichen Ausprägungen vor, z.B. als Einzelinterview oder Gruppeninterview aus Sicht der Anzahl der gleichzeitig befragten Personen. Aus Sicht der Vorgehensweise kann zusätzlich unterschieden werden zwischen unstrukturier-

tem, fokussiertem, semistrukturiertem und strukturiertem Interview. Die einzelnen Typen von Interviews sind für unterschiedliche Anforderungen geeignet.

Einzelinterview

Grundsätzlich können bei Einzelinterviews in kürzerer Zeit mehr Informationen gesammelt werden als bei Gruppeninterviews. Wittlage (1993) empfiehlt aus Kostengründen Einzelinterviews zu führen, weil hier eindeutigere Ergebnissen erzielt werden können und Meinungsverschiedenheiten wie etwa Angst vor Sanktionen durch andere Gruppenmitglieder vermieden werden.

Gruppeninterview

Es gibt Sachverhalte, die nicht in Zusammenarbeit mit einer einzigen Person geklärt und erhoben werden können. Hierzu gehören insbesondere die Sachverhalte, die bereichsübergreifend sind oder an den Schnittstellen von Funktionsbereichen liegen. In diesen Fällen bietet sich der Einsatz von Gruppeninterviews an, da die Zeit für die Erhebung gegenüber einer Einzelbefragung stark reduziert wird. Außerdem können durch Gruppeninterviews Streitfälle und Widersprüchlichkeiten unmittelbar aufgelöst werden (Wedekind, 1976).

Als genereller Vorteil von Gruppendiskussionen ist zu erwähnen, dass sich fruchtbare Diskussionen entwickeln können (Wurch, 1983). Allerdings können die Ergebnisse eines Gruppeninterviews auch durch Meinungsverschiedenheiten und Angst vor Sanktionen durch andere Gruppenmitglieder verzerrt sein. Deswegen empfiehlt Kargl (1990) bei Gruppeninterviews darauf zu achten, dass die Teilnehmer aus den gleichen Hierarchiestufen stammen, um Hemmungen bei der Beantwortung von Fragen zu vermeiden. Ist dies nicht möglich, sind die Anforderungen an den Interviewer bzw. Moderator ungleich höher. Bei Detailerörterungen sollte die Teilnehmerzahl nicht mehr als sieben betragen, um arbeitsfähig zu bleiben (Wedekind, 1976).

Unstrukturiertes Interview

Bei einem unstrukturierten Interview werden einfach allgemeine Fragen gestellt und dabei so viel an Information wie möglich aufgenommen. Diese Art von Interview erfordert den kleinsten Vorbereitungsaufwand. Hingegen sind der Nachbereitungsaufwand und die Auswertung der Ergebnisse umso aufwändiger.

Fokussiertes Interview

Beim fokussierten Interview handelt es sich um ein Gespräch mit offenen Fragen bezogen auf eine Liste von Themen und Gegenständen, die erfasst und diskutiert werden (Bradshaw, 1997). Diese Art von Interview eignet sich beispielsweise zur Anforderungsanalyse, um den Einsatzbereich des Anwendungssystems und die Benutzerbelange kennen zu lernen. Dabei sollte der Befragte vorab über die Themen

informiert werden, damit die (mentale) Vorbereitung des Interviewten ermöglicht wird. Jeder Themenbereich kann während des Interviews vertieft werden um Fakten über Benutzerrollen, Problembereiche, Funktionalitäten usw. zu erfahren. Es empfiehlt sich hierbei die Antworten aufzunehmen, da dadurch die spätere Analyse erleichtert wird (Macaulay, 1996).

Strukturiertes Interview

Ein strukturiertes Interview läuft nach einer strikten Agenda und einer Liste mit festgelegten, spezifischen Fragen ab, die sich auf Eigenschaften des Systems beziehen (Bradshaw, 1997). Diese Interviewart ist gut geeignet für die Sammlung von detaillierten Informationen. Der Interviewer sollte allerdings ein gutes Wissen über den Anwendungsbereich und die Begrifflichkeiten haben, um bei Bedarf die Befragung entsprechend den Antworten bestmöglich anpassen zu können (Macaulay, 1996).

Semistrukturiertes Interview

Beim semistrukturierten Interview ist der Rahmen für das Gespräch nicht so eng gesteckt wie beim strukturierten Interview, jedoch auch nicht so offen wie beim fokussierten Interview.

Welcher Interviewtyp schlussendlich eingesetzt wird, hängt davon ab, wie der Informationsbedarf eingeschätzt wird. Für Interviews zur vorbereitenden Erhebung der Situation in der Anwenderorganisation und zur Ermittlung von Systemanforderungen ist immer ein Trade-off zu machen zwischen den einzusetzenden Ressourcen für das Interview (Vor- und Nachbereitungszeit) und der gewünschten Informationstiefe.

Weitere Beschreibungen zu Interviewtechniken finden sich z.B. bei Moser & Kalton (1971), Schienmann (2002), Rupp (2007), Pohl (2008). Eine Diskussion von Problemen, die bei persönlichen Interviews entstehen können, wird z.B. von Suchman & Jordan (1990) ausgeführt.

Primär hermeneutische Erhebungsmethoden

Fragebogen

Der Einsatz von Fragebögen eignet sich für das Herausfinden einfacher Sachverhalte, die sich zeitlich kaum ändern und dem Befragten tief eingeprägt sind. Außerdem lassen sich Routinefragen, die sich z.B. auf Details der Aufbauorganisation (Unterstellungsverhältnisse, Abteilungszugehörigkeit) beziehen, gut per Fragebogen stellen (Wedekind, 1976). Fragebögen eignen sich nicht zur Klärung von Sachverhalten, für die es keine eindeutige Terminologie gibt. Das heißt auch, dass nur bekannte Informationen abgefragt werden können. Fragebögen können zu Beginn der Erhebungsphase hilfreich sein um herauszufinden, welche Eigenschaften ein geplantes System besitzen soll (Gorguen & Linde, 1993). Ansonsten sollten schriftliche Be-

fragungen in der Anforderungsanalyse nur als Ergänzung zum Interview aufgefasst werden (Lehmann, 1999).

Ein Fragebogen muss gut vorbereitet sein, d.h. die Erstellung bedingt einen hohen Vorbereitungsaufwand. Insbesondere auf die Formulierung der Fragen ist viel Wert zu legen. Sie sollten zweifelsfrei, leicht verständlich (Lehmann, 1999) und für den Befragten nicht verwirrend sein. Dem Befragten muss der Zweck der Untersuchung mitgeteilt werden. Die Fragen sollten einer sinnvollen Gliederung unterliegen. Es wird mindestens die Einteilung in die vier folgenden Fragegruppen empfohlen: Einleitungsfragen, Angaben zur Person, Sachfragen, Kontrollfragen.

Ergebnisse von Befragungen mittels Fragebogen sind systematisch auszuwerten und kritisch zu hinterfragen. Insbesondere bei anonymen Befragungen und Online-Befragungen kann man nicht unbedingt von der Richtigkeit der Antworten ausgehen. Laut Acker (1966) können Schönfärberei, Vertuschen von Unfähigkeiten, die Furcht vor Neuerungen wie auch ein mangelndes Erinnerungsvermögen zu groben Verzerrungen der Ergebnisse führen. Ein weiterer Nachteil von Fragebögen ist, dass sie häufig in Gemeinschaftsarbeit ausgefüllt werden, wodurch es zu zusätzlichen Verzerrungen kommt. Die Bearbeitung eines Fragebogens ist eine unbeliebte und zeitraubende Aufgabe, die im Allgemeinen nicht ernst genommen wird, wenn nicht mit der „notwendigen Autorität nachgefasst wird“ (Wedekind, 1976). Aus diesem Grund sollten wichtige Aussagen durch direkte Kommunikation mittels Methoden wie z.B. Interviews erhoben oder zumindest durch sie validiert werden.

Der Vorteil von Fragebögen liegt in den vergleichsweise geringen Kosten, z.B. in Relation zur Durchführung und Auswertung von Interviews. Des Weiteren erhält man mit Fragebögen schriftliche Ergebnisse, die auch für statistische Auswertungen benutzt werden können, wenn eine ausreichende Anzahl von Beteiligten (repräsentative Grundgesamtheit) befragt wurde. Ein weiterer Vorteil ist, dass der Interviewer den Befragten nicht beeinflussen kann. Allerdings kann er den Befragten somit auch nicht positiv beeinflussen, beispielsweise ihn zur Mitarbeit motivieren oder Unklarheiten in der Fragestellung beseitigen (Friedrichs, 1990, S. 236).

Dokumentenanalyse

Bei der Dokumentenanalyse (auch Schriftgutanalyse genannt) werden die Informationen aus ausgewählten Dokumenten gesammelt und ausgewertet. Zu Dokumenten einer Organisation zählen alle schriftlichen und bildlichen Aufzeichnungen, die Informationen über das Unternehmen, den Arbeitskontext (Arbeitsanalyse), die Aufbau- und Ablauforganisation (Organigramme, Workflowbeschreibungen) sowie die hergestellten Produkte oder Dienstleistungen liefern. Beispiele hierfür sind Schulungsunterlagen, Gesetze und Richtlinien (IT-Governance), Betriebsanleitungen, Stel-

lenbeschreibungen, Formulare, Sitzungsprotokolle, Vereinbarungen, Software- und Projektdokumentationen (Lehner et al., 1991). Darüber hinaus kann auf Fachliteratur zurückgegriffen werden, um „Aussagen über das Fachgebiet und seine Terminologie aufzustellen“ (Lehmann, 1999), bzw. die Aussagen zu validieren und zu verfeinern.

Die Dokumentenanalyse dient in der Regel zur Vorbereitung weiterer Analyseaktivitäten und wird daher gerne am Beginn der Aussagensammlung eingesetzt. Sie eignet sich auch gut als Ergänzung zu anderen Erhebungstechniken, um beispielsweise die im Gespräch mit Anwendern und Experten gewonnen Aussagen zu validieren (Lehmann, 1999). Aufbauend auf der Analyse können bei Bedarf verbesserte Abläufe und Strukturen konzipiert werden.

Primär praktische Erhebungsmethoden

Beobachtung (teilnehmende)

Es wird zwischen verschiedenen Beobachtungsformen unterschieden. Die Spanne reicht vom passiven, verdeckten Beobachten bis hin zum offenen, teilnehmenden Beobachten des Anwendungsbereichs (Lehmann, 1999). Bei der teilnehmenden Beobachtung wirkt der Beobachter (Wissensingenieur, Requirements-Engineer, Analytiker, Enterprise Engineer, Entwickler) im Gegensatz zur passiven Beobachtung für eine begrenzte Zeitdauer an der Aufgabenerfüllung des Anwenders oder dem Problemlösungsprozess des Experten mit (Wittlage, 1993).

Im Rahmen einer teilnehmenden Beobachtung wird der Arbeitsablauf in einem Anwendungsgebiet mit Wissen der Anwender vom Analytiker beobachtet und protokolliert. Hierbei handelt es sich insbesondere um das Erfassen von wahrnehmbaren Sachverhalten.

Bei der verdeckten Beobachtung gibt sich der Beobachter, im Gegensatz zur offenen Beobachtung, nicht zu erkennen. Dadurch soll verhindert werden, dass sich durch die Anwesenheit des Beobachters die Abläufe verändern. Passives Beobachten im Allgemeinen ist weniger zeitaufwendig und vermittelt ein distanziertes Bild der Arbeitsabläufe, wobei jedoch nicht auf Details der Arbeitsausführung eingegangen werden kann. Die teilnehmende Beobachtung „erleichtert das Verständnis von Arbeitsabläufen und die exemplarische Einführung von Begriffen“ (Lehmann, 1999). Sie ist auch dann nützlich, wenn in Interviews widersprüchliche Behauptungen (bewusst oder unbewusst) aufgestellt wurden, die nicht geklärt werden können.

Die Beobachter sollten die Fähigkeit erlernt haben, Vorgänge in den Teilfunktionsbereichen systematisch beobachten und dokumentieren zu können. Das Beobachten als bewusstes Wahrnehmen relevanter Tatbestände ist nicht trivial.

Auch bei Beobachtungen kann zwischen strukturierten und unstrukturierten Beobachtungen (vergleichbar mit dem Interview) unterschieden werden. Bei der strukturierten Beobachtung werden, im Gegensatz zur unstrukturierten Beobachtung, alle zu erfassenden Merkmale vorher festgelegt (Wurch, 1983). Sie sind dadurch einfacher durchführbar, allerdings ist der Vorbereitungsaufwand größer.

Der Nachteil bei dieser Erhebungsmethode ist, dass sich das System durch die Beobachtung verändern kann. Dies kann sowohl bei teilnehmenden, als auch bei passiven Beobachtungen der Fall sein. In der Regel wird der Beobachtete über die Beobachtung unterrichtet und ähnlich wie beim Interview sind dadurch Verzerrungen der Realität während der Beobachtungsphase nicht auszuschließen. Darüber hinaus ist die Vorbereitung und Durchführung einer teilnehmenden Beobachtung sehr aufwendig (Wedekind, 1976). Deswegen sollte sie für die Wissenserhebung eingesetzt werden, wenn die notwendigen Informationen durch andere Erhebungsmethoden überhaupt nicht oder nicht genau genug festgestellt werden können. Im Zuge der Integration des Usability Engineering ist eine teilnehmende Beobachtung im Sinne der Benutzer kaum zu ersetzen.

Die Inspektion entspricht der teilnehmenden Beobachtung, allerdings ist das Augenmerk eher auf eine Zustandsanalyse der Tätigkeit und die Feststellung von Systemmerkmalen gerichtet, während bei der teilnehmenden Beobachtung der Fokus eher auf dem Tätigkeitsverlauf und dessen Wechselwirkungen zum Kontext der Nutzung liegt. Die Inspektion wird daher vorzugsweise bei den Testmethoden genannt.

Exemplarische Darstellung des Zusammenspiels verschiedener Erhebungsmethoden

Nachfolgende Schilderung ist ein Beispiel dafür, wie ausgewählte Erhebungsmethoden im Rahmen der sprachkritischen Rekonstruktion in Kombination angewendet werden können. Dafür kombinieren wir eine Dokumentenanalyse mit einem Gruppeninterview und einer teilnehmenden Beobachtung, wie in Abb. 38 (S.205) skizziert.

Hermeneutische Methoden wie beispielsweise die Schriftgutanalyse (Organisationsunterlagen) eignen sich besonders zum Einsatz am Beginn der Analysearbeiten, da dadurch zunächst ein erstes Verständnis für die Organisation als Ganzes und Vorinformation zur Organisationsstruktur und zu den betreffenden Organisationsprozessen erlangt wird.

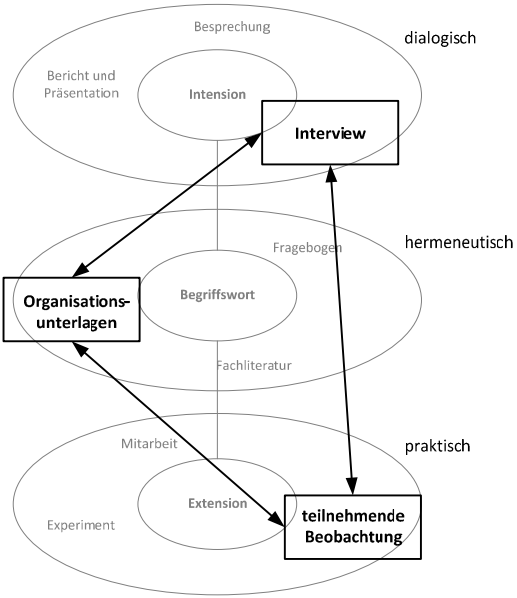

Abb. 38: Auswahl von Methoden zur Wissensrekonstruktion

Im Anschluss daran könnte ein Gruppeninterview mit Fachleuten des Anwendungsbereiches und den Systemherstellern stattfinden. In diesem Gespräch wird die geplante Vorgehensweise von Seiten der Herstellerorganisation erläutert. Die ersten Aussagen aus der Voruntersuchung werden präsentiert und es wird geklärt, was darunter zu verstehen ist. Weiters wird erläutert wie die Ergebnisse dokumentiert werden. Die Aussagensammlung, die mit der Schriftgutanalyse begonnen wurde, wird mit der Dokumentation des Gruppeninterviews fortgesetzt.

Nachdem sichergestellt ist, dass die Vorgehensweise von den Teilnehmern der Anwenderorganisation verstanden wurde, können diese mit Hilfe von Selbstaufschreibungsmethoden und Fragebögen ihre Arbeitsabläufe beschreiben. Nachdem diese Bögen durch die Herstellerseite evaluiert worden sind, kann entweder ein weiteres Interview zur Validierung der gewonnenen Aussagen stattfinden. Handelt es sich um die Rekonstruktion von komplexen Abläufen ist es von Vorteil, zusätzlich eine teilnehmende Beobachtung am Arbeitsplatz der Benutzer durchzuführen, um den Ablauf des zuvor dokumentierten Prozesses vorgeführt zu bekommen. Im Zuge dessen kann der Detaillierungsgrad der Beschreibung adaptiert werden und es können erste Missverständnisse ausgeräumt werden, bevor die Validierung durch ein erneutes Gruppeninterview erfolgt. Interviews bzw. Gruppenbesprechungen oder Workshops sind hilfreich, um im Dialog mit den Benutzern offene Fragen, Missverständnisse und Be-

griffe klären zu können und auf diese Art und Weise die erstellten Aussagen zu überprüfen.

Für alle hier exemplarisch eingesetzten Methoden gilt, dass ihr Einsatz gut vorbereitet sein muss. Beispielsweise können die zu überprüfenden Aussagen oder zu diskutierenden Fragen und Themen bereits vorab den Beteiligten mitgeteilt werden, damit sich alle Beteiligten darauf vorbereiten können. Dies führt dazu, dass die Gruppeninterviews und Workshops gleichermaßen effizient und effektiv sind, was sich wiederum positiv auf die Benutzerbeteiligung und die Systemakzeptanz auswirkt.

5.5 Spezifikation des Ordnungsrahmens zur Methodenbeschreibung

Methodenübersicht

Methodenbezeichnung:

Name der Methode.

Methodentyp:

Sofern dies möglich ist, kann die Methode typisiert werden, z.B. als Erhebungsmethode oder Testmethode.

Verwandte Methoden:

Unter dem Begriff „verwandte Methoden“ können Varianten der beschriebenen Methode oder alternativ einsetzbare Methoden für einen konkreten Zweck in einer Entwicklungsphase angeführt werden.

Methodenkomponenten:

Methoden können elementar sein oder aus zwei oder mehreren anderen Methoden bestehen. Für letzteren Fall sollten hier die einzelnen Methoden (in diesem Fall werden sie als Methodenkomponenten bezeichnet) aufgelistet werden.

Phasenzuordnung:

Methoden können in einer oder mehreren Phasen eines Vorgehensmodells Anwendung finden. Hier werden die in Frage kommenden Phasen aufgelistet.

Zweckeignung:

Die Phasenzuordnung impliziert bereits eine gewisse Zweckeignung. Eine genauere Beschreibung der Zweckeignung der jeweils beschriebenen Methode erfolgt an dieser Stelle. Hier kann auch bewertet werden, für welche Art von Entwicklung (Neuentwicklung oder Re-Engineering) die Methode besonders geeignet ist.

Qualitätsaspekt:

Innerhalb des Qualitätsaspekts können drei Unteraspekte betrachtet werden:

- **Qualitätsziele** – welche Qualitätsziele des Systems können mit dem Methodeneinsatz unterstützt werden? – z.B. Ziele betreffend Anwendbarkeit, Akzeptanz, Lauffähigkeit, Funktionalität, Zuverlässigkeit, Wartbarkeit, Übertragbarkeit, Änderbarkeit, Benutzbarkeit (nach DIN EN ISO 9126-1), softwaretechnische Qualitätsziele, Aufgabenangemessenheit, Effizienz, Effektivität, Gebrauchstauglichkeit – als ganzheitliche Qualitätssicherung mittels Usability Engineering.
- **Qualitätsmerkmale** – Welche Qualitätsmerkmale können durch den Methodeneinsatz beeinflusst werden? – z.B. Merkmale für Akzeptanz gemäß UTAUT-Modell wie Leistungserwartung, Erwartung über den Arbeitsaufwand, soziale Einflüsse, erleichternde Bedingungen.
- **Qualitätsmessung** – Welche Art von Qualitätsmessung kann mit der Methode erfolgen? – z.B. welche Einstufung in ein Qualitätsmodell wie CMMI kann mit der Methode erreicht werden; Hinweis auf Messgrößen bei Testmethoden; Qualitätseinstufung nach dem DATech-Prüfmodell für Usability Engineering.

Normativer Aspekt:

Im normativen Aspekt wird angegeben, welche Regelwerke für die Handhabung der Methode maßgebend sind und zu berücksichtigen sind. Es kann sich dabei um relevante Bestimmungen aus gesetzlichen Regelungen ebenso handeln wie um organisationale Normen und Bestimmungen, die z.B. in einer SOA-Governance festgelegt sind. Ebenso sind zu verwendende Standards wie z.B. Sprachstandards der OMG (etwa aus der UML-Familie), hier aufzuführen, aber auch Regeln, auf die im Umgang mit den Menschen Wert gelegt wird.

Kausaler Aspekt:

Der Kausalitätsaspekt beschreibt Anforderungen und Abhängigkeiten der Methode. Was ist vorher zu tun bzw. was muss vorher geschehen, und was ist nachher zu tun bzw. was kann nachher geschehen – im Sinne von Ursache und Wirkung.

Usability Aspekt:

Die Gebrauchstauglichkeit eines Systems findet bereits beim Qualitätsaspekt entsprechende Berücksichtigung. An dieser Stelle soll noch ergänzt werden, welche Usability-Dimensionen durch den Methodeneinsatz beeinflusst werden können. Die Usability-Dimensionen repräsentieren die Mensch-Computer-Interaktion in drei Dimensionen:

- **Physiologische Dimension** – Einfluss auf Ebene der Wahrnehmung mit den Sinnen.
- **Epistemologische Dimension** – Einfluss auf die Interaktion über die Wissensebene.
- **Psychologische Dimension** – Einfluss auf der Gefühlsebene z.B. durch Spaß oder Angst.

Ressourcenübersicht:

Der Ressourcenaspekt kann mitunter ziemlich umfangreich sein. Um dem Methodenanwender einen raschen Überblick geben zu können, wird an dieser Stelle der Ressourcenaspekt zusammengefasst. Dies kann unter Anführung folgender Punkte passieren:

- **Personen/Rollen** – Anzahl erforderlicher Personen in der jeweiligen Rolle.
- **Know-how** – Besondere Anforderungen hinsichtlich der Qualifikation der Personen.
- **Zeit** – Geschätzter Zeitaufwand in Tagen.
- **Werkzeuge** – Auflistung der erforderlichen Werkzeuge für den Methodeneinsatz.
- **Infrastruktur** – Auflistung der erforderlichen Infrastruktur in Stichworten.

Strukturelle Aspekte

Informationsaspekt:

Der Informationsaspekt beschreibt sowohl die erforderlichen Datensichten, die einer Methode zugrunde liegen. Die Struktur des Informationsflusses. Welcher Input ist notwendig, wie sehen die Ergebnisse aus (Input - Output)? Der Datenfluss wird dem prozessualen Aspekt zugerechnet.

Ressourcenaspekt:

Im Ressourcenaspekt sind alle notwendigen Ressourcen wie Arbeitsmittel, Arbeitsleister, Arbeitsgegenstand, Arbeitsplatz mit den geforderten Eigenschaften anzugeben. Auch die geschätzte erforderliche Zeit für den Methodeneinsatz findet hier Platz, da die Ressourcen für den erforderlichen Zeiteinsatz verfügbar gemacht werden müssen. Dem Charakter nach ist die Zeit dem prozessualen Aspekt zuzurechnen und wird auch dort berücksichtigt. Die dadurch entstehende Vernetzung erfolgt bewusst und fördert das Verständnis der Methodenbeschreibung für den Benutzer.

Der Ressourcenaspekt kann demnach untergliedert werden in:

- **Personen / Rollen** – Welche Personen (Rollen) sind bei Anwendung der Methode beteiligt? Wer führt die Methode aus, wer unterstützt, wer ist beteiligt?
- **Know-how** – Welches spezielle Know-how ist von welchen Beteiligten gefordert. Es sollten hier konkrete Fähigkeiten und Fertigkeiten beschrieben sein und nicht lediglich pauschal auf abstrakte Qualifikationen verwiesen werden.
- **Zeit** – Personen und Arbeitsmittel sowie Infrastruktur sind im Methodeneinsatz für bestimmte Zeit gebunden. Es sollten hier grundsätzliche Hinweise zur benötigten Zeit angeführt werden, die nach einem jeweiligen Methodeneinsatz auch an dieser Stelle durch Erfahrungswerte ergänzt werden. Insbesondere sind Hinweise auf Vorbereitungszeit, Ausführungszeit und Nachbereitungszeit von Nutzen.
- **Arbeitsmittel** – Zu den Arbeitsmitteln zählen einsetzbare Werkzeuge genauso wie Formulare, Checklisten, Fragebogen und Ähnliches. Es sollte bei den Arbeitsmitteln angemerkt werden, welche unbedingt erforderlich sind und auf welche unter Umständen verzichtet werden kann. Auch die zur erforderlichen Infrastruktur gehörenden Verbrauchsmaterialien fallen unter die Arbeitsmittel, wie z.B. Blöcke für Flip-Chart, Filzstifte, Folien.

- **Infrastruktur** – Dazu zählen besondere Raumanforderungen und der Einsatz von bestimmten Medien und Hilfsmitteln wie z.B. Overheadprojektor, Beamer, Flip-Chart, Tonbandgerät, Labor, Testumgebung. Zur Infrastruktur werden jene physischen Hilfsmittel gerechnet, die sich im Einsatz selbst nicht verbrauchen.

Ein weiterer Gliederungspunkt könnte die Anführung der erforderlichen finanziellen Ressourcen betreffen. Zusätzliche Gliederungspunkte können ergänzt werden.

Funktionaler Aspekt:

Im funktionalen Aspekt einer Methodenbeschreibung sind die mit dem Methodeneinsatz verbundenen Aufgaben aufgelistet und beschrieben. Es handelt sich dabei um eine statische Auflistung der zu erledigenden Aufgaben. Die Angabe einer Empfehlung, wer (Rolle) die jeweiligen Aufgaben durchführen sollte, erscheint sinnvoll. In der Aufgabenbeschreibung kann auf grundlegende Methoden wie z.B. Durchführung eines Interviews verwiesen werden, wobei zusätzliche kontextspezifische Hinweise gegeben werden sollten. Auch eine Funktionsstruktur kann hier beschrieben werden.

Prozessuale Aspekte

Operationaler Aspekt:

Der operationale Aspekt spezifiziert die Vorgehensweise der Methodenausführung. Es geht dabei sowohl um eine Beschreibung der Arbeitsvorgänge als auch um die Beschreibung von anzuwendenden Arbeitsverfahren. Welche Arbeitsschritte oder Tätigkeiten sind in welcher Reihenfolge zu tätigen?

Workflow:

In einer Workflowdarstellung für eine Methode können alle relevanten Aspekte zu einem Gesamtbild des Methodeneinsatzes zusammengeführt werden. Es handelt sich dabei vorrangig um die Verbindung des prozessualen mit dem strukturellen Aspekt, zusätzlich können punktuell Bezüge zu relevanten Aspekten aus der Methodenübersicht (Qualitätsaspekt, Normenaspekt, Usability-Aspekt und kausaler Aspekt) über die gesamte Arbeitsfolge hergestellt werden.

Erfahrungswerte

Vorteile:

Positive Erfahrungswerte, die mit Methoden in der Anwendung gemacht wurden, sollten an dieser Stelle eingetragen werden. Wird dies laufend gepflegt, gewinnt die gesamte Methodenbeschreibung an Qualität für künftige Benutzer der Beschreibung.

Risiken:

Sollte sich im Methodeneinsatz herausstellen, dass damit bestimmte Risiken oder Gefahren für das zu entwickelnde System oder für das Entwicklungsprojekt verbunden sind, ist dies unbedingt hier anzumerken und idealerweise auch laufend zu ergänzen.

Weiterführende Literatur:

An dieser Stelle soll verfügbare weiterführende Literatur, die von Mitarbeitern des Unternehmens, in dem die beschriebene Methode einsetzt wird, bereits gelesen wurde und für relevant und gut befunden wurde, angeführt werden. Eine bloße Auflistung von nicht gelesener Literatur ist wenig zielführend.

Glossar

Abstraktion Unter einer Abstraktion verstehen wir „eine Vereinfachung eines realen Systems durch Reduzierung auf solche Teile, die für die jeweilige Fragestellung wesentlich sind“ (Kowalk, 1996, S. 30).

Acceptance Engineering Unter Acceptance Engineering verstehen wir die systematische und ingenieurmäßige Vorgehensweise zur Beschreibung und Beeinflussung von relevanten Aktivitäten und Systemeigenschaften die Akzeptanz betreffend (z.B. über die Akzeptanzfaktoren nach dem UTAUT-Modell). Das Konzept des Acceptance Engineering ermöglicht u.a. die systematische Betrachtung von Verhalten und Emotionen der Benutzer in Bezug auf Anwendungssysteme und ist demnach weiter gefasst als das des Usability Engineering bzw. diesem sogar übergeordnet (vgl. Abschnitt 2.4.1, S. 58ff).

Adoption (adoption) Der Begriff Adoption (lat. „optare“ = wählen, wünschen bzw. „adoptare“ = hinzuerwählen; Duden 7, 2001, S. 22) beschreibt die Kaufentscheidung bezogen auf die Annahme (adoption) oder Ablehnung (rejection), welche in der Regel auf erfolgten Untersuchungen, Erprobungen oder Tests beruht.

Adoptionsforschung (adoption research) Die Adoptionsforschung beschäftigt sich mit der Übernahme von Innovationen durch Individuen und/oder Organisationen und den mit dieser Veränderung verbundenen Konsequenzen. Die Adoption in diesem Sinne ist ein Konzept der Diffusionstheorie *(Diffusionsforschung)*.

Äquipollenz Umfangsgleiche, aber inhaltsverschiedene Begriffe mit unterschiedlichen Bezeichnungen werden als Äquipollenzen bezeichnet. Dieselben Objekte werden aus unterschiedlichen Perspektiven betrachtet.

Beispiel: Einwohner versus Patient.

Behandlung: Begriffssituation muss analysiert und transparent gemacht werden.

Aktivität Unter Aktivität verstehen wir ein Geschehen. Eine Aktivität ist definiert als atomares Konstrukt eines *Prozesses* und kann aus dieser Sicht als Sonderfall eines Prozesses bezeichnet werden (vgl. Abschnitt 2.2.4, S. 26).

Akzeptanz (acceptance) Dynamisches (handlungsbezogenes), wahrnehmbares, aber nicht greifbares Konstrukt (=Phänomen). Sie bringt zum Ausdruck, mit welchem Grad Individuen, Organisationen oder die Gesellschaft schlechthin, etwas (z.B. Innovationen) annehmen, anerkennen oder mit etwas einverstanden sind. In Anlehnung an Kollmann (2006, S. 265-266) wird Akzeptanz als ein dynamisches Konstrukt verstanden, welches in der Anwendungsentwicklung als zentrales Qualitätsziel in Verbindung mit Veränderungen in Organisationen verfolgt wird. Für die Entwicklung von Anwendungssystemen kann das Qualitätsziel Akzeptanz über die Qualitätsmerkmale von Akzeptanz (z.B. Einflussfaktoren nach dem UTAUT-Modell als weitgehend statisches Modell) greifbar werden. Diese Faktoren gilt es bei Bedarf im Entwicklungsprozess systematisch zu beeinflussen. Akzeptanz stellt in Anlehnung an Nielsen (1993, S. 24-25) für die Anwendungsentwicklung den übergeordneten Begriff zu Usability dar.

Anforderungsspezifikation (requirements definition) Der Vorgang, der zu einer widerspruchsfreien Liste von Anforderungen für ein Anwendungssystem führt, wird als Anforderungsspezifikation bezeichnet. Das dabei entstehende Produkt (eben diese Liste) wird ebenfalls als Anforderungsspezifikation bezeichnet. Die Anforderungsspezifikation ist Teil des Requirements Engineering.

Anthropomorpher Agent Ein Agent ist ein Computerprogramm, das Anwender unterstützt und bei der Erledigung von Aufgaben assistiert. Es agiert dabei mehr oder weniger autonom und intelligent. Ein anthropomorpher Agent ist in diesem Kontext in der Lage, menschenähnlich zu agieren. Er kann Emotionen und Befindlichkeiten zeigen und auch so etwas wie einen Charakter darstellen.

Anwendungssystem (application system) Darunter wird ein komplexes System verstanden, welches Hardware, Software (Anwendungssoftware und Basissoftware), Technologieträger und die Einbeziehung der Systemumgebung in Gestalt der Anwender und Anwenderorganisation umfasst.

Anwendungsumgebung Sie umfasst das Umfeld, in dem Software zum Einsatz kommt, z.B. Unternehmen oder andere Organisationsformen.

Aspekt Ein Aspekt ist eine Sichtweise, eine Perspektive, ein Gesichtspunkt, unter dem etwas betrachtet wird. Die Auffassung kommt aus dem Gebiet des Workflow-Managements, wo Aspekte Modellierungsbereiche sind, von denen aus schrittweise ein Gesamtbild (Schema) z.B. einer Workflow-Management-Anwendung entsteht. Analog dazu werden Aspekte für die Modellierung von Methoden verwendet. Es wird gefordert, dass Aspekte (weitgehend) orthogonal im Hinblick auf den sie umgebenden Kontext (andere Aspekte) zu bestimmen sind (Jablonski et al., 1999, S. 485).

Attitude (Einstellung) Kollmann (1999) umschreibt diesen Begriff als innere positive Bereitschaft eines Menschen, ein Produkt zu kaufen, als Beschreibung eines beabsichtigten Verhaltens, als wahrnehmungsorientierte, gefühlsbezogene Antriebskomponente des menschlichen Verhaltens.

Bedeutung „Die Bedeutung eines Wortes ist sein Gebrauch in der Sprache." (Wittgenstein, 1977, S. 311). Die Bedeutung eines Wortes entsteht durch den Gebrauch eines Wortes im sprachlichen und - außersprachlichen Kontext. Unter Gebrauch wird die durch Regeln gesteuerte Verwendung eines Wortes verstanden, die durch die Grammatik festgelegt ist. Danach ist Bedeutung nicht mehr das Korrelat zum Lautbild im sprachlichen Zeichen, das in Bezug zum Gegenstandsbereich steht, sondern eine Größe des Kommunikationsprozesses bzw. der semantischen Situation. Bezugssystem der Sprache ist das menschliche Handeln. Spekulation soll durch genaue Beobachtung ersetzt werden (López, 2008).

Begriff „ Ein Begriff ist eine Funktion, deren Wert immer ein Wahrheitswert ist." (Gottlob Frege, 1848-1925). In der sprachbasierten Informatik werden Begriffe nach Intension und Extension jeweils unterschieden, nach innen und nach außen definiert und abgegrenzt (Abb. 8, S. 33).

Begriffsanomalien siehe Begriffsdefekte.

Begriffsdefekte Ist die Zuordnung von Benennung und/oder Gegenstand zu einem Begriff nicht eindeutig, so liegt ein Begriffsdefekt vor (synonym: *Begriffsanomalien*). Typen von Begriffsdefekten sind *Synonyme, Homonyme, Äquipollenzen*, Vagheiten, falsche Bezeichner (Ortner & Söllner, 1989).

Begriffsklärung Die systematische Erschließung von Begriffen in den möglichen Repräsentationen, z.B. anhand des Schemas der Begriffsrepräsentationen nach Lehmann (1999, S. 136-146), wird als Begriffsklärung bezeichnet (Ortner, 2005, S. 239-240).

Benutzer Ein Benutzer ist für den vorliegenden Kontext eine Person in einer Anwenderorganisation.

Benutzungsschnittstelle (user interface) Eine Benutzungsschnittstelle bilden „Alle Bestandteile eines interaktiven Systems (Software oder Hardware), die Informationen und Steuerelemente zur Verfügung stellen, die für den Benutzer notwendig (!) sind, um eine bestimmte Aufgabe mit dem interaktiven System zu erledigen“ (ISO 9241-110, 2006).

Claim Ein Claim ist eine Forderung im Sinne eines Anspruchs an ein zu entwickelndes System. Claims werden in einer Claims Analysis systematisch auf ihre Relevanz und Wirtschaftlichkeit überprüft.

Claims Analysis Eine Claims Analysis ist eine analytische Methode zur Ermittlung der im Rahmen eines Szenarios auftauchenden Designelemente (Features) mit ihren positiven und negativen Auswirkungen auf die Usability eines Systems. Konkret werden dabei Vor- und Nachteile von Forderungen und Lösungsvorschlägen für ein Anwendungssystem systematisch gegenübergestellt (Rosson & Carroll, 1992, S 77).

Cobit Control Objectives for Information and Related Technology (Cobit) ist ein international anerkanntes Framework zur IT-Governance. Es werden dort Prozesse und Steuerungsvorgaben definiert, wobei nicht das WIE, sondern das WAS im Vordergrund steht. Cobit wird auch als Modell zur Sicherstellung der Einhaltung gesetzlicher Anforderungen (Compliance) eingesetzt und ist Bindeglied zwischen unternehmensweiten Steuerungs-Frameworks (z.B. COSO) und IT-spezifischen Modellen.

Cognitive-Walkthrough Der Cognitive-Walkthrough ist ein expertenbasiertes Verfahren zur Analyse der Gebrauchstauglichkeit von Anwendungssystemen. Mithilfe des Cognitive Walkthrough werden gezielt einzelne Prozesse (oder Abläufe) einer Anwendung untersucht und begutachtet. Die Begutachtung erfolgt im Gegensatz zum *Pluralistic-Walkthrough* durch Experten.

COSO Dies ist die Abkürzung für ein Rahmenmodell zur Internen Steuerung und Kontrolle. Das Framework wurde 2004 um Elemente des Enterprise Risk Managements ergänzt (Helbeck, 2008). Das Modell kommt aus den USA, in Europa gibt es teilweise nationale Governance-Modelle.

Customizing Mit Customizing ist das „Zuschneiden“ von Standard- oder Branchensoftware auf die spezifischen Anforderungen (sprachlich, fachlich, technisch, organisatorisch) der Benutzer bzw. der Anwenderorganisation gemeint. Orientierungsgrundlage ist die ergonomische Effizienz des Gesamtsystems (Ortner, 2005, S. 131).

Diffusion (diffusion) Diffusion is „the process by which an innovation is communicated through certain channels over time among the members of a social system“ (Rogers, 1995, S. 5). Die Diffusion gilt als Akkumulierung einzelner Adoptionen. Sie beschreibt den Prozess von der ersten Adoption bis zur letzten und bringt die Durchdringung (penetration) von etwas, z.B. eines bestimmten Marktes, zum Ausdruck.

Diffusionsforschung (diffusion research) Dieser Forschungsbereich ist ein Teilbereich der Soziologie. Sie untersucht die Regeln, nach denen sich Innovationen in einem sozialen System ausbreiten, und beschreibt Modelle und Funktionen, mit denen dieser Prozess prognostiziert werden kann.

Effektivität (effectivity) Darunter wird die Genauigkeit und Vollständigkeit verstanden, mit der Benutzer ein bestimmtes Ziel erreichen (DIN EN ISO 9241-11, 1999). Das heißt, eine Anwendung ermöglicht es dem Benutzer, sein Ziel überhaupt zu erreichen, unabhängig vom Aufwand, der damit verbunden ist. Die dabei aufzuwendenden Ressourcen werden im Rahmen der „Effizienz" betrachtet.

Effizienz (efficiency) Effizienz zielt darauf ab, dass der Benutzer zur Erreichung eines vorgegebenen Ziels den geringstmöglichen Einsatz benötigt. Der relevante Aufwand kann psychische oder physische Beanspruchung, Zeit, Material oder monetäre Kosten enthalten (DIN EN ISO 9241-11, 1999, S. 8).

Empraktische Erhebungsmethoden Sie zeichnen sich dadurch aus, dass die Gegenstände der Rede vorhanden oder explizit durch nichtsprachliche Handlungen erstellt worden sind. Praxisbegleitende Erhebungstechniken sind z.B. Beobachtung der Mitarbeiter oder Durchführung von Experimenten. Hierbei wird das Wissen praktisch rekonstruiert indem das aktuelle Geschehen (Benutzer bei der Arbeit) im Anwendungsbereich beobachtet und dokumentiert wird. Man spricht in diesem Zusammenhang von empraktischen Erhebungstechniken (Bühler, 1978).

End User Development End User Development versucht die Grenzen, an die die klassische Softwareentwicklung stößt, zu öffnen und von der Benutzerseite alternative Wege zu erschließen. Mit End User Development wird versucht, die klassische User-Entwickler Dichotomie aufzulösen und zwar dieser Art, dass der Benutzer in die Lage versetzt wird, Software selbst zu konfigurieren, adaptieren und auch weiterzuentwickeln. Die aktuellen *Aktivitäten* in dieser Richtung sind (noch) weitgehend technischer Natur. Es gilt aber, diese Entwicklungen zu verfolgen und bei Bedarf eine Annäherung bzw. Integration anzustreben (Liebermann, Paterno & Wulf, 2006).

Enterprise Engineer Darunter wird eine Person verstanden, die methoden- und modellbasiert Rekonstruktionen und Konstruktionen für Unternehmen des Informationszeitalters vornimmt (in Anlehnung an Österle & Winter, 2003, S. 7). Ein Enterprise Engineer benötigt als Profil eine Kombination aus breiter Geschäftserfahrung (Breitenwissen zum Verständnis für den Anwendungsbereich) und umfassendem IT-Know-how (Tiefenwissen mit Spezialisierung), um die geschäftlichen und die technischen Konzepte verstehen zu können, insbesondere zur Nutzung des SOA-Konzeptes und damit auch zur Umsetzung des Enterprise Engineering. Diese Kombination wird als T-shaped bezeichnet.

Enterprise Engineering Mit der Serviceorientierung und den damit verbundenen Architekturen gilt es, nicht nur die Software, sondern auch die Ablauf- und Aufbauorganisation der Anwender(-organisation) sowie weitere Bereiche wie Hardware, Netze und Technologieträger wie Maschinen, Autos oder auch Gebäude mit informatischen Methoden der Modellierung (sprachlich) zu „gestalten". Genau dann, wenn alle diese Bereiche beschrieben d.h. (re-)konstruiert und gleichzeitig verbessert werden, wird von Enterprise Engineering (Ortner, 2008b) geredet.

Epipraktische Erhebungsmethoden Hierbei handelt es sich um Erhebungstechniken, die losgelöst von aktuellen Geschehnissen stattfinden, wie z.B. Interviews, Berichte, Dokumentenanalysen, Besprechungen oder Diskussionen. Das Wissen wird dialogisch rekonstruiert, indem über das Handeln geredet wird. Bühler (1978) spricht in diesem Fall von epipraktischen Erhebungstechniken, bei denen die Gegenstände der Rede in der Erhebungssituation nicht unmittelbar präsent sind.

Epistemisch bedeutet, für die Erkenntnisgewinnung bestimmt.

Erfordernis (implied need) Eine notwendige Voraussetzung, die es ermöglicht, den in einem Sachverhalt des Nutzungskontexts enthaltenen Zweck effizient zu erfüllen (DATech, 2008, S. 199).

Evokatorisches Objekt Ein evokatorisches Objekt ist ein Gegenstand, der Gedanken und Assoziationen hervorruft. Der Computer (das Gerät an sich, aber auch das Gerät in Verbindung mit der Software) als ein evokatorisches Objekt kann bei Benutzern starke emotionale Reaktionen hervorrufen, wie Gefühle, Leidenschaften, Frustrationen aber auch Widerstände, Wünsche und Phantasien (Leithäuser, 1994, S. 76-77).

Extension (extension) Die Extension eines Begriffs setzt sich aus den Objekten, die unter einen Begriff fallen, und dem Begriffsumfang zusammen.

Falsche Bezeichner Abweichung der suggerierten Wortbedeutung von der tatsächlichen Wortbedeutung;

Beispiel: Götter in Weiß versus Ärzte;

Behandlung: Man muss sich entschließen, die zwar vertraute, aber nicht mehr adäquate Bezeichnung eines Begriffes durch eine bessere (hier: „Götter in Weiß“ durch „Ärzte“) zu ersetzen.

Formative Evaluation Bei einer formativen Evaluation handelt es sich im Gegensatz zur summativen Evaluation um eine flexible, wenig komplizierte Form der Bewertung von Ergebnissen während des Entwicklungsprozesses. Ihre Ergebnisse sollten ein direktes Feedback zur Optimierung des betrachteten Entwicklungsteils geben.

Gebrauchssprache (common language) Eine Gebrauchssprache ist eine kommunikative Sprache, in der sich Menschen miteinander unterhalten (z.B. natürliche Sprache, Alltagssprache, Umgangssprache). Fachsprachen mit spezifischem Fachvokabular in Unternehmen sind ebenfalls Gebrauchssprachen (synonym: gemeinsame, kontrollierte Fachsprache).

Gemeinsame (kontrollierte) Fachsprache (common language) siehe *Gebrauchssprache.*

Governance Der Begriff „Governance“ kommt aus dem Französischen und bezeichnet allgemein Steuerungs- und Regelungssysteme von Organisationseinheiten. Mit einem Präfix versehen (z.B. IT-Governance, SOA-Governance) kommt der Gültigkeitsbereich des Regelwerkes (der Governance) zum Ausdruck. Es geht dabei um das Setzen und Einhalten von Verhaltensregeln, die sowohl von außen als auch von innen kommen können, z. B. in Form von anerkannten Standards und Empfehlungen oder selbst entwickelten Leitlinien. Auch Kontrollstrukturen können in der Governance festgelegt sein. Der Grad der Verbindlichkeit einer Governance kann unterschiedlich sein. Übertragen auf ein Unternehmen bezeichnet der Begriff Corporate Governance alle notwendigen Ressourcen, Prozesse und Regeln für eine verantwortungsvolle, nachhaltige und auf langfristige Wertschöpfung ausgerichtete Organisation und Steuerung von Aktivitäten. Dies umschließt alle Menschen, Metriken und Prozesse zur strategischen Planung und Kontrolle (Johannsen & Goeken, 2006, S. 13) und umfasst auch die dazu erforderlichen Systeme.

Homonym Homonyme sind umfangs- und inhaltsverschiedene Begriffe mit einem gemeinsamen Bezeichner.

Beispiel: Beschwerden (körperliche Leiden) versus Klagen, mit denen man sich über etwas beschwert.
Behandlung: Homonyme müssen aufgelöst werden, d.h. auf Bezeichnungsebene unterscheidbar gemacht werden (z.B. durch Verwendung alternativer Bezeichnungen).

Hypothese In einer Hypothese wird eine Behauptung über den Zusammenhang zwischen Qualitäten und Eigenschaften von bestimmten Phänomenen aufgestellt. Es muss in der Hypothese über den Zusammenhang selbst oder die Art der Beziehung etwas ausgesagt werden. Es muss also auch die Art und Weise des beobachtbaren Zusammenhangs angegeben sein. Oder einfacher: Eine Hypothese ist eine Annahme zur Erklärung und Vorhersage von Sachverhalten der externen Welt (Wandmacher, 2002, S. 21).

Ideal Language Philosophy Diese Auffassung geht z. B auf den frühen Wittgenstein sowie Frege G., Russell B. und Carnap R. zurück und erreichte ihren Höhepunkt in den 1950er und 1960er Jahren. Sie ging der Frage nach, wie eine ideale, den höchsten Ansprüchen an Logik und Exaktheit genügende Sprache aussehen müsste (Ernst, 2002, S. 79).

Innovation Der Begriff Innovation wird hier im Sinne von Rogers (1995, S. 5) definiert als Veränderung von Ideen, Arbeitsweisen und Objekten, die nicht tatsächlich neue Erfindungen sein müssen, sondern lediglich von Organisation oder Individuum als neu wahrgenommen werden, demnach also objektiv gesehen nicht mal eine Neuerung darstellen müssten (es handelt sich um subjektive Innovationen).

Integration Unter Integration wird hier allgemein das systematische Zusammenführen von „etwas“ verstanden. Aus Sicht von Technik bzw. Konzeptionen kann es sich dabei um Modelle handeln. Aus Sicht der Sprachwissenschaften handelt es sich um ein konstruktives Arbeiten an einem treffenden Begriff.

Intension Die Intension eines Begriffs gibt dessen spezifische Merkmale an (Begriffsinhalt) und definiert ihn dadurch (Lehmann, 1999, S. 141).

Interaktion Unter Interaktion (Mensch-Maschine-Interaktion) verstehen wir die wechselseitige Beeinflussung von Benutzer und Computer, d.h. die Aufgabenerfüllung durch den Benutzer wird mit Wissen seitens der Software unterstützt, die Aufgabenerfüllung durch die Software wird mit Eingaben (Wissen) seitens des Benutzers unterstützt (Endruweit & Trommsdorff, 2002).

Kognitive Psychologie Die kognitive Psychologie, oder auch Kognitionspsychologie genannt, erforscht die Grundlagen verschiedener geistiger Fähigkeiten wie z.B. das Vorstellungsvermögen, Aufmerksamkeit, Wahrnehmung, Schlussfolgern und Problemlösen. Gegenstand der kognitiven Psychologie sind auch jene psychischen Strukturen und Prozesse, die zwischen der Aufnahme von Information und dem Verhalten vermitteln. Neuere Forschungen zeigen auf, dass auch emotionale und motorische Faktoren zum Verständnis kognitiver Prozesse beitragen. Die kognitive Psychologie ist insbesondere mit den Computer- und Neurowissenschaften interdisziplinär verankert und leistet wichtige Beiträge für andere Forschungsfelder wie z.B. Ergonomie und Didaktik (Universität Bern, 2009).

Logische Propädeutik Dies ist die Lehre von den Grundformen und Grundbausteinen der wissenschaftlichen Aussage. Sie geht zurück auf Kamlah & Lorenzen (1975).

Mapping Allgemein kann Mapping als Metapher für die Zusammenführung bzw. Zuordnung von zwei oder mehreren Bereichen, Gegenständen oder Ähnlichem verstanden werden. Es geht dabei darum, Analogien (auf sprachlicher oder nicht sprachlicher Basis) zwischen den betrachteten Gegenstandsbereichen zu identifizieren und auf dieser Grundlage ein gegenseitiges Verständnis der Bereiche herbeizuführen bzw. diese nach Möglichkeit zu einer Einheit zu verbinden. Dies kann gelingen, wenn die Übereinstimmungen entsprechend groß sind.

Materialsprachlichkeit Unter Materialsprachlichkeit wird generell die Berücksichtigung begrifflicher Festlegungen im Anwendungsbereich verstanden. Dies wird durch Entwicklung und Verwendung von Inhaltsstandards und Fachwörterbüchern auf inhaltlicher Ebene durch die Festlegung der Semantik, d.h. der *Bedeutung* der verwendeten Fachsprache, umgesetzt (Ortner, 2005, S. 32 u. 252). Konkret handelt es sich um rekonstruierte und normierte Anwenderfachbegriffe aus dem Anwendungsbereich als inhaltlicher Teil der Sprache (z.B. Lexika) in Verbindung mit Grammatik als Formalteil der Sprache (z.B. Satzbausystem), welche für die jeweilige Entwicklung im Anwendungsbereich von Bedeutung sind. Das Adjektiv „material“ könnte auch durch „inhaltlich“ ersetzt werden (Ortner, 2005, S. 250). Eine materiale Sprache hat (im Gegensatz zu Programmiersprachen) immer einen Bezug zum Anwendungsbereich.

Mentales Modell „Ein mentales Modell ist ein geistiges Gebilde, das die wesentlichen Merkmale des Sachverhalts zutreffend repräsentiert. Ein gutes mentales Modell ist beweglich; man kann in der Vorstellung quasi bestimmte Eingaben variieren und sich dann ausdenken, wie sie sich auswirken. Man spielt im Geiste verschiedene Zustände des Modells durch. Damit kann man nicht nur Neues verstehen, sonder auch Vorhersagen treffen. Je besser das mentale Modell ist, desto höher ist die Chance, dass die Vorhersage auch eintritt. Das unterscheidet Experten von Laien.“ (Weidenmann, 1999, S. 42)

Methode Eine Methode kann universell als eine Sprache in Verbindung mit einer Vorgehensweise für den Spracheinsatz definiert werden (Ortner, 2005, S. 230-231). Dass dies als Menge von Regeln zur Erreichung eines bestimmten Ziels verstanden wird (Zelewski, 1999, S 34; Balzert, 2000; S. 36), stellt eine anwendungsorientierte Auffassung von Methoden i.w.S. von Ortner dar. Eine implementierte Methode wird Werkzeug genannt.

Methodenneutralität heißt im Zusammenhang mit der sprachkritischen Systementwicklung, dass sich Entwickler und Anwender der normalen Sprache (Gebrauchssprache) bedienen, um einen Fachentwurf zu erarbeiten. Die eingesetzte *Normsprache* stellt weder terminologisch noch strukturell (grammatisch) eine spezielle Entwurfssprache (gekennzeichnet durch eine bestimmte Methode sowie Vorgehensweise) in den Vordergrund. Es soll eine Gebrauchssprache zur Erarbeitung des Fachentwurfs benutzt werden, die für alle Beteiligten (Entwickler, Anwender, Planer) verständlich ist. Erst nachdem alle Begriffe und Aussagen geklärt sind, sollten spezifische Sprachen wie beispielsweise Diagrammsprachen zur Systemmodellierung verwendet werden. Außerdem sollte die Beschreibung des Anwendungsgebietes und seiner fachlichen Zusammenhänge neutral gegenüber einem softwaretechnischen Lösungsparadigma sein, damit der Entwurfsprozess nicht frühzeitig von Anforderungen der Systementwickler eingeengt wird (Schienmann, 1997). Ein weiterer Vorteil dieses Vorgehens ist, dass der Entwurfsprozess wiederverwendbar ist, auch bei einer Änderung des Anwendungssystemtyps.

Methodologie Eine Methodologie (z.B. ProCEM®) ist als universeller Rahmen zu verstehen**,** d.h. der Rahmen umfasst neben Vorgehensweisen eine spezielle Methodenlehre für Anwendungssysteme (Ortner, 2005, S. 183).

Modell Ein Modell ist „eine Zusammenfassung von Merkmalen eines realen (oder empirischen) künstlichen Systems sowie eine Festlegung der Beziehungen zwischen diesen Merkmalen; da ein Modell niemals alle Merkmale eines Systems umfassen kann, ist ein Modell eine Abstraktion eines realen *Systems*." (Wedekind et al., 1998, S. 266, mit Bezug auf Kowalk, 1996).

Multimodalität Unter Multimodalität versteht man eine Mensch-Computer-Interaktion, bei der verschiedene Sinnesmodalitäten (Sinneskanäle) angesprochen werden, wie z.B. Hörsinn (akustisch), Sehsinn (optisch), Tastsinn (haptisch, taktil).

Multiplizität „Multiplizität" ist ein Begriff aus der Informatik, insbesondere aus dem Systementwurf. Sie gibt an, wie viele DINGE konkret an einer BEZIEHUNG beteiligt sein können bzw. allgemein, wo die obere und die untere Schranke für die Anzahl der beteiligten DINGE ist (Fowler & Scott, 2000, S. 46).

Normsprachlichkeit Nachdem ein Terminus einer bestimmten Verwendung rekonstruiert wurde, steht es nicht mehr im Belieben der Sprachteilnehmer (Anwender, Rechnersystem), diesen Terminus den Gegenständen (willkürlich) zu- oder abzusprechen (Ortner, 2005). Normsprachlichkeit bedeutet demnach, dass die verwendete Terminologie rekonstruiert und eine normierte, vereinfachte Syntax für die Strukturierung der Aussagen verwendet wird. So können die in Zusammenarbeit von Entwicklern und Anwendern erfassten entwicklungsrelevanten Aussagen normiert werden und später in adäquate Diagramm- oder Spezifikationssprachen überführt werden (Lehmann, 1999).

Nutzungsanforderung (usage requirement) Eine Nutzungsanforderung beschreibt eine erforderliche Benutzeraktion an einem interaktiven System und zwar in einer Art, mit der Tätigkeiten beschrieben werden, und nicht in technisch realisierter Weise. Beispiel: „Der Benutzer muss lesen können." und nicht „Die Beleuchtungs-stärke muss einstellbar sein." Nutzungsanforderungen beruhen auf Erfordernissen des Nutzungskontexts. Wenn gesicherte ergonomische Erkenntnisse für die Merkmale eines Produkts (z. B. Farbkodierung, Hintergrund einer Anzeige, Zeichengröße, Kontrast) gegeben sind, können die Eigenschaften dieser Merkmale selbst als Anforderung dienen. In der Regel werden Nutzungsanforderungen jedoch nicht als Anforderungen an Merkmale, sondern als Anforderungen an Tätigkeiten formuliert (DATech, 2008, S. 203-204).

Ontologie (ontology) Aus der übergeordneten Definition als Lehre vom Sein, von den Ordnungs-, Begriffs- und Wesensbestimmungen des Seienden, sind Ontologien im vorliegenden Fall (Sprach-)Modelle von Anwendungsbereichen, d.h. systematische Wissensrepräsentationen (Schemata) eines Anwendungsbereichs in Gestalt von Sprache und Sprachartefakten. Die Bestandteile einer solchen systematischen Darstellung einer gemeinsamen Fachsprache (common language) des Anwendungsbereichs beruhen auf geklärten Begriffen. Davenport & Prusak (1999) bezeichnen eine Ontologie im Sinne einer Gebrauchssprache (common language) als „working knowledge". Zweck ist die Darstellung und Formalisierung von Wissen, wodurch Mehrdeutigkeit vermieden werden soll. Eine Ontologie ist nicht nur eine geordnete Begriffssammlung, sondern Darstellung der „Welt" des Anwendungsbereiches.

Ordinary Language Philosophy Diese Richtung der Philosophie beschäftigt sich mit der Alltagssprache (Umgangssprache) und ihrer Tauglichkeit für philosophische Untersuchungen. Die Alltagssprache oder Umgangssprache wird dabei nicht im Sinne einer sprachlichen Varietät verstanden, sondern als reale Ausprägung einer natürlichen Sprache im Gegensatz zu einer konstruierten. Zu ihren Vertretern gehörten u.a. der späte Wittgenstein, Austin und Searle (Ernst, 2002, S. 79).

Orthogonalität Ein System besteht aus orthogonalen Komponenten, wenn diese unabhängig von dem sie umgebenden Kontext (anderen Komponenten) sind. Dabei können Grade der Unabhängigkeit unterschieden werden. Als Kriterien können z.B. die Austauschbarkeit oder die Kombinierbarkeit von Komponenten gewählt werden (Jablonski et al., 1999, S. 488).

Orthosprache Ist ein Terminus der konstruktiven Wissenschaftstheorie. Jedes Wort, das zum Bestand einer Fachsprache gehört, wird zunächst in seiner Bedeutung explizit geklärt und eindeutig festgelegt (rekonstruiert) (Ortner, 1993, S. 32).

Paradigma Ein Paradigma ist im ursprünglichen, epistemologischen Sinne eine Weltanschauung. Etwas abgeschwächt und im Bedeutungsraum eingeschränkt ist in dieser Hinsicht die Verwendung des Begriffes in verschiedenen Wissenschaftsbereichen, so auch in der Software-Entwicklung. Der Begriff „Paradigma" steht dort für eine besondere, fokussierte Sichtweise auf einen (möglichst grundlegenden) *Aspekt* des jeweiligen Fachgebietes. So wird beispielsweise vom Paradigma der „Wiederverwendbarkeit von Software", vom Paradigma des schlanken Managements (lean management), von der ganzheitlichen Anwendungssystementwicklung oder vom sprachkritischen Entwicklungsparadigma gesprochen. Es besteht eine gewisse *Vagheit* des Begriffs.

Perceived ease of use (wahrgenommene einfache Nutzung) ist “the degree to which a person believes that using a particular system would be free from effort” (Davis, 1989, S. 320). Wahrgenommene einfache Nutzung bezieht sich auf einen Rahmen innerhalb dessen eine Person davon ausgeht, dass die Nutzung eines Systems keiner mentalen Anstrengung bedarf. Oft wird dies auch als Leichtigkeit der Nutzung bezeichnet, d.h. es wird subjektiv wahrgenommen, dass die Nutzung von etwas „leicht“ im Sinne von „nicht schwierig“ ist und somit keiner großen Anstrengungen bedarf.

Perceived usefulness (wahrgenommene Zweckmäßigkeit) ist "the degree to which a person believes that using a particular system would enhance his or her job performance" (Davis, 1989, S. 320). Wahrgenommene Zweckmäßigkeit ist definiert als ein Spielraum, den eine Person wahrnimmt, die davon ausgeht, dass ein Anwendungssystem ihre eigene Leistungsfähigkeit im Anwendungsbereich verbessert. Oft wird dies auch als wahrgenommene Brauchbarkeit bzw. Nützlichkeit bezeichnet, d.h. es wird wahrgenommen, dass etwas imstande (geeignet, fähig, tauglich) ist, zweckmäßig, vorteilhaft und nützlich eingesetzt zu werden.

Persona Personas sind (Rollen-)Modelle aus dem Bereich der Mensch-Computer-Interaktion. Es handelt sich dabei um eine Gestaltung von hypothetischen Archetypen zur Unterstützung der Entwicklungsarbeit. Personas sind sehr spezifisch dargestellt (Name, Bild, Beruf, Hobbys usw.) und durch ihre Einstellungen, Verhaltensweisen und Ziele definiert. Sie sind an sich fiktiv, werden jedoch auf Basis realer Benutzer entwickelt. Personas helfen potenzielle Zielgruppen (Benutzer) zu erkennen und zu beschreiben (Nieters, Ivaturi & Ahmed, 2007).

Phonem Ein Phonem wird als kleinste bedeutungsunterscheidende Einheit der gesprochenen Sprache definiert (Adjektiv: phonematisch).

Pluralistic Walkthrough Der Pluralistic Walkthrough ist ein Verfahren zur Feststellung der Benutzerfreundlichkeit einer Benutzeroberfläche und wird vor allem in den frühen Entwicklungsstadien eingesetzt.

Programmieren im Großen (programming in the large) Durch Komponentenorientierung soll ein Programm in mehrere leicht handhabbare Teile aufgespalten werden, welche über genau definierte Schnittstellen zusammenarbeiten (Zuser et al., 2004, S. 62). Dabei gilt es, die Programmteile bzw. Komponenten oder Services in einer „Sprache“ zu beschreiben, die später relativ einfach in eine Programmiersprache überführt werden kann. So kann z.B. BPMN als Sprache zur Prozessbeschreibung (programming language) verwendet werden. Die Beschreibungen können relativ einfach in BPEL (execution language) übersetzt werden und auf dieser Basis ausgeführt werden.

Die Prozessbeschreibung findet in der Regel auf der Auftraggeberseite (Kunde, Anwenderorganisation) statt. Abgeleitet aus den obigen Ausführungen werden die Arbeiten von Grobanalyse und Detailanalyse auf Seiten des Auftraggebers auch unter dem Begriff „Entwicklung im Großen“ zusammengefasst.

Proposition Unter Proposition versteht man in der Linguistik den Inhalt eines Satzes. Übertragen auf die Anwendungssystementwicklung mit der Grundlegung der sprachkritischen Vorgehensweise handelt es sich also um das, was in einem geäußerten Satz oder in einer aufgeschlagenen Benutzeroberfläche am Computer, in einem bestimmten Kontext über bestimmte Gegenstände (auch Sachverhalte lassen sich auf Gegenstände reduzieren) des Anwendungsbereichs ausgedrückt wird.

Prozess (process) Ein Prozess ist der gerichtete Ablauf eines Geschehens. Ein Prozess kann aus einer oder mehreren Aktivitäten bestehen. Das Geschehen kann atomar sein (Prozess = *Aktivität*) oder aus mehreren Prozessen oder Aktivitäten bestehen. Die Anordnung von realen Aktivitäten bzw. Prozessen ergibt die zeitliche Abfolge von (wieder-)eintretenden Ausprägungen von Zustandstypen wie Prozessen und Aktivitäten. Dementsprechend sind sowohl Prozesse als auch Aktivitäten als Schemata zu betrachten und ihrerseits wiederum eine Ausprägung eines Zustandstyps.

Indem ein Prozess auf ein vereinbartes Schema gebracht wurde, ist er verfügbar wie ein verwendbares Gerät (Ortner & Eller, 2008).

Es kann unterschieden werden zwischen ergonomischen Prozessen und technischen Prozessen. Die Grundlage für ergonomische Prozesse (Arbeitsvorgänge, Verrichtungen) bildet menschliches Handeln. Ein isolierter Prozess resultiert dabei aus einer Folge von menschlichen Handlungen (Tätigkeiten) eines Arbeitsträgers (Stelleninhabers, Aufgabenträger) mit Arbeitsmitteln (Sachmitteln) an einem Arbeitsobjekt (Gegenstand). Menschliches Handeln basiert auf Tendenzgesetzen (Normen, Regeln), die von Aufgabenträgern annähernd erfüllt werden. Ergonomische Prozesse sollten als Zweck-Mittel-Ketten rekonstruiert werden.

Technischen Prozessen liegen Verfahren zugrunde, die auf Ursache-Wirkungsketten basieren. Die Grundlage für die Spezifikation technischer Prozesse bilden Natur- oder Verlaufsgesetze (Jablonski et al., 1999, S. 488).

Rationale Grammatik Eine Rationale Grammatik ist ein Universalschema, das wegen seiner Allgemeingültigkeit viel einfacher ist, als empirische Grammatiken. Eine Rationale Grammatik ist Grundlage für alle Sprachen, die ein Rechner versteht. Sprachen auf Basis einer Rationalen Grammatik nennt man rationale Sprachen. Menschen können sich in einer rationalen Sprache nicht unterhalten, dazu ist sie viel zu primitiv, sie können aber mit Hilfe einer rationalen Sprache eine methodische Rekonstruktion unserer natürlichen Sprache vornehmen. Man spricht auch von einer Schematisierung (Wedekind et al., 2004b, S. 265-266).

Rekonstruktion (reconstruction) „Rekonstruktion“ bedeutet die vernünftige, systematische (methodische) Erarbeitung (Wiedererarbeitung) von Wissen in Form einer Sammlung relevanter Aussagen zur Beschreibung eines Anwendungsgebietes in normierter Form, auf Basis normalsprachlicher Aussagen der Anwender. Die Rekonstruktion geht weit über den „Nachbau“ im eigentlichen Sinne des Wortes hinaus, da sie auch noch nicht realisierte Ansprüche der Anwender und Innovationen für den Anwendungsbereich integriert.

Reliabilität (reliability) Reliabilität stellt neben der Validität und der Objektivität ein Gütekriterium für empirische Untersuchungen dar. Reliabilität ist ein Maß für die Replizierbarkeit von Untersuchungsergebnissen unter gleichen Bedingungen. Sie ist dann gegeben, wenn z.B. ein „Test“ das, was er misst, auch zuverlässig misst (in Anlehnung an von Rosenstiel, 2007, S. 161).

Repository-System Ein Repository-System ist ein Dokumentationssystem, welches sowohl auf Metaebene (Metadaten-Repository) als auch auf Schemaebene Systembeschreibungen in Form von Sprachen, Strukturen, Verfahren, Objekten usw. und deren zeitliche Entwicklung, aber auch Beschreibungen der relevanten Systemumwelt enthält.

Saussure’sches Zeichenmodell Nach diesem Modell besteht ein sprachliches Zeichen aus zwei Seiten, die sich gegenseitig hervorrufen.

Schema Ein Schema ist eine universelle Beschreibung von Gegenständen unserer Welt. Ausprägungen sind demgegenüber singuläre Beschreibungen. Das Begriffspaar Schema-Ausprägung spielt in der Informatik eine zentrale Rolle (vgl. Abschnitt 2.2.3, S. 24ff; z.B. Wedekind et al., 2004b).

Simulation Eine Simulation ist die numerisch approximative Behandlung von Modellen, deren mathematische Darstellung nicht zu geschlossenen Lösungen führt (Kowalk, 1996, S. 31 u. 35).

Software Engineering „The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is the application of engineering to software." (IEEE, 1990, S. 67) Im deutschen Sprachgebrauch wird in diesem Zusammenhang vorwiegend der Begriff bzw. Überbegriff *Software-Technik* verwendet.

Software-Technik Neben den bereits von der IEEE angeführten Aspekten zum *Software Engineering* stehen in der Softwaretechnik folgende Dinge im Zentrum der Betrachtung (Heinrich et al., 2004, S. 26): „Handlungen, die durch Software unterstützt, automatisiert oder rationalisiert werden können; Problemlösungswissen […], das in Form von Daten und Programmen dargestellt werden kann; Gestaltung der Interaktion zwischen Menschen und programmgesteuerten Maschinen (→ *Benutzerschnittstelle*)."

Sprachartefakt Ein Sprachartefakt ist das Ergebnis einer Zusammensetzung von Grammatik, Lexikon und einem kategorialen Ansatz auf Ausprägungsebene. Sprachartefakte sind als Ergebnisse von Sprachhandlungen zu verstehen, die den Charakter von *Sprachspielen* bzw. Sprechakten haben. Schemata von Sprachartefakten sind Sprachen (vgl. Abschnitt 2.2.2, S. 23-24). Synonym: Sprachprodukt.

Sprachspiel Ein Sprachspiel ist eine „natürliche" Funktionseinheit der Sprache (Wittgenstein). Sprachspiele sind durch Verben benannt (Ernst, 2002, S. 90).

Synonym Ein Synonym ist eine Benennung, die denselben Begriff bezeichnet wie eine andere Benennung (DIN 2342-1, 2004). Es sind Umfangs- und inhaltsgleiche Begriffe, deren verschiedene Bezeichner gegeneinander ausgetauscht werden können.

Beispiel: Krankenhaus versus Spital versus Klinik.
Behandlung: Synonyme Bezeichnungen sind zulässig, wenn sie bekannt sind und kontrolliert werden können (z.B. systeminterne, automatische Übersetzung in die vorab festgelegte Vorzugsbezeichnung).

System Als System gilt „eine räumlich abgeschlossene, logisch zusammengehörende und zeitlich begrenzte Einheit, die voneinander abhängige Komponenten umfasst (Kowalk, 1996, S. 27).

Szenario Ein Szenario ist die Beschreibung von existierenden oder geplanten Systemen aus der Perspektive eines oder mehrerer Benutzer. Es werden damit Ziele bzw. Ergebnisse, Absichten und Reaktionen der Benutzer in Bezug auf das System und die damit verbundene Aufgabenstellung beschrieben und dokumentiert. Die Szenarios dienen dazu, Anforderungen für die Gebrauchstauglichkeit des Systems zu klären und nach Rosson & Carroll (2002) den iterativen Entwicklungsprozess zu steuern. Sie sind in natürlicher Sprache formuliert und haben dadurch den Vorteil, dass sie sowohl von den Benutzern als auch von den Entwicklern verstanden werden können.

Terminologie Nach DIN 2342-1 (2004) ist eine Terminologie die Gesamtheit der Begriffe und Benennungen in einem Fachgebiet (Fachwortschatz).

Transdisziplinarität Als Terminus der neueren Wissenschaftstheorie dient der Begriff der Transdisziplinarität zur Charakterisierung von Forschungsformen, die problembezogen über die, historisch bedingt divergente, fachliche und disziplinäre Konstitution der Wissenschaft hinausgehen, vor allem hinein in die soziale Wirklichkeit, wie es Nicolini (2001) zum Ausdruck bringt. Im Gegensatz zur Interdisziplinarität hält die Transdisziplinarität nicht an den Fachgrenzen oder Grenzen zwischen Disziplinen fest, und es sollte damit keine Gefahr drohen, dass Grenzen von Fächern und Disziplinen zu Erkenntnisgrenzen werden. Ortner konkretisiert dies und meint mit Transdisziplinarität, die sich einstellen kann, dass interdisziplinäre Kooperationen mit der Zeit zu einer anderen, die fachlichen und disziplinären Orientierungen der einzelnen Fachgebiete verändernden, neuen methodischen und inhaltlichen Orientierung in einem gemeinsamen Forschungsgebiet der Disziplinen führen können. Transdisziplinarität wird dort wirksam, wo alleine fachliche oder disziplinäre Definitionen von Problemlagen und Lösungsansätzen nicht möglich sind. Sie ist daher als ein integratives (zusammenführendes), aber nicht generalisierendes (überwissenschaftliches) Konzept zu verstehen (Ortner, 2005, S. 52-53).

Triangulation Unter Triangulation versteht man eine anwendungsorientierte Kombination mehrerer Datenquellen bzw. im übertragenen Sinn eine Kombination mehrerer Methoden.

Usability Engineering Unter Usability Engineering wird die systematische und ingenieurmäßige Vorgehensweise zur Beschreibung und Schaffung von Systemeigenschaften die Usability in all ihren Dimensionen verstanden. Das Usability Engineering ist ein Teilbereich des Acceptance Engineering.

Vagheit Als Vagheit wird ein Begriffsdefekt bezeichnet, bei dem die *Intension* (Inhalt) eines Begriffes nicht eindeutig definiert ist. Dadurch kann es zu Unsicherheiten und Unklarheiten bei der Zuordnung von Gegenständen zu dem Begriff (*Extension* = Umfang) kommen. Beispiel: „Berg" ist ein Begriff mit einer gewissen Vagheit, diese könnte ausgeräumt werden, indem die Intension präzisiert wird, damit der Begriffsumfang scharf getrennt werden kann. Dies kann beispielsweise durch Angabe einer Höhe geschehen, ab welcher von einem Berg gesprochen werden kann. Es gibt eine Vielzahl von Typen und Graden von Vagheit. In der Regel ist eine Vagheit nicht intersubjektiv festzustellen, sondern muss im Sinne einer pragmatischen Interpretation einer Informationssituation beurteilt werden. Diese Beurteilung dient dann als Basis zur Beseitigung der Vagheit im Zuge der Begriffsklärung im Rekonstruktionsprozess (Ortner, 2005, S. 68; Lehmann, 1999, S. 160-161).

Beispiel: Arzt versus Mediziner (Unklarheit hinsichtlich der Frage der Berufsberechtigung).
Behandlung: Klärung und Präzisierung des Inhaltes führt zu scharfer Trennung des Umfangs (z.B. Bindung der Berufsberechtigung an Zulassung durch Ärztekammer).

Validität ist dann gegeben, wenn ein Untersuchungsgegenstand (Objekt, Test, Verfahren, Modell, Methode, Anforderung…) das, was er auszusagen vorgibt, auch tatsächlich aussagt (in Anlehnung an von Rosenstiel, 2007, S. 161).

Verstehen Das Verstehen von etwas ist dem Aufbau einer mentalen Repräsentation (*mentales Modell*) dessen, was verstanden werden soll, gleichzusetzen.

Vorgangstyp Ein Vorgangstyp bezeichnet einen Abschnitt im Multipfad-Vorgehensmodell und wird synonym zum Begriff „*Phase*" verwendet.

Wissen Wissen als Teil der Praxis muss sozial verstanden werden – Wissen produziert und reproduziert soziale Prozesse und Strukturen und ist gleichzeitig deren Produkt (Cockburn, 2003, S. 313). Die Auffassung von Wissen als Schema ist für die sprachbasierte Informatik grundlegend. Das Wissen eines Menschen sind seine Schemata im Gehirn, die seinen Handlungsrahmen bestimmen.

Workflow Ein Workflow ist eine zum Teil automatisiert (algorithmisch) – von einem *Workflow-Management-System* gesteuert – ablaufende Gesamtheit von Aktivitäten, die sich auf Teile eines Geschäftsprozesses oder andere organisationale Vorgänge bezieht. Ein Workflow besteht aus Abschnitten (Subworkflows), die weiter zerlegt werden können. Er hat einen definierten Anfang, einen organisierten Ablauf und ein definiertes Ende. Ein Workflow-Management-System steuert die Ausführung eines Workflows. Workflows sind überwiegend als ergonomische (mit Menschen als Aufgabenträgern) und nicht als technische (z.B. Einsatz von Maschinen) Prozesse zu sehen (Jablonski et al., 1999, S. 490).

Workflow-Management Management umfasst nach allgemeiner Auffassung Handlungen wie Organisieren, Planen, Entscheiden, Kontrollieren, Steuern und Führen. Die Ausübung dieser Handlungen im Zusammenhang mit Arbeitsvorgängen wird beim Einsatz eines Workflow-Management-Systems „Workflow-Management“ genannt. Dabei ist die Frage, inwieweit Handlungen wie Organisieren, Planen, Entscheiden, Kontrolliere, Steuern oder Führen durch ein (technisches) System ausgeübt oder unterstützt werden können, durch den Entwicklungsstand von Workflow-Management-Systemen und Workflow-Management-Anwendungen bestimmt (Jablonski et al., 1999, S. 491).

Zirkelfreies Vorgehen Zirkelfreies Vorgehen heißt, bei Erstellung einer Definition darauf zu achten, dass nicht Begriffe zur Erklärung verwendet werden, die selbst noch nicht definiert wurden.

Zufriedenheit *(satisfaction)* Neben Effizienz und Effektivität ist Zufriedenheit der dritte Usability-Bestandteil nach DIN EN ISO 9241-11 (1999) und gleichzeitig die am schwersten fassbare Größe. Maße der Zufriedenheit beschreiben die Beeinträchtigungsfreiheit und die Akzeptanz der Nutzung. Diese können sich auf Einstellungen beziehen, ein Produkt zu benutzen, oder auf das Benutzerurteil über Aspekte wie Effizienz, Nützlichkeit oder Lernförderlichkeit (DIN EN ISO 9241-11, 1999). Zufriedenheit entsteht, wenn die Erwartungen des Benutzers (der Nutzer) mindestens erfüllt oder sogar übertroffen werden. Um Zufriedenheit erreichen zu können, müssen die Erwartungen des Benutzers (Nutzers) bekannt sein (Beier, 2002, S. 3).

Literaturverzeichnis

Abran, A. & Moore, J. W. (2004). SWEBOK® - Guide to the Software Engineering Body of Knowledge. Letzter Zugriff am 31.05.2007 unter http://www.swebok.org/ironman/pdf/SWEBOK_-Guide_2004.pdf.

Acker, H. B. (1966). *Organisationsanalyse* (3. Aufl.). Baden-Baden: Gehlen.

Agile Alliance (2008). 10 Key Principles of Agile Software Development. Letzter Zugriff am 14.06.2008 unter http://www.agile-software-development.com/2007/02/10-things-you-need-to-know-about-agile.html.

Aikio, K. (2006). Integrating usability engineering with software engineering: a preliminary view on aspects surrounding the topic of usability integration. *Proceedings of the 29th Information systems research seminar in Scandinavia (IRIS29)* (S. 1-13).

Ajzen, I. (1991). The Theory of Planned Behavior. *Organizational Behavior and Human Decision Processes 50(2)*, 179-211.

Ajzen, I. & Fishbein, M. (1980). *Understanding attitudes and predicting social behaviour.* Englewood Cliffs: Prentice-Hall.

Alexander, C. (2002). *The Phenomenon of Life: Book 1, The nature of order: An essay on the art of building and the nature of the universe.* Oxford: Center for Environmental Structure.

Andoh-Baidoo, F. K., White, E. F. R. & Kasper, G. M. (2004). Information Systems' Cumulative Research Tradition: A Review of Research Activities and Outputs Using Pro forma Abstracts. *Proceedings of AMCIS 2004* (S. 4195-4202).

Apel, K. (1976). *Transformation der Philosophie. Sprachkritik, Semiotik, Hermeneutik* (Bd. 1). Frankfurt a. Main: Suhrkamp.

April, A. & Abran, A. (2008). *Software Maintenance Management: Evaluation and Continuous Improvement.* Hoboken: Wiley-IEEE Computer Society Press.

Arntz, R., Picht H. & Mayer, F. (2004). *Einführung in die Terminologiearbeit - Studien zu Sprache und Technik* (5. Aufl.). Hildesheim: Olms.

Artim, J. et al. (1998). Incorporationg work, process and task analysis into commercial and industrial object-oriented systems development. *SIGCHI Bulletin 30(4)*, 33-36.

Auramäki, E., Lehtinen, E. & Lyytinen, K.(1988). A Speech-Act-Based Office Modeling Approach. *ACM Transactions on Information Systems 6(2),* 11-23.

Austin, J. L. (1979). *Zur Theorie der Sprechakte (How to Do Things with Words)*. Stuttgart: Reclam.

Bagozzi, R. P., Davis, F. D. & Warshaw, P. R. (1992). Development and test of a theory of technological learning and usage. *Human Relations, 45(7)*, 660-686.

Balzert, H. (2000). *Lehrbuch der Software-Technik. Software Entwicklung* (2. Aufl.). Heidelberg: Spektrum.

Bandura, A. (1986). *Social Foundations of Thought and Action: A Social Cognitive Theory*. Englewood Cliffs: Prentice Hall.

Bartlett, F. C. (1932). *Remembering: A Study in experimental and social psychology.* Cambridge: Cambridge University Press.

Bartsch, C. (2001). Die Verständlichkeit von Software-Hilfesystemen. Eine sprachwissenschaftliche Analyse am Beispiel von Microsoft Word 2000. In J. Hennig & M. Tjarks-Sobhani (Hrsg.), *Tekom Hochschulschriften (6).* Lübeck: Schmidt-Römhild.

Bass, L., Clements, P. & Kazman, R. (2003). Software Architecture in Practice (2. Aufl.). *SEI Series in Software Engineering.* Amsterdam: Addison-Wesley.

Battle, L. & Lockheed, M. (2005). Patterns of Integration: Bridging User Centered Design into the Software Development Lifecycle. In A. Seffah, J. Gulliksen & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 287-308). Dordrecht: Springer.

Baum, M. (1978). Extension/Intension. In E. Braun & H. Radermacher (Hrsg.), *Wissenschaftstheoretisches Lexikon* (S. 189-191). Graz: Styria.

Beck, K. (2003). *Extreme Programming: Die revolutionäre Methode für Softwareentwicklung in kleinen Teams. Das Manifest.* München: Addison-Wesley.

Becker, J. & Pfeiffer, D. (2006). Beziehungen zwischen behavioristischer und konstruktionsorientierter Forschung in der Wirtschaftsinformatik. In S. Zelewski & N. Akca (Hrsg.), *Fortschritt in den Wirtschaftswissenschaften* (S. 1-17). Wiesbaden: Deutscher Universitätsverlag.

Beier, M. & von Gizyki, V. (2002). *Usability. Nutzerfreundliches Web-Design.* Berlin: Springer.

Bell, G. C. (2001). A Personal Digital Store. *Communications of the ACM 44(1),* 86-91.

Bell, G. C. & Gray, J. (1997). The Revolution Yet to Happen. In P. J. Denning & R. M. Metcalfe (Hrsg.), *Beyond Calculation* (S. 5-32). New York: Springer.

Bell, G. C., Gemmel, J. & Lueder, R. (2006). MyLifeBits: A Personal Database for Everything. *Communications of the ACM 49(1),* 88-95.

Bender, R. (2002). Analyse und Bewertung des V-Modell 97 hinsichtlich seiner Eignung für komponentenbasierte Softwareentwicklung. Diplomarbeit, Technische Universität München.

Benyon, D. & Macaulay, C. (2002). Scenarios and the HCI-SE design problem. *Interacting with Computers 14(4)*, 397-405.

Berr, M. (1990). *Technik und Körper.* Berlin: Dietrich Reimer.

Beuscher, B. (1994). *Schnittstelle Mensch: Menschen und Computer – Erfahrungen zwischen Technologie und Anthropologie.* Heidelberg: Asanger.

Beyer, H. & Holtzblatt, K. (1998). *Contextual Design - Defining customer-centered systems.* San Francisco: Morgan Kaufmann.

Bieberstein, N., Laird, R. G., Jones, K. & Mitra, T. (2008). *Executing SOA. A practical Guide for the Service-Oriented Architect.* Upper Saddle River, NJ: IBM Press, Pearson.

Biere, B. U. (1989). *Verständlich-Machen. Hermeneutische Tradition – historische Praxis – sprachtheoretische Begründung.* Tübingen: Niemeyer.

Bischofberger, W. R. (1990). Prototyping-Oriented Incremental Software Development – Paradigms, Methods, Tools and Implications. Dissertation, Johannes Keppler Universität, Linz.

Bleicher, K. (2004). *Das Konzept Integriertes Management: Visionen – Missionen – Programme* (7. überarb. u. aktual. Aufl.). Frankfurt a. Main: Campus.

Blomkvist, S. (2005). Towards a model for bridging agile Development and User-Centered Design. In A. Seffah, J. Gulliksen & M. C. Desmariais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 219-244). Dordrecht: Springer.

Bloomberg, J. (2007). The Value of SOA Governance. zapthink. Letzter Zugriff am 10.04.2008 unter http://whitepaper.pcwelt.de/index.cfm?pid=1&pk=1905.

Boas, F. (1911a). „Introduction". In F. Boas (Hrsg.), *Handbook of American Indian Languages* (S. 1-83). Washington: Government Printing Office.

Boas, F. (1911b). *The Mind of Primitive Man*. New York: MacMillan.

Boehm, B. W. (1988). A Spiral Model of Software Development and Enhancement. *IEEE Computer 21(5)*, 61-72.

Boehm, B. (1991). Software risk management: Principles and Practice, *IEEE Software 8(1)*, 32-41.

Boose, J. H. (1993). A Survey of Knowledge Acquisition Techniques and Tools. In B. G. Buchanan & D. S. Wilkins (Hrsg.), *Readings in Knowledge Acquisition and Learning* (S. 39-56). San Mateo: Morgan Kaufmann.

Bora, A. (1997). Sachhaltigkeit versus Verfahren. Einige methodologische Konsequenzen konstruktivistischer Wissenschaftssoziologie. In T. Suter (Hrsg.), *Beobachtung verstehen, Verstehen beobachten. Perspektiven einer konstruktivistischen Hermeneutik* (S. 228-252). Opladen: Verlag für Sozialwissenschaften.

Bradshaw, J. M. (1997). *Software Agents*. Cambridge, MA: MIT Press.

Bremer, G. (1998). Genealogie von Entwicklungsschemata. In R. Kneuper, G. Müller-Luschnat & A. Oberweis (Hrsg.), *Vorgehensmodelle für die betriebliche Anwendungsentwicklung* (S. 32-59). Stuttgart: Teubner.

Brinker, K., Antos, G., Heinemann, W. & Sager, S. F. (2000). *Text- und Gesprächslinguistik - Ein internationales Handbuch zeitgenössischer Forschung.* Berlin: Walter de Gruyter.

Britzelmaier, B. (1999). *Informationsverarbeitungs-Controlling - Ein datenorientierter Ansatz*. Stuttgart: Teubner.

Britzelmaier, B. & Eller, B. (2004). Aspekte einer Dynamisierung der Lebenszykluskostenrechnung. *Controller Magazin, 29(6)*, 527-532.

Brockhaus (2006). *Brockhaus-Enzyklopädie* (Bd. 1, 21. völlig neu bearb. Aufl.). Leipzig: Brockhaus.

Bruder, R. & Schiele, B. (2007). Intelligente Schnittstellen. *Thema Forschung (1),* 40-43.

Bultmann, R. (1988). Das Problem der Hermeneutik. In J. Schreiter (Hrsg.), *Hermeneutik – Wahrheit und Verstehen* (S. 373-400). Berlin: Akademie-Verlag.

Bunge, M. (1967). *Scientific Research*. New York: Springer.

Busch, M. W. & Von der Oelsnitz, D. (2008). Multiskilling als Ansatzpunkt kompetenzerweiternder Mitarbeiterqualifikation. In A. Eisenkopf, C. Opitz & H. Proff (Hrsg.), *Strategisches Kompetenz-Management in der Betriebswirtschaftslehre - Eine Standortbestimmung* (S. 47-70). Wiesbaden: Gabler.

Butler, K. A. (1996), Usability engineering turns 10. *Interactions 3(1),* 58-75.

Bühler, K. (1978). *Sprachtheorie - Die Darstellungsfunktion der Sprache.* Frankfurt: Ullstein.

Bürg, O. & Mandl, H. (2004). *Akzeptanz von E-Learning in Unternehmen.* Forschungsbericht Nr. 167. München: LMU, Department Psychologie, Institut für Pädagogische Psychologie.

Carter, S. (2007). *The New Language of Business. SOA & Web 2.0.* Upper Saddle River, NJ: IBM Press, Pearson.

Cajander, A., Gulliksen, J. & Boivie, I. (2006). Management Perspectives on Usability in a Public Authority – A Case Study. *Proceedings of the 4th Nordic conference on Human-Computer Interaction* (pp. 38-47). Oslo: ACM.

Charlton, M. (2004). Entwicklungspsychologische Grundlagen. In R. Mangold, P. Vorderer & G. Bente (Hrsg.), *Lehrbuch der Medienpsychologie* (S. 130-150). Göttingen: Hofgrefe.

Carnap, R. (1962). *Logical Foundations of Probability* (2. Aufl.). Chicago: Chicago University Press.

Carnap, R. (1976). *Einführung in die Philosophie der Naturwissenschaft* (3. Aufl.). München: Nymphenburger.

Carnap, R. (1998). *Der logische Aufbau der Welt.* Hamburg: Meiner.

Carroll, J. M. (2000). *Making use: Scenario-based design of human-computer interactions.* Cambridge: MIT Press.

Carroll, J. M. (2003) (Ed.). *HCI Models, Theories and Frameworks. Toward a Multidisciplinary Science.* San Francisco: Elsevier Science.

Chau, P. Y. K. (1996). An empirical assessment of a modified technology acceptance model. *Journal of Management Information Systems 13(2),* 185-204.

Christmann, U. & Groeben, N. (1999). Psychologie des Lesens. In B. Franzmann, K. Hasemann, D. Löffler & E. Schön (Hrsg.), *Handbuch Lesen* (S. 145-223). München: Saur.

Claessens, S. (2006). Corporate Governance and Development. *WB Research Observer 21(1)*, 91-122.

Claparède, E. (1932). La dècouverte de l'hypothèse. *Journal de psychologie normale et pathologieque (29)*, 648-656.

CMMI (2006). CMMI® for Development (Version 1.2). Pittsburgh: Carnegie Mellon Software Engineering Institute.

Cockburn, A. (1997). Structuring use cases with goals. *Journal of Object-Oriented Programming* 09-10/97 u. 11-12/97.

Cockburn, A. (2003). *Agile Software-Entwicklung.* Bonn: mitp.

Codd, E. F. (1990). *The Relational Model for Database Management* (Version 2). München: Addison-Wesley.

Compeau, D. R. & Higgins, C. A. (1995a). Application of Social Cognitive Theory to Training for Computer Skills. *Information Systems Research 6(2),* 118-143.

Compeau, D. R. & Higgins, C. A. (1995b). Computer Self-Efficiacy: Development of a measure and Initial Test. *MIS Quarterly 13(3),* 319-339.

Constantine, L. L. & Lockwood, L. A. D. (1999). *Software for use: A Practical Guide to the Models and Methods of usage-Centered Design*. Massachusetts: Addison-Wesley.

Connell, J. L. & Shafer, L. B. (1989). *Structured Rapid Prototyping. An Evolutionary Approach to Software Development.* Prentice Hall: Yourdon.

Cooper, A. & Reimann, R. (2000). *About Face 2.0: The Essentials of User Interface Design.* Hoboken, NJ: Wiley.

Corsten, H. & Reiß, M. (Hrsg.). (1999). *Betriebswirtschaftslehre* (3. Aufl.). München: Oldenbourg.

Cossey, G. R. (1987). *Prototyper. Users Guide*. Portland: SmethersBarnes.

Dahm, M. (2006). *Grundlagen der Mensch-Computer-Interaktion*. München: Pearson.

Daimler Benz (1998). E-Book of Software Ergonomic Knowledge. Letzter Zugriff am 06.08.2008 unter http://web.inf.tu_dresden.de/ST2/pw/lehre/lv_bdt/hyperbase/buch/vorgehensweise/vorgehensweise_grundidee.htm.

Daimler Chrysler (2003). *E-Book of Software Ergonomic Knowledge*. Ulm: Daimler Chrysler Forschungsteam UCDP.

da Silva, P. P. & Paton, N. W. (2001). A UML-based design environment for interactive applications. *Proceedings of the Second International Workshop on User Interface to Data Intensive Systems* (UIDIS '01) (S. 60-71), IEEE Computer Society.

DATech (2008). *Leitfaden Usability. Gestaltungsrahmen für den Usability Engineering Prozess* (Version 1.1). Frankfurt a. Main: Deutsche Akkreditierungsstelle Technik. Letzter Zugriff am 10.09.2008 unter http://www.datech.de/share/files/Leitfaden-Usability.pdf.

Davenport, T. H. & Prusak, L. (1999). *Wenn Ihr Unternehmen wüßte, was es alles weiß…*. Landsberg a. Lech: moderne industrie.

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use and User Acceptance on Information Technology. *MIS Quarterly 13(3)*, 319-340.

Davis, F. D., Bagozzi, R. P. & Warshaw, P. R. (1989). User acceptance of computer technology: A comparison of two theoretical models. *Management Science 35*, 982-1003.

Davis, F. D., Bagozzi, R. P. & Warshaw, P. R. (1992). Extrinsic and Intrinsic Motivation to Use Computers in the Workplace. *Journal of Applied Social Psychology 22(14)*, 1111-1132.

Davis, F. D. & Venkatesh, V. (2000). A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies. *Management Science 46(2)*, 186-204.

de Antonellis, V. & Demo, B. (1983). Requirements Collection and Analysis. In S. Ceri (Hrsg.), *Methodology and Tools for Data Base Design* (S 9-24). Amsterdam: Elsevier Science.

de Beaugrande, R. A. & Dressler, W. U. (2001). *Einführung in die Textlinguistik* (3. überarb. Aufl.) Tübingen: Uni-Taschenbücher.

DeMarco, T. (1978). *Structured Analysis and System Specification*. Englewood Cliffs: Yourdon Press.

DeRemer, F. & Kron, H. (1975). Programming-in-the large versus programming-in-the-small. *Proceedings of the international conference on Reliable software* (S. 114-121). New York: ACM.

Dethloff, C. (2004). *Akzeptanz und Nicht-Akzeptanz von technischen Produktinnovationen*. Lengerich: Papst.

DIN 2330 (1993). *Begriffe und Benennungen: Allgemeine Grundsätze.* Normenausschuss Terminologie (NAT) im DIN, Deutsches Institut für Normung e.V. Berlin: Beuth.

DIN 2342-1 (2004). *Begriffe der Terminologielehre, Grundbegriffe*. Deutsches Institut für Normung e.V.. Berlin: Beuth.

DIN EN ISO 9001 (2000). *Qualitätsmanagementsysteme – Anforderungen*. Genf: ISO.

DIN EN ISO 9241-1 (2002). *Ergonomie der Mensch-System-Interaktion. Teil 1: Allgemeine Einführung.* Genf: ISO.

DIN EN ISO 9241-2 (1993). *Ergonomie der Mensch-System-Interaktion. Teil 2: Anforderungen an die Arbeitsaufgaben*. Genf: ISO.

DIN EN ISO 9241-11 (1999). *Ergonomie der Mensch-System-Interaktion. Teil 11: Anforderungen an die Gebrauchstauglichkeit.* Genf: ISO.

DIN EN ISO 9241-110 (2006). *Ergonomie der Mensch-System-Interaktion. Teil 110: Grundsätze der Dialoggestaltung.* Genf: ISO.

DIN EN ISO 13407 (1999). *Gestaltung von benutzerorientierten interaktiven Systemen.* Genf: ISO.

Dion, F. (2003). What is the CMMI? Letzter Zugriff am 30.04.2006 unter http://www.dovico.com/-documents/What-is-the-CMMI-process-improvement-v1-00-20030730.pdf.

Djajadinigrat, J. P., Overbeeke, C. J. & Wensveen, S. A. G. (2000). Augmenting Fun and Beauty: A Pamphlet. *Proceedings of DARE 2000: Designing Augmented Reality Environments* (S. 131-134). Elsinore: ACM.

Donahue, G. M. (2001). Usability and the bottom line. *IEEE Software 18(1)*, 31-37.

Donahue, G. M., Weinschenk, S. & Nowicki, J. (1999). Usability Is Good Business. Letzter Zugriff am 22.07.2007 unter http://interface.free.fr/Archives/Usability_Is_Good_Business.pdf.

Dray, S. M. & Mrazek, D. (1996). A Day in the Life of a Familiy: An International Ethnographic Study. In D. R. Wixon & J. Ramey (Hrsg.), *Field Methods Casebook for Software Design.* New York: John Wiley & Sons.

DSDM (2008). *DSDM Consortium*. Letzter Zugriff am 01.08.2008 unter www.dsdm.org.

Duden 5 (2007). *Das Fremdwörterbuch* (Bd. 5, 9. aktual. Aufl.). Mannheim: Dudenverlag.

Duden 7 (2001). *Herkunftswörterbuch. Etymologie der deutschen Sprache* (Bd. 7, 3. erw. u. völlig neu bearb. Aufl.). Mannheim: Dudenverlag.

Dumke, R. (2003). *Software Engineering* (4. überarb. u. erw. Aufl.). Wiesbaden: Vieweg & Sohn.

Duncker, K. (1945). On problem-solving. In J. F., Dashiell (Hrsg.), *Psychological Monographs of the American Psychological Association (58)*. Washington: APA.

Dzida, W. & Konradt, U. (Hrsg.). (1995). *Psychologie des Software-Entwurfs*. Göttingen: Verlag für Angewandte Psychologie.

Dzida, W., Herda, S. & Itzfeld, W. D. (1978). User-perceived quality of interactive systems. *Proceedings of the 3rd international conference on Software engineering* (S.188-195). Atlanta: IEEE Press.

Dzida, W., Hofmann, B., Freitag, R., Redtenbacher, W., Baggen, R., Geis, T., Beimel, J., Zurheiden, C., Hampe-Neteler, W., Hartwig, R. & Peters, H. (2001). *Gebrauchstauglichkeit von Software. ErgoNorm: Ein Verfahren zur Konformitätsprüfung von Software auf der Grundlage von DIN EN ISO 9241 Teile 10 und 11*. Bremerhaven: Wirtschaftsverlag NW.

Earthy, J. V. (1999). Trump Version. Usability Maturity Model: Processes. Version 2.2. Letzter Zugriff am 25.11.2008 unter http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/-Usability-Maturity-Model%5B2%5D.PDF.

Easterbrook, S., Singer, J., Storey, M. & Damian, D. (2008). Selecting Empirical methods for Software Engineering Research. In F. Shull, J. Singer & D. I. K. Sjoberg (Hrsg.), *Guide to Advanced empirical Software Engineering* (S. 285-311). London: Springer.

EC Wise Inc. (2003). Integrating User Centered Design in Agile Development processes. Letzter Zugriff am 25.11.2008 unter http://www.ecwise.com/content/whitepapers/ECWise_IntegratingAgile-Design.pdf.

Eisenkopf, A., Opitz, C. & Proff, H. (Hrsg.). (2008). *Strategisches Kompetenz-Management in der Betriebswirtschaftslehre - Eine Standortbestimmung*. Wiesbaden: Gabler.

Ehn, P. (1988). *Work-Oriented Design of Computer Artifacts*. Philadelphia: Lawrence Erlbaum Association.

Eller, B. (2004). Rekonstruktion einer Patientenakte für den Leistungserstellungsverbund des österreichischen Gesundheitswesens am Beispiel der Modellregion Vorarlberg. Master Thesis, Hochschule Liechtenstein.

Eller, B. (2008a). Usability Engineering zur Förderung der Nachhaltigkeit in der Anwendungssystementwicklung. In M. Bichler et al. (Hrsg.), *Multikonferenz Wirtschaftsinformatik 2008* (S. 1481-1492). Berlin: GITO.

Eller, B. (2008b). *Life long learning meets knowledge usability and knowledge understandability*. Beitrag zur LLL08, Workshop “Perspektiven des lebenslangen Lernens - dynamische Bildungsnetzwerke, Geschäftsmodelle, Trends.“ Universität Hannover, April 2008.

Endruweit, G. & Trommsdorff, G. (Hrsg.). (2002). *Wörterbuch der Soziologie* (2. verb. und erw. Aufl.). Stuttgart: UTB.

Erl, T. (2008). *SOA - Entwurfsprinzipien für serviceorientierte Architektur*. München: Addison-Wesley.

Ernst, P. (2002). *Pragmalinguistik. Grundlagen – Anwendungen – Probleme*. Berlin: Walter de Gruyter.

Esposito, A., Faundez-Zany, M., Keller, E. & Marinaro, M. (Hrsg.). (2007). Verbal and Nonverbal Communication Behaviours. COST Action 2102 International Workshop. LNCS 4775. Berlin. Springer.

Felber H. & Budin G. (1989). *Terminologe in Theorie und Praxis.* Tübingen: Gunter Narr.

Ferstl, O. K. & Sinz, E. J. (1990). Objektmodellierung betrieblicher Informationssysteme im Semantischen Objektmodell (SOM). *Wirtschaftsinformatik 32(6),* 566-581.

Ferstl, O. K. & Sinz, E. J. (2006). *Grundlagen der Wirtschaftsinformatik* (5. überarb. u. erw. Aufl.). München: Oldenbourg.

Fettke, P., Intorsureanu, I. & Loos, P. (2002). Komponentenorientierte Vorgehensmodelle im Vergleich. In K. Turowski (Hrsg.), *4. Workshop komponentenorientierte betriebliche Anwendungssysteme* (WKBA) (S. 19-43). Letzter Zugriff am 01.08.2008 unter http://archiv.tu-chemnitz.de/pub/2002/0075.

Fettke, P. & Loos, P., (2006). *Reference Modelling for Business Systems Analysis.* Hershey, USA: Idea Group.

Fielenbach, K. & Niehaves, B. (2008). Theories of Language in IS-Research – A Review. In J. Becker, H. Krcmar, B. Niehaves, B. (Hrsg.), *Wissenschaftstheorie und gestaltungsorientierte Wirtschaftsinformatik,* Multifkonferenz Wirtschaftsinformatik München, 26.02.2008, Arbeitsberichte des Instituts für Wirtschaftsinformatik, Westfälische Wilhelms-Universität Münster, Münster, Arbeitsbericht Nr. 120, 2008, S. 87-110.

Fink, K. (2000). *Know-how-Management. Architektur für den Know-how-Transfer*. München: Oldenbourg.

Fischer, E. (2006). Grundlagen eines Formalisierungsframeworks für Vorgehensmodelle. In C. Hochberger, R.Liskowsky (Hrsg), *Informatik für Menschen* (Bd 1, S. 654-660), GI Lecture Notes in Informatics (LNI), Proceedings, Informatik 2006.

Fishbein, M. & Ajzen, I. (1975). Belief, Attitude, Intention and Behavior: *An Introduction to Theory and Research.* Reading: Addison-Wesley.

Floyd, C. (1984). A Systematic Look at Prototyping. In R. Budde, K. Kuhlenkamp, L. Matthiassen & H. Züllighoven (Hrsg.), *Approaches to Prototyping* (S. 1-18). New York: Springer.

Foster, R. (1982). A call for vision in managing technology. *McKinsey Quarterly*, Summer 1982, 26-36.

Fowler, M. & Scott, K. (2000). *UML konzentriert* (2. aktual. Aufl.). München: Addison-Wesley.

Frank, U. (1994). *Multiperspektivische Unternehmensmodellierung. Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung.* München: Oldenbourg.

Frank, U. (1997). Erfahrung, Erkenntnis und Wirklichkeitsgestaltung – Anmerkungen zur Rolle der Empirie in der Wirtschaftsinformatik. In O. Grün & L. J. Heinrich (Hrsg.), *Wirtschaftsinformatik – Ergebnisse empirischer Forschung.* Heidelberg: Springer.

Frank, U. (2003). Für Sie gelesen: IS Research Relevance Revisited: Subtle Accomplishment, Unfulfilled Promise or Serial Hypocrisy? *Wirtschaftsinformatik 45(3),* 354-357.

Franke, H.J. (1975). Methodische Schritte beim Klären konstruktiver Aufgabenstellungen. *Konstruktion 27*, 395-402.

Frege, G. (1961). Die Grundlagen der Arithmetik [1884]. Letzter Zugriff am 23.09.2008 unter http://www.ac-nancy-metz.fr/enseign/philo/textesph/Frege.pdf.

Frege, G. (2002). *Funktion – Begriff – Bedeutung*. M. Textor (Hrsg.). Göttingen: Vandenhoeck & Ruprecht.

Freiling, J., Gersch, M. & Goeke, C. (2008). Die kompetenztheoretische Erklärung von Unternehmungen anhand des Organisationalen Ambientes. In A. Eisenkopf, C. Opitz & H. Proff (Hrsg.), *Strategisches Kompetenz-Management in der Betriebswirtschaftslehre - Eine Standortbestimmung* (S. 3-12). Wiesbaden: Gabler.

French, J. R. P., Rodgers, W. & Cobb, S. (1974). Adjustment as person-environment fit. In G. V. Coelho, D. A. Hamburg & J. E. Adams (Hrsg.), *Coping and adaption* (S. 316-333). New York: Springer.

Friedrich, J., Hammerschall, U., Kuhrmann, M. & Sihling, M. (2008). *Das V-Modell XT (Informatik Im Fokus)*. Berlin: Springer.

Friedrichs, J. (1990). *Methoden empirischer Sozialforschung* (14. Aufl.). Opladen: Verlag für Sozialwissenschaften.

Fröschle, H. & Strahringer, S. (Hrsg.). (2006). IT-Governance. *Praxis der Wirtschaftsinformatik*. Heidelberg: dpunkt.

Früh, W. (2007). *Inhaltsanalyse* (6. überarb. Aufl.). Konstanz: UVK.

Gallivan, M. J. (2001). Oganizational adoption and assimilation of complex technological innovations: Development and Application of a new Framework. *Database for Advances in Information Systems 32(3)*, 51-85.

Garret, J. J. (2003). *The Elements of User Experience – User-Centered Design for the Web.* New York: American Institute of Graphic Arts.

Gartner (2005). Hype Cycle for Application Development 2005, Gartner, Inc., Letzter Zugriff am 30.09.2005 unter http://www.gartner.com/DisplayDocument?doc_cd=127755.

Gaver, W. W. & Martin, H. (2000). Alternatives. Exploring Information Appliances through Conceptual Design Proposals. *Proceedings of the CHI 2000 Conference on Human Factors in Computing (*S. 209-216).

Gebert, D. (1974). *Organisationsentwicklung. Probleme des geplanten organisatorischen Wandels.* Stuttgart: Schäffer-Poeschel.

Gethmann, C. F. (1987). Vom Bewußtsein zum Handeln. Pragmatische Tendenzen in der deutschen Philosophie der ersten Jahrzehnte des 20. Jahrhunderts. In H. Stachowiak (Hrsg.), *Pragmatik. Handbuch des Pragmatischen Denkens* (Bd. 2, S. 202-232). Hamburg: Felix Meiner.

Ghani, H., Koll, C., Kunz, C., Sahin, T. & Yalcin, A. (2007). *Konzept zum Hochschulwettbewerb ‚Beste Prozessarchitektur'*, BITKOM University Challenge.

Gieryn, T. F. (1999). *Cultural Boundaries - Credibility in the Line.* Chicago: University of Chicago Press.

Goodhue, D. L. (1995). Understanding User Evaluations of Information Systems, *Management Science 41(12)*, 1827-1844.

Goodhue, D. L. & Thompson, R. L. (1995). Task-Technology Fit and Individual Performance, *MIS Quarterly 19(2)*, 213-236.

Gorguen, J. A. & Linde, C. (1993). Techniques for Requirements Elicitation. *Proceedings International Symposium of Requirements Engineering* (S. 152-164).

Graeser, A. (2002). *Positionen der Gegenwartsphilosophie: vom Pragmatismus bis zur Postmoderne.* München: Beck.

Grice, H. P. (1993). Logik und Konversation. In G. Meggle (Hrsg.), *Kommunikation und Bedeutung* (S. 243-265). Frankfurt: Suhrkamp.

Grochla, E. (1982). *Grundlagen der organisatorischen Gestaltung.* Stuttgart: Poeschel.

Grochla, E. & Meller, F. (1974). *Datenverarbeitung in der Unternehmung 1: Grundlagen.* Reinbek: Rohwolt.

Grudin, J. (1991). Systematic Sources of Suboptimal Interface Design in Large Product Development Organizations. *Human Computer Interaction 6*, 147-196.

Grundmann, M. (2005). Prozessmodellintegration am Beispiel einer RUP-Erweiterung durch das V_Modell XT. Diplomarbeit, Hochschule Reutlingen, Fachbereich Informatik, Letzter Zugriff am 01.10.2008 unter ftp://ftp.uni-kl.de/pub/v-modell-xt/Release-1.1/Infomaterial/Prozessmodell-integration.pdf.

Guida, G. & Tasso, C. (1994). *Design and Development of Knowledge Based Systems.* Chichester: Wiley.

Gulliksen, J., Lantz, A. & Boivie, I. (1998). User-centered design in practice – problems and possibilities. *SIGCHI Bulletin 31(2),* 25-35. Letzter Zugriff am 29.04.2008 unter www.nada.kth.se/cid/-pdf/cid_40.pdf.

Gulliksen, J., Lantz, A. & Boivie, I. (2001). How to make user centred design usable. *SIGCHI Bulletin 33(3)*, 16-31. Letzter Zugriff am 29.4.2008 unter http://cid.nada.kth.se/pdf/cid_72.pdf.

Gulliksen, J., Göransson, B., Boivie, I., Persson, J., Blomkvist, S. & Cajander, A. (2005). Key Principles for User-Centered Systems Design. In A. Seffah, J. Gulliksen & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 17-36). Dordrecht: Springer.

Gundelsweiler, F., Memmel, T. & Reiterer, H. (2004). Agile Usability Engineering. In R. Keil-Slawik, H. Selke & G. Szwillus (Hrsg.), *Mensch & Computer 2004: Allgegenwärtige Interaktion (*S. 33-42). München: Oldenbourg. Letzter Zugriff am 28.07.2007 unter http://hci.uni-konstanz.de/-downloads/fg_tm_hr_mc_2004.pdf.

Hacker, W. (1987). Software-Ergonomie: Gestalten rechnergestützer geistiger Arbeit?! In W. Schönpflug & M. Wittstock (Hrsg.), *Software Ergonomie 87: Nützen Informationssysteme dem Benutzer?* (S. 31-54). Stuttgart: Teubner.

Härder, T. & Rahm, E. (1999). *Datenbanksysteme - Konzepte und Techniken der Implementierung.* Berlin: Springer.

Hamel, G. & Välikangas, L. (2003). The Quest for Resilience. *Harvard Business Review 81(9)*, 52-63.

Hansen, H. R. & Neumann, G. (2005). *Wirtschaftsinformatik I: Grundlagen und Anwendungen* (9. Aufl.). Stuttgart: Lucius & Lucius.

Harris, D. (Hrsg.). (2007). Engineering Psychology and Cognitive Ergongomics. *Proceedings of 7th International Conference, EPCE 2007*, LNCS. Berlin: Springer.

Harrison, D. A., Mykytyn, P. P. & Riemenschneider, C. K. (1997). Executive Decisions about Adoption of Information Technology in Small Business: Theory and Empirical Tests. *Information Systems Research, 8(2)*, 171-195.

Harrison, S., Back, M. & Tatar, D. (2006). "It's Just a Method!": A Pedagogical Experiment in Interdisciplinary Design. *Proceedings of the 6th ACM conference on Designing Interactive Systems* (S. 261-270). New York: ACM Press.

Hartmann, D. (2005). Willensfreiheit und die Autonomie der Kulturwissenschaften. Letzter Zugriff am 07.08.2008 unter http://www.jp.philo.at/texte/HartmannD1.pdf.

Hartmann, E. A. (2005). Arbeitssysteme und Arbeitsprozesse. In E. Ulich (Hrsg.), *Mensch, Technik, Organisation*. Zürich: vdf Hochschulverlag.

Hartmann, T. (2004). Computervermittelte Kommunikation. In R. Mangold, P. Vorderer & G. Bente (Hrsg.), *Lehrbuch der Medienpsychologie* (S. 674-693). Göttingen: Hofgrefe.

Hartwig, R. (2005). Usability Standards, Gesetze und Entwicklungsprozesse. *IM Information Management & Consulting 20(3)*, 64-67.

Hartson, H. R., Castillo, J. C., Kelso, J. & Neale, W. C. (1996). Remote Evaluation: The network as an Extension of the usability laboratory. *Proceedings of CHI 96 conference* (S. 228-235).

Hassenzahl, M., Beu, A. & Burmester, M. (2001). Focus Usability: Engineering Joy. *IEEE Software 1(1)*, 2-8.

Hassenzahl, M., Burmester, M. & Beu, A. (2001). Engineering Joy. *IEEE Software,* January/February, 70-76.

Hassenzahl, M. & Hofvenschiöld, E. (2003). „If it doen't feel right, who cares if it works?" oder Muss Software mehr als nur gebrauchstauglich sein? *Proceedings of the 1st annual German Chapter der Usability Professionals Association - Track* (S. 135-139). Stuttgart.

Hausendorf, H. (1997). Konstruktivistische Rekonstruktion. Theoretische und empirische Implikationen aus konversationsanalytischer Sicht. In T. Suter (Hrsg.), *Beobachtung verstehen, Verstehen beobachten. Perspektiven einer konstruktivistischen Hermeneutik* (S. 254-272). Opladen: Verlag für Sozialwissenschaften.

Heberle, S. A. (2002). Förderung eines benutzerzentrierten Software-Entwicklungsprozesses in Unternehmen am Beispiel von UBS Schweiz. Master Thesis, Uni Konstanz.

Heinecke, A. M. (2004). *Mensch-Computer-Interaktion*. München: Hanser.

Heinemann, E. (2006). *Sprachlogische Aspekte rekonstruierten Denkens, Redens und Handelns.* Wiesbaden: Deutscher Universitätsverlag.

Heinemann, E. & Ortner, E. (2004). DarWin: A Project for language-based Schema and Knowledge Management. *Proceedings der IKE'04, International Conference on Information and Knowledge Engineering* (S. 442-446).

Heinemann, E., Ortner, E. & Sternhuber, J. (2004). Sprachbasierte Wissensrekonstruktion am Beispiel des Einkommensteuergesetzes. *Proceedings der Tagung MobIS im Rahmen der MKWI 2004* (S. 91-111).

Heinemann, E., Ortner, E. & Wedekind, H. (2004). A Framework for Language-Based Schema Management and Epistemic Application Systems. *Proceedings I-KNOW '04, International Conference on Knowledge Management*, Graz.

Heinrich, L. J. (1995). State of the Art und Editorial zum Schwerpunktthema – Ergebnisse empirischer Forschung. *Wirtschaftsinformatik 37(1),* 3-9.

Heinrich, L. (2002). *Informationsmanagement* (7. vollst. überarb. u. erg. Aufl.). München: Oldenbourg.

Heinrich, L. J., Heinzl, A. & Roithmayr, F. (2004). *Wirtschaftsinformatik-Lexikon* (7. vollst. überarb. u. erw. Aufl.). München: Oldenbourg.

Heinrich, L. J. & Wiesinger, I. (1997). Zur Verbreitung empirischer Forschung in der Wirtschaftsinformatik. In O. Grün & L. J. Heinrich (Hrsg.), *Wirtschaftsinformatik – Ergebnisse empirischer Forschung* (S. 37-49). Wien: Springer.

Heinsen, S. & Vogt, P. (Hrsg.) (2003). *Usability praktisch umsetzen. Handbuch für Software, Web, Mobile Devices und andere interaktive Produkte*. München: Hanser.

Helbeck, J. C. (2008). *Internal Control System in der Praxis – ein Umsetzungsleitfaden zur Steuerung operationeller Risiken in Geschäftsprozessen*. Saarbrücken: VDM.

Henne, H. (1975). Sprachpragmatik. Nachschrift einer Vorlesung. *Germanistische Linguistik 3(8)*, Tübingen: Niemeyer.

Henne, H. & Rehbock, H. (2001). *Einführung in die Gesprächsanalyse* (4. durchges. u. bibl. erg. Aufl.). Berlin: de Gruyter.

Herczeg, M. (2005). *Software Ergonomie – Grundlagen der Mensch-Computer-Kommunikation* (2. vollst. überarb. Aufl.). München: Oldenbourg.

Herczeg, M. (2006). *Interaktionsdesign. Gestaltung interaktiver und multimedialer Systeme.* München: Oldenbourg.

Heringer, H. J. (1979). Verständlichkeit: ein genuiner Forschungsbereich der Linguistik? *Zeitschrift für germanistische Linguistik 7*, 225-278.

Hevner, A. R. & March, S. (2003). The Information Systems Research Cycle. *IEEE Computer, 36(11)*, 111-113.

Hevner, A. R., March, S., Park, J. & Ram, S. (2004). Design Science Research in Information Systems. *MIS Quarterly 28(1)*, 75-105.

Hewett et al. (2008). ACM SIGCHI Curricula for Human-Computer Interaction. Letzter Zugriff am 28.12.2008 unter http://sigchi.org/cdg/cdg2.html#2_1.

Hildenbrand, T., Behm, A., Rashid A. & Geisser, M. (2006). Entwicklungsmethodiken zur kollaborativen Softwareerstellung – Stand der Technik. Letzter Zugriff am 01.08.2008 unter http://wifo1.bwl.uni-mannheim.de/fileadmin/files/publications/CollaBaWue-KSE-Arbeitsbericht-Stand_der_Technik-Methodiken.pdf.

Hillmann, K.-H. (2007). *Wörterbuch der Soziologie* (5. vollst. überarb. u. erw. Aufl.). Stuttgart: Kroener.

Hindelang, G. (2000). *Einführung in die Sprechakttheorie*. (3. Aufl.). Tübingen: Niemeyer.

Hirschmeier, M. (2005). *Wirtschaftlichkeitsanalysen für IT-Investitionen*. Duisburg: WiKu-Verlag.

Hoegh, R. T. (2006). Usability Problems: Do Software Developers Already Know? *ACM International Conference Proceeding Series 206*, 425-428.

Hörmann, K., Dittmann, L., Hindel, B. & Müller, M. (2006). *SPICE in der Praxis.* Heidelberg: dpunkt.

Holzer, J. (2005). *Linguistische Antropologie. Eine Rekonstruktion.* Bielefeld: transcript.

Holzinger, A. (2001). *Basiswissen Multimedia. Design.* (Bd. 3). Würzburg: Vogel.

Holzinger, A. (2004). Thinking-aloud – eine Königsmethode im Usability Engineering. *Journal der österreichischen Computergesellschaft 02(29)*, S. 4-5. Letzter Zugriff am 1.12.2008 unter http://user.meduni-graz.at/andreas.holzinger/holzinger%20de/papers%20de/G49_-HOLZINGER%20(2006)%20Thinking%20aloud.pdf.

Holzinger, A. (2005). Usability Engineering Methods (UEMs) for Software Developers. *Communications of the ACM 48(1)*, 71-74.

Holzinger, A. & Slany, W. (2006). *XP+UE+XU. Praktische Erfahrungen mit eXtreme Usability*. Berlin: Springer.

Holtzblatt, K., Wendell, J. B. & Wood, S. (2005). *Rapid Contextual Design. A How-to Guide to Key Techniques for User-Centered Design.* San Francisco: Morgan Kaufmann.

Homans, G. C. (1972). *Grundfragen soziologischer Theorie.* Opladen: Westdeutscher Verlag.

Horvath, P., Gleich, R. & Voggenreiter, D. (2001). *Controlling umsetzen* (3. überarb. u. erw. Aufl.). Stuttgart: Schäffer-Poeschel.

Hughes, M. (2006). A Pattern Language Approach to Usability Knowledge Management. *Journal of Usability Studies 1(2)*, 76-90.

IBM (2008). Erweitern Sie Ihr SOA-Know-how. Letzter Zugriff am 07.10.2008 unter http://www-01.ibm.com/software/de/solutions/soa/smartsoa/personal_value.html.

IEEE (1990). Std. 610.12-1990. *IEEE Standard Glossary of Software Engineering Terminology.* New York: The Institute of Electrical and Electronics Engineers.

Iivari, J., Hirschheim, R. & Klein, H. K. (2001). A dynamic Framework for Classifying Information Systems Development Methodologies and Approaches. *Journal of Management Information Systems 17(3),* 179-218.

Interface consult (2008). Was ist Usability. Letzter Zugriff am 28.12.2008 unter http://www.usability.at/usability/usability.html.

Iskander, M. (Hrsg.) (2008). *Innovative Technologies in Instruction Technology, E-learning, E-assessment and Education.* Berlin: Springer.

ISO/IEC 15504 (2006). SPICE Software Process Assessment - Part 7: Guide for use in process improvement. Genf: International Organization for Standardization.

ISO/IEC 25051 (2006). Software engineering – Software Product Quality Requirements and Evaluation (SQuaRE) – Requirements for Quality of Commercial Off-The-Shelf (COTS) Software Product and Instructions for Testing. Genf: ISO.

ISO/TR 16982 (2002). Ergonomics of human-system interaction – Usability methods supporting human-centred design. Genf: ISO.

Jablonski, S., Böhm, M. & Schulze, W. (Hrsg.). (1999). *Workflow-Management. Entwicklung von Anwendungen und Systemen* (korrigierter Nachdruck). Heidelberg: dpunkt.

Jablonski, S., Petrov, I., Meiler, C. & Mayer, U. (2004). *Guide to Web Application and Platform Architectures.* Berlin: Springer.

Jacobson, I., Christerson M., Jonsson, P. & Overgaard G, (1992). *Object-Oriented Software Engineering: A Use Case Driven Approach.* Reading, MA: ACM Press.

Jacobsen, J. (2006). Methoden der Usability. Letzter Zugriff am 13.11.2006 unter http://www.-contentmanager.de/magazin/artikel_630_methoden_der_usability.html.

James, G. A. & Gartner, Inc. (Hrsg.) (2008). *Magic Quadrant for Enterprise Architecture Tools.* Wiesbaden: Gartner.

Janich P. (1996). *Konstruktivismus und Naturerkenntnis: Auf dem Weg zum Kulturalismus.* Frankfurt a. Main: Suhrkamp.

Jarke, M. (1999). Scenarios for modelling. *Communications of the ACM 42(1),* 47-48.

Jayaratna, M. (1994). *Understanding and Evaluating Methodologies: NIMSAD, a systematic Framework*. Columbus: McGraw-Hill.

Jörgensen, A. H. (1984). On the Psychology of Protoyping. In R. Budde, K. Kuhlenkamp, L. Matthiassen & H. Züllighoven (Hrsg.), *Approaches to Prototyping* (S. 278-289). New York: Springer.

Johannesson, P. (1995). Representation and Communication – a Speech Act Based Approach to Information Systems Design. *Information Systems 20(4)*, 291-303.

Johannsen, W. & Goeken, M. (2006). IT-Governance - neue Aufgaben des IT-Managements. In H. Fröschle & S. Strahringer (Hrsg.), *IT-Governance* (S. 7–20). Heidelberg: dpunkt.

Johannsen, W. & Goeken, M. (2007). *Referenzmodelle für IT-Governance - Strategische Effektivität und Effizienz mit COBIT, ITIL & Co*. Heidelberg: dpunkt.

John, B. E., Bass., L., Kazman, R. & Chen, E. (2004). Identifying gaps between hci, software engineering, and design, and boundary objects to bridge them. *Proceedings of CHI '04: CHI 04 extended abstracts on Human factors in computing systems* (S. 1723-1724). New York: ACM Press.

Jokela T., Iivari, N., Karukka, M. & Matero, J. (2003). The Standard of User-Centered Design and the Standard Definition of Usability: Analyzing ISO 13407 against ISO 9241-11. *ACM International Conference Proceeding Series 46* (S. 53-60). New York: ACM Press.

JRules (2007). ILOG JRules. Letzter Zugriff am 22.03.2007 unter http://www.ilog.com/products/jrules.

Kalbach, J. (2003). Von Usability überzeugen. In S. Heinsen & P. Vogt (Hrsg.). *Usability praktisch umsetzen* (S. 7-21). München: Hanser.

Kalex, U. (2007). Von der Geschäftsarchitektur zur SOA-Governance. In G. Starke & S. Tilkov (Hrsg.), *SOA-Expertenwissen. Methoden, Konzepte und Praxis serviceorientierter Architekturen* (S. 325–340). Heidelberg: dpunkt.

Kambartel, F. (1981). Rekonstruktion und Rationalität – Zur Normativen Grundlage einer Theorie der Wissenschaft. In O. Schwemmer (Hrsg.), *Vernunft, Handlung und Erfahrung – Über die Grundlagen und Ziele der Wissenschaften* (S. 11-21). München: Beck.

Kamlah, W. (1975). *Von der Sprache zur Vernunft. Philosophie und Wissenschaft in der neuzeitlichen Profanität.* Zürich: Bibliographisches Institut.

Kamlah, W. & Lorenzen, P. (1967). *Logische Propädeutik oder Vorschule des vernünftigen Redens.* Mannheim: Bibliographisches Institut.

Kargl, H. (1990). *Fachentwurf für DV-Anwendungssyteme* (2. erg. Aufl.). München: Oldenbourg.

Kazman, R., Bass, L. & Bosch, J. (2003). Workshop overviews: Bridging the gaps between software engineering and human-computer interaction. *Proceedings of the 25th international conference on Software engineering* (S. 777-778). Washington: IEEE Computer Society.

Kecher, C. (2007). *UML 2.0. Das umfassende Handbuch* (2. Aufl.). Bonn: Galileo.

Keen, P. (1992). *Informationstechnologie - Der Weg in die Zukunft*. Wien: Ueberreuter.

Keller, R. K. (1989). *Prototypingorientierte Systemspezifikation: Konzepte, Methoden, Werkzeuge und Konsequenzen.* Hamburg: Kovac.

Keller, W. (2007). SOA-Governance - SOA langfristig durchsetzen und managen. In G. Starke & S. Tilkov (Hrsg.), *SOA-Expertenwissen. Methoden, Konzepte und Praxis serviceorientierter Architekturen* (S. 289–307). Heidelberg: dpunkt.

Klein, M. (1989). *Einführung in die DIN-Normen* (10. neubearb. u. erw. Aufl.). Stuttgart: Teubner.

Kobielus, J. (2006). SOA Governance: Preventing rogue services. Network World. Letzter Zugriff am 02.06.2008 unter http://www.networkworld.com/supp/2006/ndc3/062606-ndc-soagovernance.html.

Kock, N., Gray, P., Hoving, R., Klein, H., Myers, M. & Rockart, J. (2002). IS Research Relevance Revisited: Subtle Accomplishment, Unfulfilled Promise, or Serial Hypocrisy? *Communication of the Association for Information Systems 8* (S. 330-346).

Kollmann, T. (1998). *Akzeptanz innovativer Nutzungsgüter und -systeme. Konsequenzen für die Einführung von Telekommunikations- und Multimediasystemen.* Wiesbaden: Gabler.

Kollmann, T. (1999). Das Konstrukt der Akzeptanz im Marketing. *WiSt 3(3),* 125-130.

Kollmann, T. (2000). Die Messung der Akzeptanz bei Telekommunikationssystemen. *Wissenschaftsjournal 2*, 67-78.

Kollmann, T. (2004). Attitude, adoption or acceptance? – measuring the market success of telecommunication and multimedia technology. *International Journal of Business Performance Management 6(2)*, 133-152.

Kollmann, T. (2006). *E-Entrepreneurship* (2. Aufl.). Wiesbaden: Gabler.

Kosiol, E. (1972). *Die Unternehmung als wirtschaftliches Aktionszentrum – Einführung in die Betriebswirtschaftslehre.* Reinbek: Rowohlt.

Kosiol, E. (1976) *Organisation der Unternehmung* (2. durchges. Aufl.). Wiesbaden: Gabler.

Kowalk, W. (1996). *System – Modell – Programm.* Heidelberg: Spektrum Akademischer Verlag.

Kramberg, V. (2006). Pattern-based Evaluation of IBM WebSphere BPEL. Studienarbeit Nr. 2052, Universität Stuttgart. Letzter Zugriff am 20.10.2008 unter http://www.informatik.uni-stuttgart.de/cgi-bin/NCSTRL/NCSTRL_view.pl?id=STUD-2052&inst=FAK&mod=&engl.

Krämer, N. C. (2004). Mensch-Computer-Interaktion. In R. Mangold, P. Vorderer & G. Bente (Hrsg.), *Lehrbuch der Medienpsychologie* (S. 643-671). Göttingen: Hofgrefe.

Krauth, L. (1997). *Die Philosophie Carnaps* (2. Aufl.). Wien: Springer.

Krcmar, H. (2004). *Informationsmanagement* (4. überarb. und erw. Aufl.). Berlin: Springer.

Kroeber-Riel, W., & Weinberg P. (2003). *Konsumentenverhalten* (8. Aufl.). München: Vahlen.

Kroemer, K. H. E. & Grandjean, E. (2003). *Fitting the Task to the Human* (5. Aufl.). London: Taylor & Francis.

Kruchten, P. (1999). Use-case storyboards in the rational unified process. *Proceedings of the Workshop on Object-Oriented Technology* (S. 249-250). New York: Springer.

Kruchten, P. (2000). *Der Rational Unified Process: An Introduction* (2. Aufl.). Massachusetts: Addison-Wesley.

Krug, S. (2006). *Don't Make Me Think! A Common Sense Approach to Web Usability* (2. Aufl.) Berkerley: New Riders.

Kulpa, M. & Johnson, K. A. (2003). *Interpreting the CMMI*. Boca Rato: Auerbach.

Kwasnicki, W. (1996). *Knowledge, Innovation and Economy. An Evolutionary Exploration.* Cheltenham Brookfield: Edward Elgar Publishing.

Laartz, J., Funke. C. & Hoch, D. J. (2007). Software and Service Engineers to the Rescue – Die Sicherung einer führenden Rolle im internationalen Wettbewerb fordert einen neuen Typus von Ingenieur: Der Standort im Zeichen von Industrialisierung und Globalisierung. *Wirtschaftsinformatik, Sonderheft,* 144-145.

Lai, J. L. (1997). An assessment of the influence of organizational characteristics on information technology adoption decision: A discriminative approach. *IEEE Transactions on Engineering Management* (S. 146-157).

Landauer, T. K. (1995). *The Trouble with Computers. Usefulness, Usability, and Productivity.* Cambridge, MA: MIT Press.

Lassmann, W. (2006) (Hrsg.). *Wirtschaftsinformatik*. Wiesbaden: Gabler.

Latour, B. (1999). *Pandora's Hope. Essays on the Reality of Science Studies*. London: Havard University Press.

Lehmann, F. R. (1999). *Fachlicher Entwurf von Workflow-Management-Anwendungen*. Stuttgart: Teubner.

Lehmann, F. R. & Ortner, E. (1996). *Modellierung von Workflow-Management-Anwendungen.* Bericht, Technische Universität Darmstadt, Fachgebiet Wirtschaftsinformatik I.

Lehmann, F. R. & Ortner, E. (1999). *Repositorium-unterstützte Entwicklung von Workflow-Management-Anwendungen.* Bericht, Technische Universität Darmstadt, Fachgebiet Wirtschaftsinformatik I.

Lehner, F., Auer-Rizzi, W., Bauer, R., Breit, K., Lehner, J. & Reber, G. (1991). *Organisationslehre für Wirtschafsinformatiker.* München: Hanser.

Lehner, F. (2003). *Mobile und drahtlose Informationssysteme. Technologien, Anwendungen, Märkte.* Berlin: Springer.

Leist, S. (2004). *Methoden der Unternehmensmodellierung. Möglichkeiten und Grenzen ihrer Anwendung.* Frankfurt an der Oder: Universitätsfakultät für Wirtschaftswissenschaften.

Leithäuser, T. (1994). Ganzheit und Komplexität. Theoretische Perspektiven einer sozialpsychologischen Technikforschung. In B. Beuscher (Hrsg.*), Schnittstelle Mensch: Menschen und Computer – Erfahrungen zwischen Technologie und Anthropologie* (S. 69-82). Heidelberg: Asanger.

Lenz, A. (1991). *Knowledge Engineering für betriebliche Expertensysteme. Erhebung, Analyse und Modellierung von Wissen.* Wiesbaden: Deutscher Universitäts-Verlag.

Lexikon der Psychologie (2008). Spektrum Akademischer Verlag. Letzter Zugriff am 08.03.2008 unter http://www.wissenschaft-online.de/artikel/110571.

Leyking, K., Dreifus, F. & Loos, P. (2007). Serviceorientierte Architekturen. Vergleichende Literaturstudie. *Wirtschaftsinformatik (49)*, 394-402.

Liebau, H. B. (2007). Business Rules auf Basis des Semantic Web. Diplomarbeit, Technische Universität Dresden.

Liebermann, H., Paterno, F. & Wulf, V. (Hrsg.). (2006). *End user Development*. Dodrecht: Springer.

Liebhart, D. (2007). *SOA goes real: Service-orientierte Architekturen erfolgreich planen und einführen.* München: Hanser.

Lievrouw, L. & Livingstone, S. (2002). Introduction: The Social Shaping and Consequences of ICTs. In L. Lievrouw & S. Livingstone (Hrsg). *The Handbook of New Media* (S. 1-16). London: Sage.

Lievrouw, L. & Livingstone, S. (Hrsg.) (2002). *The Handbook of New Media.* London: Sage.

Lin, C. A. (2003). An Interactive Communication Technology Adoption Model. *Communication Theory 13(4)*, 345-365.

Linke, A., Ortner, H. & Portmann-Tselikas, P. R. (Hrsg.). (2003). *Sprache und mehr. Ansichten einer Linguistik der sprachlichen Praxis.* Tübingen: Niemeyer.

Lonthoff, J. (2007). *Externes Anwendungsmanagement. Organisation des Lebenszyklus komponentenbasierter, mobiler Anwendungen.* Wiesbaden: Deutscher Universitäts-Verlag.

Loos, C. F. (2005). Adoption von Unternehmensübergreifenden Projektplattformen. Letzter Zugriff am 16.06.2008 unter http://wifo1.bwl.uni-mannheim.de/dc2005/Beitraege_PDF/Loos_-Frederik.pdf.

Lorenzen, P. (1985). *Grundbegriffe technischer und politischer Kultur*. Frankfurt a. Main: Suhrkamp.

Lorenzen, P. (2000). *Lehrbuch der konstruktiven Wissenschaftstheorie.* Stuttgart: Metzler.

Ludwig, B., Mandl, S., Reiß, P., Görz, G. & Stoyan, H. (2006). Natürlichsprachliche Bedienung technischer Systeme. In C. Hochberger & R. Liskowsky (Hrsg.*), Informatik 2006 – Informatik für Menschen. LNI-Proceedings* (Bd. 2, P-94, S. 146-153). Bonn: Gesellschaft für Informatik.

Lüders, C. (2006). Teilnehmende Beobachtung. In R. Bohnsack, W. Marotzki & M. Meuser (Hrsg.), *Hauptbegriffe Qualitativer Sozialforschung* (2. Aufl., S.151-153). Stuttgart: UTB.

Luhmann, N. (1984). *Soziale Systeme. Grundriss einer allgemeinen Theorie.* Frankfurt a. Main: Suhrkamp.

Lyons, J. (1969). *Introduction to theoretical linguistics.* Cambridge: Cambridge University Press.

Macaulay, L. (1996). *Requirements Engineering.* Berlin: Springer.

Mai, C. (1994). Quality Function Deployment (QFD). In W. Masing (Hrsg.), *Handbuch Qualitätstechnik* (3. Aufl.). München: Hanser.

Mangold, R., Vorderer, P. & Bente, G. (Hrsg.). (2004). *Lehrbuch der Medienpsychologie.* Göttingen: Hofgrefe.

Manhartsberger, M. & Musil, S. (2001). *Web Usability. Das Prinzip des Vertrauens.* Bonn: Galileo Press.

March, S., Hevner, A. & Ram, S. (2000). "An Agenda for Information Technology Research in Heterogeneous and Distributed Environments," *Information Systems Research 11(4),* 327-341.

March, S. T. & Smith, G. (1995). Design and Natural Science Research on Information Technology. *Decision Support Systems 15(4),* 251-266.

Marchewka, J. T., Liu, C. & Kostiwa, K. (2007). An Application of the UTAUT Model for Understanding Student Perceptions Using Course Management Software. *Communicatios of the IIMA 7(2)*, 93-104.

Martin, J. (1991). *Rapid Application Development.* New York: MacMillan.

Mathieson, K. (1991). Predicting User Intentions: Comparing the Technology Acceptance Model with the Theory of Planned Behavior. *Information Systems Research 2(3),* 173-191.

Mayhew, D. J. (1994). Managing the Design of the User Interface. *Conference on Human Factors in Computing Systems* (S. 401-402). New York: ACM Press.

Mayhew, D. J. (1999a). *The Usability Engineering Lifecycle. A practitioner's Handbook for User Interface Design.* San Diego: Academic Press.

Mayhew, D. J. (1999b). *The Usability Engineering Lifecycle. Conference on Human Factors in Computing Systems – Tutorial Session* (S. 147-148). New York: ACM Press.

Meleghy, T. (2007). Verhaltenstheoretische Soziologie: George Caspar Homans. In J. Morel, T. Meleghy, H. J. Niedenzu, M. Preglau & H. Staubmann (Hrsg.), *Soziologische Theorie - Abriß der Ansätze ihrer Hauptvertreter*, (8. überarb. Aufl., S. 30-51). München: Oldenbourg.

Meffert, H. (1976). Die Durchsetzung von Innovationen in der Unternehmung und im Markt. *Zeitschrift für Betriebswirtschaft 2(46),* 77-100.

Memmel, T., Reiterer, H., Ziegler, H. & Oed, R. (2007). Visuelle Spezifikation zur Stärkung der Auftraggeberkompetenz bei der Gestaltung interaktiver Systeme. In K. Roese & H. Brau (Hrsg.), *Proceedings of 5th Workshop of the German Chapter of the Usability Professionals Association e.V.* (S. 99-104). Stuttgart: Frauenhofer IRB Verlag.

Memmel, T., Gundelsweiler, F. & Reiterer, H. (2007a). Agile Human-Centered Software Engineering. *Proceedings of 21st BCS HCI Group conference* (S. 167-175).

Memmel, T., Gundelsweiler, F. & Reiterer, H. (2007b). CRUISER: a Cross-Discipline User Interface & Software Engineering Lifecycle. In J. Jacko (Hrsg.), *Human-Computer Interaction - Interaction Design and Usability* (Part I, S. 174–183). Berlin: Springer.

Mertens, P., Bodendorf, F., König, W., Picot, A., Schumann, M. & Hess, T. (2005). *Grundzüge der Wirtschaftsinformatik.* Berlin: Springer.

Merz, M. (2002). *E-commerce und e-business: Marktmodelle, Anwendungen und Technologien* (2. überarb. u. aktual. Aufl.). Heidelberg: dpunkt.

mik21 (2004). Projektvorhaben: Migrationskompetenz als Schlüsselfaktor der Ökonomie des 21. Jahrhunderts. Kurzbeschreibung des Projektvorhabens, Universität Kassel. Letzter Zugriff am 28.04.2005 unter: http://www.mik21.uni-kassel.de/download/mik21-Projektvorhaben.pdf.

Mingers, J. (2003). The paucity of multimethod research: a review of the information systems literature. *Information Systems Journal 13(3)*, 233-249.

Mittelstraß, J. (1974). *Die Möglichkeit von Wissenschaft*, Frankfurt a. Main: Suhrkamp.

Mittelstraß, J. (1998). Information oder Wissen – vollzieht sich ein Paradigmenwechsel? *Phys. Bl. 54(5)*, Weinheim: Wiley-VCH Verlag.

Mittelstraß, J. (2000). *Zwischen Naturwissenschaft und Philosophie. Versuch einer Neuvermessung des wissenschaftlichen Geistes*. Konstanz: UVK.

Mittelstraß, J. (2001). *Wissen und Grenzen. Philosophische Studien.* Frankfurt a. Main: Suhrkamp.

Mittelstraß, J. (Hrsg.). (2004a). *Enzyklopädie Philosophie und Wissenschaftstheorie* (Bd. 2. H-O). Stuttgart: Metzler.

Mittelstraß, J. (Hrsg.). (2004b). *Enzyklopädie Philosophie und Wissenschaftstheorie* (Bd. 3. P-So). Stuttgart: Metzler.

Mittelstraß, J. (Hrsg.). (2004c). *Enzyklopädie Philosophie und Wissenschaftstheorie.* (Bd. 4. Sp-Z). Stuttgart: Metzler.

Mönke, H. (1978). Definitionstypen und Definitionsmatrix. *Nachrichten für Dokumentation, 29,* 51-60.

Molich, R., Coyne, K. P., Perkins, R. & Mayhew, D. J. (2003). Politics and Usability: Test Your Skills Against the Experts. *Conference on Human Factors in Computing Systems – Panel Session* (S. 694-695). New York: ACM Press.

Monk, A. F. & Frohlich, D. (1999). Computers and Fun. *Personal Technology, 3(1)*, 91.

Moore, G. C. & Benbasat, I. (1996). Integrating Diffusion of Innovations and Theory of Reasoned Action Models to Predict Utilitzation of Information Technology by End-Users. In K. Kautz & J. Pries-Hege (Hrsg.), *Diffusion and Adoption of Information Technology* (S. 132-146). London: Chapman and Hall.

Morel, J., Meleghy, T., Niedenzu, H.-J., Preglau, M. & Staubmann, H. (Hrsg). (2007). *Soziologische Theorie (Abriß der Ansätze ihrer Hauptvertreter)* (8. überarb. Aufl.). München: Oldenbourg.

Morris, C. W. (1937). *Logical Positivism, Pragmatism, and Scientific Empiricism.* Paris: Hermann & Cie.

Morris, C. W. (1946). *Signs, Language, and Behavior.* New York: Prentice-Hall.

Moser, C. & Kalton, G. (1971). *Survey Methods in Social Investigation* (2. Aufl.). London: Heinemann.

Müller-Böling, D. & Müller, M. (1986). *Akzeptanzfaktoren der Bürokommunikation*. München: Oldenbourg.

Nachira, F., Dini, P. & Nicolai, A. (2007). *A Network of Digital Business Ecosystems for Europe: Roots, Processes and Perspectives*. Digital Business Ecosystems. Luxembourg: Office for Official Publications of the European Communities.

Naël, M. (2000). User Interface within Telecommunications. In R. Reichwald & M. Lang (Hrsg.), *Anwenderfreundliche Kommunikationssysteme*. Heidelberg: Hüthig.

Nicolini, M. (2001). *Sprache Wissenschaft Wirklichkeit – zum Sprachgebrauch in inter- und transdisziplinärer Forschung.* Klagenfurt: Bundesministerium für Bildung, Wissenschaft und Kultur.

Nicolini, M., Freyer, B., Zinggl, W. & Treusch-Dieter, G. (2002). *Konturen. Statusbericht 22.04.2002*. Wien: Austrian Landscape Research.

Nielsen, J. (1993). *Usability Engineering*. San Diego: Academic Press.

Nielsen, J. (1995). Scenarios in discount usability engineering. In J. R., Carroll (Hrsg.), *Scenario-Based Design: Envisioning Work and Technology in System Development* (S. 59-83), Hoboken, NJ: John Wiley & Sons.

Nielsen, J. (1999). Web research: Believe the data. Alertbox, Letzter Zugriff am 03.08.2008 unter http://www.useit.com/alertbox/990711.html.

Nielsen, J. (2001). *Designing Web Usability*. München: Markt + Technik.

Nielsen, J. (2006). It's Worth the Hassle! The Added Value of Evaluation the Usability of Mobile Systems in the Field. *ACM International Conference Proceeding Series 189* (S. 272-280). New York: ACMPress.

Nielsen, J., Barnum, C., Bevan, N., Cockton, G., Spool, J. & Wixon, D. (2003). The "Magic Number 5": Is It Enough for Web Testing? *ACM CHI 2003* (S. 698-699). Fr. Lauderdale: ACM Press.

Nieters, J. E., Ivaturi, S. & Ahmed, I. (2007). Making personas memorable. *Proceedings of the Conference on Human Factors in Cumputing Systems* (S. 1817-1824). New York: ACM Press.

Nisbett, R. E. & Wilson, T. D. (1977). Telling More Than We Can Know: Verbal Reports on Mental Processes. *Psychological Review 84*, 231-259.

Noack, J. & Schienmann, B. (1999). Objektorientierte Vorgehensmodelle im Vergleich. *Informatik Spektrum 22(3)*, 166-180.

Nolting, H. P. & Paulus, P. (2008). *Psychologie lernen. Eine Einführung und Anleitung* (8. Aufl.). Berlin: Beltz.

Nonaka, I. (1991). The Knowledge-Creating Company. *Harvard Business Review 69(6)*, 96-104.

Nonaka, I. & Takeuchi, H. (1995). *The Knowledge-Creating Company. How Japanese Companies Create the Dynamics of Innovation.* New York: Oxford University Press.

Nordsieck, F. (1955). *Rationalisierung der Betriebsorganisation* (2. Aufl.). Stuttgart: Poeschel.

Nordsieck, F. (1962). *Die schaubildliche Erfassung und Untersuchung der Betriebsorganisation* (6. Aufl.). Stuttgart: Poeschel.

Norman, D. A. (1997). Why It's Good That Computers Don't Work Like the Brain. In P. J. Denning & R. M. Metcalfe (Hrsg.), *Beyond Calculation* (S. 105-116), New York: Springer.

Norman, D. A. (2002). *The Design of Everyday Things* (2. Aufl.). New York: Basic books.

Nunes, N. J. & e Cunha, J. F. (2000). Towards a UML Profile for Interaction Design: the Wisdom Approach. In A. Evans, S. Kent & B. Selic (Hrsg.), *"UML" 2000 – The Unified Modeling Language.* LNCS. Berlin: Springer.

Oestereich, B. (1998). *Objektorientierte Softwareentwicklung* (4. aktual. Aufl.). München: Oldenbourg.

Oestereich, B. & Weiss, C. (2008). *APM – Agiles Projektmanagement. Erfolgreiches Timeboxing für IT-Projekte*. Heidelberg: dpunkt.

Österle, H. & Winter, R. (2003). Business Engineering. In H. Österle & R. Winter (Hrsg.), *Business Engineering – Auf dem Weg zum Unternehmen des Informationszeitalters* (2. Aufl., S. 3-19). Berlin: Springer.

Oevermann, U. (1983). Zur Sache. Die Bedeutung von Adornos methodologischem Selbstverständnis für die Begründung einer materialen soziologischen Strukturanalyse. In J. Habermas & L. v. Friedburg (Hrsg.). *Adorno-Konferenz* (S. 234-289). Frankfurt a. Main: Suhrkamp.

OMG (2006). The Object Management Group. XMI-Schemadefinitionen zu SBVR. Letzter Zugriff am 01.11.2006 unter www.omg.org/docs/bei/05-08-02.zip.

OMG (2008a). The Object Management Group. Letzter Zugriff am 21.08.2008 unter http://www.omg.org.

OMG (2008b). Business Process Modeling Notation, V1.1. Letzter Zugriff am 01.10.2009 unter http://www.omg.org/docs/formal/08-01-17.pdf.

O'Neill, E. J. (2000). *User-Developer Cooperation in Software Development: Building Common Ground and Usable Systems*. Heidelberg: Springer.

Ortner, E. (1983). *Aspekte einer Konstruktionssprache für den Datenbankentwurf.* Darmstadt: Toeche-Mittler.

Ortner, E. (1993). Software-Engineering als Sprachkritik. Die sprachkritische Methode des fachlichen Software-Entwurfs. *Konstanzer Universitätsreden, 187*. Konstanz: UVK.

Ortner, E. (1994a). *Datenmodellierung – Systemunabhängiger Entwurf von Datenstrukturen*. Bericht, 44-94, Universität Konstanz, FG Informationswissenschaft.

Ortner, E. (1994b). KASPER - Ein Projekt zur natürlichsprachlichen Entwicklung von Informationssystemen. *Wirtschaftsinformatik, 36(6),* 570-579.

Ortner, E. (1997). *Methodenneutraler Fachentwurf. Zu den Grundlagen einer anwendungsorientierten Informatik.* Stuttgart: Teubner.

Ortner, E. (2003a). Vom Wort zum Bauelement. *Thema Forschung 1*. Darmstadt: TUD.

Ortner, E. (2003b). Entwicklung von eBusiness – Anwendungen. Skriptum zur Vorlesung „E-Business“ im Masterstudiengang „Business Information Systems“ der FH Liechtenstein. Darmstadt: TUD.

Ortner, E. (2005). *Sprachbasierte Informatik. Wie man mit Wörtern die Cyber-Welt bewegt.* Leipzig: Edition am Gutenbergplatz.

Ortner, E. (2008a). From Software Engineering to Enterprise Engineering – Introduction to a Language-critical Approach. In M. Iskander (Hrsg.), *Innovative Technologies in Instruction Technology, E-learning, E-assessment and Education* (S. 135-143). Berlin: Springer.

Ortner, E. (2008b). Process-centric Enterprise Modeling & Management (ProCEM®). *Proceedings of the 3rd International Conference on Evaluation of Novel Approaches to Software Engineering.* Madeira.

Ortner, E. & Eller, B. (2008). Das elastische Unternehmen. Von der nutzerzentrierten Systemanalyse zu effektiven SOA-Anwendungen. *Tutorium an der MKWI 2008*, München.

Ortner, E., Eller, B. & Elzenheimer, M. (2008). *Das elastische Unternehmen – Wie die heutige Informationstechnologie von dynamischen Unternehmen eingesetzt werden könnte.* Bericht 01. Technische Universität Darmstadt, Wirtschaftsinformatik I.

Ortner, E. & Söllner, B. (1989). Semantische Datenmodellierung nach der Objekttypenmethode. *Informatik-Spektrum 12(1),* 31-42.

Ortner, E. & Wedekind, H. (2003). *Die Zukunft des Bürgers im Internet*. In E. Ortner (Hrsg.), Arbeitsbericht des Fachgebiets Wirtschaftsinformatik I. Darmstadt: TUD.

Paech, B. (2000). *Aufgabenorientierte Softwareentwicklung. Integrierte Gestaltung von Unternehmen, Arbeit und Software.* Berlin: Springer.

Papazoglou, M., Traverso, P., Dustdar, S. & Leymann, F. (2006). Service-Oriented Computing Research Roadmap. Letzter Zugriff am 23.05.2008 unter http://infolab.uvt.nl/pub/papazogloump-2006-96.pdf.

Papazoglou, M. & Van Den Heuvel, W. J. (2006). Service-oriented design and development methodology. *International Journal of Web Engineering and Technology 4(2)*, 412–442.

Paternò, F. (2001). Towards a UML for interactive systems. *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction* (S. 7-18). New York: Springer.

Paulk, M., Chrissis, M. B., Weber, C. & Perdue, J. (1997*). The Capability Maturity Model for Software,* Version 2B. Pittsburgh, PA 15213-3890, 92.

Pawlow, T. (1980). *Begriffsbildung und Definition*. Sammlung Göschen, 2213. Berlin: Walter de Gruyter.

Petersen, P. (1997). Der Terminus Gewalt: Versuch einer terminologischen Bestimmung auf Grundlage des methodischen Konstruktivismus. In Arbeitsgruppe Konstruktive Erziehungswissenschaft am Institut für Pädagogik (Hrsg.), *Monographien zur Erziehungswissenschaft* (Bd. 4.) Kiel.

Peyret, H. (Hrsg.). (2007). The Forrester Wave: Enterprise Architecture Tools, Quartal 2. Letzter Zugriff am 28.07.2008 unter http://www.metastorm.com/library/reports/AR_PV_Forrester_-Wave_EA_Q207.pdf.

Pichler, R. (2008). *Scrum – Agiles Projektmanagement erfolgreich einsetzen.* Heidelberg: dpunkt.

Picot, A. (1982). Transaktionskostenansatz in der Organisationstheorie. *Die Betriebswirtschaft 42(2)*, S. 267-284.

Picot, A. & Dietl, H. (1990). Transaktionskostentheorie. *Wirtschaftswissenschaftliches Studium (4),* S. 178-184.

Poi, S., Sleight, B., Viezel, J. & Snavely, D. (2007). Enabling SOA through organizational change and governance - IBM. Letzter Zugriff am 25.03.2007 unter ftp://ftp.software.ibm.com/software/-solutions/soa/gov/GBS_SOAgov_Org_Change.pdf.

Pohl, K. (2008). *Requirements Engineering.* Heidelberg: dpunkt.

Pomberger, G., Pree, W. & Stritzinger, A. (1992). Methoden und Werkzeuge für das Prototyping und ihre Integration. *Informatik Forschung und Entwicklung 7*, 49-61.

Pomberger, G., Bischofberger, W., Kolb, D., Pree, W. & Schlemm, H. (1991). Prototyping-Oriented Software Development – Concepts and Tools. *Structured Programming 12(1)*, 43-60.

Porter, M. E. (1989). *Wettbewerbsvorteile. Spitzenleistungen erreichen und behaupten* (Sonderausgabe). Frankfurt a. Main: Campus.

Preece, J., Rogers, Y. & Sharp, H. (2002). *Interaction Design – beyond human-computer interaction*. New York: Wiley.

Preglau, M. (2007). Kritische Theorie: Jürgen Habermas. In J. Morel, T. Meleghy, H.-J. Niedenzu, M. Preglau & H. Staubmann (Hrsg), *Soziologische Theorie (Abriß der Ansätze ihrer Hauptvertreter)* (8., überarb. Aufl., S. 240-265). München: Oldenbourg.

Probst, G., Raub, S. & Romhardt K. (2006). *Wissen managen. Wie Unternehmen ihre wertvollste Ressource optimal nutzen* (5. Aufl.). Wiesbaden: Gabler.

Pyla, P. S., Howarth, J. R., Catanzaro, C. & North, C. (2006). Vizability: A Tool for Usability Engineering Process Improvement through the Visualization of Usability Problem Data. *Proceedings of the 44th annual Southeast regional conference* (S. 620-625), New York: ACM Press.

Pyla, P. S., Pérez-Quinones, M. A., Arthur, J. D. & Hartson, H. R. (2005). Ripple: An Event Driven Design Representation Framework for Integrating Usability and Software Engineering Life Cycles. In A. Seffah, J. Gulliksen & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 245-265). Dordrecht: Springer.

Quiring, O. (2006). Methodische Aspekte der Akzeptanzforschung bei interaktiven Medientechnologien. Münchner Beiträge zur Kommunikationswissenschaft 6(12), Letzter Zugriff am 08.03.2008 unter http://epub.ub.uni-muenchen.de/1348/1/mbk_6.pdf.

Rausch, A. & Broy, M. (2006). *Das V-Modell XT – Grundlagen, Erfahrungen, Werkzeuge* (1. Aufl.). Heidelberg: dpunkt.

Rechenberg, P. & Pomberger, G. (Hrsg.). (2006*). Informatik-Handbuch* (4. aktual. u. erw. Aufl.). München: Hanser.

Reeps, I. (2004). Usability, Design and Joy of Use. State-of-the-Art Analyse. Letzter Zugriff am 30. 04. 2006 unter http://hci.uni-konstanz.de/downloads/STAR_Reeps.pdf.

Reich, K. (Hrsg.). Methodenpool. Letzter Zugriff am 28.03.2008 unter http://methodenpool.uni-koeln.de.

Reichwald, R. (1982). *Neue Systeme der Bürotechnik – Beiträge zur Büroarbeitsgestaltung aus Anwendersicht.* Berlin: Schmidt.

Reiß, M. (1997). Aktuelle Konzepte des Wandels. In M. Reiß, L. von Rosenstiel & A. Lanz (Hrsg.), *Change Management* (S. 31-90). Stuttgart: Schäffer-Poeschel.

Reiterer, H. (2005). Beiträge zur Forschungsdisziplin Mensch-Computer Interaktion. Letzter Zugriff am 27.12.2008 unter http://www.ub.unikonstanz.de/kops/volltexte/2007/3168/.

Riedemann, K. (1988). Beruhigend. In R. Bülow (Hrsg.), *Denk Maschine!* (S. 340). München: Heyne.

Rode, P., Spors, K. & Krömker, H. (2007). Einsatz virtueller Produktdarstellungen für Kundentests in der Produktentwicklung. In G.,Kempter & M., Dontschewa (Hrsg), *uDay V. Informieren mit Comupteranimation.* Berlin: Papst Science Publishers.

Roberts, D., Berry, D., Isensee, S. & Mullaly, J. (1998). *Designing for the User with OVID: Bridging the Gap Between Software Engineering and User Interface Design,* Indianapolis: MacMillan Technical Publishing.

Rogers, E. M. (1962). *Diffusion of Innovations* (1. Aufl.). New York: The Free Press.

Rogers, E. M. (1983). *Diffusion of Innovations* (3. Aufl.). New York: The Free Press.

Rogers, E. M. (1995). *Diffusion of Innovations* (5. Aufl.). New York: The Free Press.

Rogers, E. M. & Shoemaker, F. F. (1971). *Communication of Innovations: A Cross Cultural Approach*. New York: The Free Press.

Rogers, E.M. & Shoemaker, F.F. (1983). *The Diffusion of Innovations.* New York: The Free Press.

Roithmayr, F. & Kainz, G. A. (1994). Umfrage zu laufenden Dissertationen im Fachgebiet Wirtschaftsinformatik – ein Beitrag zur Paradigmendiskussion. *Wirtschaftsinformatik 36(2)*, 178-184.

Rosson, M. B. (1999). Integrating development of task and object models. *Communications of the ACM 42(1)*, 49-56.

Rosson, M. B. & Carroll, J. M. (1992). Getting around the task-artifact frame work: How to make claims and design by scenario. *ACM Transactions on Information Systems 10(2),* 1046-1088.

Rosson, M. B. & Carroll, J. M. (2002). *Usability Engineering – Scenario-Based Development of Human Computer-Interaction.* San Diego: Academic Press.

Roth, K. (2000). *Konstruieren mit Konstruktionskatalogen: Band 1: Konstruktionslehre.* Berlin: Springer.

Rupp, C. (2007). *Requirements Engineering und –Management* (4. aktual. u. erw. Aufl.). München: Hanser.

Rupp, C., Queins, S. & Zengler, B. (2007). *UML 2 glasklar* (3. aktual. Aufl.). München: Hanser.

Russel, K. (2002). System Development Life Cycle. Letzter Zugriff am 03.04.007 unter http://www.computerworld.com/printthis/2002/0,4814,71151,00.html.

Sanderson, P. M. (2003). Cognitive Work Analysis. In J. M. Carroll (Hrsg.), *HCI Models, Theories and Frameworks. Toward a Multidisciplinary Science.* San Francisco: Elsevier Science.

Sarodnick, F. & Brau, H. (2006). *Methoden der Usability Evaluation. Wissenschaftliche Grundlagen und praktische Anwendung.* Bern: Hans Huber.

Sauer, C. (1995). Ein Minimalmodell zur Verständlichkeitsanalyse und –optimierung. In B. Spillner (Hrsg.), *Sprache: Verstehen und Verständlichkeit. Kongressbeitrage zur 25. Jahrestagung der Gesellschaft für Angewandte Linguistik* (S. 149-171). Frankfurt a. Main: Lang.

Scharbert, K. (2005). *Requirements Analysis realisieren*. Wiesbaden: Vieweg & Sohn.

Scheer, A. W. (2002). *ARIS. Vom Geschäftsprozess zum Anwendungssystem*. Berlin: Springer.

Schelp, J., Stutz, M. & Winter, R. (2007). SOA-Risiken und SOA-Governance. In G. Starke & S. Tilkov (Hrsg.), *SOA-Expertenwissen. Methoden, Konzepte und Praxis serviceorientierter Architekturen* (S. 661–668). Heidelberg: dpunkt.

Schienmann, B. (1997). *Objektorientierter Fachentwurf – Ein terminologiebasierter Ansatz für die Konstruktion von Anwendungssystemen*. Stuttgart: Teubner.

Schienmann, B. (2002). *Kontinuierliches Anforderungsmanagement. Prozesse – Techniken - Werkzeuge.* München: Addison-Wesley.

Schmidt, B. M. (2003). Wie kann avancierte Systemtheorie mit einer Wissenschaft des Bewusstseins kooperieren? In O. Jahraus & N. Ort (Hrsg.), *Theorie-Prozess-Selbstreferenz. Systemtheorie und transdisziplinäre Theoriebildung* (S. 49-68) Konstanz: UVK.

Schmidt-Eisenlohr, K. (2007). Wandel der Arbeitswelt und der Berufe in der IT-Branche. Hausarbeit am Institut für BWL, Fachgebiet Wirtschaftsinformatik. Darmstadt: TUD.

Schnotz, W. (1994). *Aufbau von Wissensstrukturen. Untersuchungen zur Kohärenzbildung bei Wissenserwerb mit Texten.* Weinheim: Beltz.

Schnotz, W. (2000). Das Verstehen schriftlicher Texte als Prozeß. In K. Brinker, G. Antos, W. Heinemann & S. F. Sager (2000), *Text- und Gesprächslinguistik. Ein internationales Handbuch zeitgenössischer Forschung* (S. 497-506). Berlin: de Gruyter.

Schober, P. (2008). Individuelles und kollektives Wissen in Organisationen. Das Verfahren GABEK® als Darstellungs- und Analyseinstrument. In J. Zelger, M. Raich & P. Schober (Hrsg.), *GABEK III – Organisationen und ihre Wissensnetze* (S. 123-142). Innsbruck: Studien Verlag.

Schrodt, R. (1999). *Sprechakte, Sprachhandlungen und Kommunikationstechnik. In P. Ernst (Hrsg.), Einführung in die synchrone Sprachwissenschaft* (2. Aufl., Kap. 6). Wien: Edition Praesens.

Schubert, K. & Klein, M. (2006). *Das Politiklexikon* (4. aktual. Aufl.). Bonn: Dietz.

Schuler, H. & Sonntag, K. (Hrsg.). (2007). *Handbuch der Psychologie. Handbuch der Arbeits- und Organisationspsychologie* (Bd. 6). Göttingen: Hofgrefe.

Schwender, C. (1997). Versuch einer Definition. Gebrauchsanleitung – eine Anleitung zum Gebrauch. Letzter Zugriff am 21.09.2008 unter http://www.schwender.in-berlin.de/td/index.html.

Searle, J. R. (1969). *Speech Acts.* Cambridge: Cambridge University Press.

Searle, J. R. (1976). A Classification of illocutionary acts, *Language in Society 5*, 1-23.

Searle, J. R. (1982). *Ausdruck und Bedeutung. Untersuchungen zur Sprechakttheorie.* Frankfurt a. Main: Suhrkamp.

Searle, J. R. (1983). *Sprechakte.* Frankfurt a. Main: Suhrkamp.

Searle, J. R. (1997). *Die Konstruktion der gesellschaftlichen Wirklichkeit. Zur Ontologie sozialer Tatsachen.* Hamburg: Rowohlt.

Seffah, A. & Hayne, C. (1999). Integrating Human Factors into Use Cases and Object-Oriented Methods. *Lecture Notes in Computer Science, 1743*, 240-250.

Seffah, A., Desmarais, M. C. & Metzker, E. (2005). HCI, Usability and Software Engineering Integration: Present and Future. In A. Seffah, J. Gulliksen & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 37-57). Dordrecht: Springer.

Seffah, A., Gulliksen, J. & Desmarais, M. C. (Hrsg.). (2005a). Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle. *Human-Computer Interaction Series, 8.* Dordrecht: Springer.

Seffah, A., Gulliksen, J. & Desmarais, M. C. (2005b). An Introduction to Human-Centered Software Engineering: Integrating Usability in the Development Process. In A. Seffah, J. Gulliksen & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering – Integration Usability in the Software Development Lifecycle* (S. 3-14). Dordrecht: Springer.

Seiffert, H. (2003). *Einführung in die Wissenschaftstheorie 1 (Sprachanalyse, Deduktion, Induktion in Natur- und Sozialwissenschaften)* (12. Aufl.). München: Beck.

Seland, G. (2006). System Designer Assessments of Role Play as a Design Method: A Qualitative Study. *ACM International Conference Proceeding Series, 189* (S. 222-231). New York: ACM Press.

Sen, A. (2000). The Discipline of Cost-Benefit Analysis. *Journal of Legal Studies 29(2)*, 931-952.

Sessions, R. (2007). Interview with John Zachman, in Perspectives of the International Association of Software Architects. Letzter Zugriff am 20.09.2008 unter www.iasaarchitects.org.

Seufert, A. & Seufert, S. (1998). Wissensgenerierung und -transfer in Knowledge Networks. Knowledge Networks als Basis für die Steigerung der Lernfähigkeit von Unternehmen. *IO Management 67(10)*, 76-84.

Sheppard, B.H., Hartwick, J. & Warshaw, P.R. (1988). The theory reasoned action: A meta-analysis of past research with recommendations for modifications and future research. *Journal of Consumer Research 15*, 325-343.

Shneiderman, B. & Plaisant, C. (2005). *Designing the User Interface – Strategies for effective Human-Computer-Interaction* (4. Aufl.). New York: Pearson.

Sieckenius de Souza, C. (2004). *The Semiotic Engineering of Human-Computer-Interaction.* Cambridge, MA: MIT Press.

Simon, B. (2001). *E-Learning an Hochschulen. Gestaltungsräume und Erfolgsfaktoren von Wissensmedien*. Köln: Josef Eul.

Sjoberg, D. I. K., Dyba, T. & Jörgensen, M. (2007). The Future of Empirical Methods in Software Engineering Research. *International Conference on Software Engineering* (S. 358-378). Washington: IEEE Computer Society.

Sjoberg, D. I. K., Dyba, T., Anda, B. C. D. & Hannay, J. E. (2008). Building Theories in Software Engineering. In F. Shull, J., Singer & D. I. K. Sjoberg (Hrsg.), *Guide to Advanced empirical Software Engineering* (S. 312-336). London: Springer.

Sommerville, I. (2001). *Software Engineering* (6. Aufl.). München: Pearson.

Sommerville, I. & Kotonya, G. (1998). *Requirements Engineering: Processes and Techniques.* New York: Wiley.

Stahlknecht, P. & Hasenkamp, U. (2005). *Einführung in die Wirtschaftsinformatik* (11. Aufl.). Berlin: Springer.

Stapelkamp, T. (2007). *Screen- und Interfacedesign. Gestaltung und Usability für Hard- und Software.* Berlin: Springer.

Starke, G. & Tilkov, S. (Hrsg.)(2007). *SOA-Expertenwissen - Methoden, Konzepte und Praxis serviceorientierter Architekturen.* Heidelberg: dpunkt.

Stary, C. (Hrsg.). (2005). *Mensch & Computer 2005. Kunst und Wissenschaft – Grenzüberschreitung der interaktiven ART.* München: Oldenbourg.

Steitz, J. (2008). Usability Engineering – Arbeitsabnahme, Arbeitserleichterung und Arbeitsbefähigung. Diplomarbeit. Technische Universität Darmstadt, Fachgebiet Wirtschaftsinformatik I.

Stelzer, D. (2000). Stand des Qualitätsmanagements in der Softwareentwicklung. In W. Hummeltenberg (Hrsg.), *Information Management for Business and Competitive Intelligence and Excellence* (S. 313-326). Wiesbaden: Vieweg.

Stewart, T. (1997). *Intellectual Capital. The New Wealth of Organizations.* New York: Currency Doubleday.

Sterr, U. & Stöckel, F. (2003). Kontrollierte Verbesserung von Prozessmetriken zur Erreichung von Unternehmenszielen gemäß CMMI Standard. Letzter Zugriff am 30. 04. 2006 unter http://www.hood-group.com/de/veroeffentlichungen/.

Stoecklin, S., Smith, S. & Serino, C. (2007). Teaching Students to Build Well formed Object-oriented Methods through Refactoring. *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (S. 145-149). New York: ACM Press.

Strahringer, S. (1996). *Metamodellierung als Instrument des Methodenvergleichs. Eine Evaluierung am Beispiel objektorientierter Analysemethoden.* Aachen: Shaker.

Strahringer, S. (Hrsg.). (2005). Business Engineering. *HMD – Praxis der Wirtschaftsinformatik, 241.* Heidelberg: dpunkt.

Suchman, L. & Jordan, B. (1990). Interactional troubles in face-to-face survey interviews. *Journal of the American Statistical Assoziation 85(409)*, 232-241.

Sutcliffe, A. G. (2005). Convergence or Competition Between Software Engineering and Human Computer Interaction. In A. Seffah, J. Gulliksen, & M. C. Desmarais (Hrsg.), *Human-Centered Software Engineering - Integration Usability in the Software Development Lifecycle* (S. 71-84). Dordrecht: Springer.

Sveiby, K. (1997). *The New Organizational Wealth. Managing & Measuring Knowledge-Based Assets.* San Francisco: Berret-Koehler Publishers.

Szyperski, C. (1998). *Component Software – Beyond Object-Oriented Programming.* Harlow: Addison-Wesley.

Taylor, S. & Todd, P. A. (1995a). Assessing IT Usage: The Role of Prior Experience. *MIS Quarterly 19(2),* 561-570.

Taylor, S. & Todd, P. A. (1995b). Understanding Information Technology Usage: A Test of Competing Models. *Information Systems Research, 6(4),* 144-176.

TECHNUM (2009). BPMN to Text. Letzter Zugriff am 24.02.2009 unter http://www.technum.biz/de-/technum/services/comtools/bpmn_to_text.php.

Tedeschi, B. (2002). *Putting tinsel into web shopping*. New York Times.

Teubner, R. A. (1999). Organisations- und Informationssystemgestaltung. Theoretische Grundlagen und integrierte Methoden. Wiesbaden: Dissertationsschrift Universität Münster.

Thiesse, F. (2001). Prozessorientiertes Wissensmanagement: Konzepte, Methode, Fallbeispiele. Dissertationsschrift der Universität St. Gallen, Nr. 2475, Bamberg: Difo-Druck.

Thome, R. (2006). Definition von Integration. Letzter Zugriff am 30.03.2008 unter http://www.wiinf.uni_wuerzburg.de/Dateianlagen/Lehrveranstaltungen/Wiinf/Sonstiges/integrationsdefinition.pdf.

Thompson, R. L., Higgins, C. A. & Howell, J. M. (1991). Personal Computing: Toward a Conceptual Model of Utilization. *MIS Quarterly 15(1),* 124-143.

Tornatzky, L. G. & Klein, K. J. (1982). Innovation Characteristics and Innovation Adoption-Implementation: A Meta-Analysis of Findings. *IEEE Transaction on Engineering Management 29(1),* 28-45.

Trenner, L. & Bawa, J. (1998). *The politics of usability: a practical guide to designing usable systems in industry*. London: Springer.

Triandis, H. C. (1977). *Interpersonal Behavior.* Monterey, CA: Brooke/Cole.

Turowski, K. (2001). Fachkomponenten – Komponentenbasierte betriebliche Anwendungssysteme. Habilitationsschrift der Universität Magdeburg.

Ulrich, P. & Fluri, E. (1995). *Management* (7. Aufl.). Stuttgart: UTB.

Universität Bern (2009). Abteilung für Kognitionspsychologie. Letzter Zugriff am 09.05.2009 unter http://www.kog.psy.unibe.ch/content/.

Universität Köln (2008). Konstruktiver Methodenpool. Letzter Zugriff am 01.12.2008 unter http://methodenpool.uni-koeln.de/frameset_uebersicht.htm.

Usability first (2008). Usability in Website and Software Design. Letzter Zugriff am 28.12.2008 unter http://www.usabilityfirst.com/index.txl.

usability.gov (2008). Step-by-Step Usability Guide. Letzter Zugriff am 28.12.2008 unter http://www.usability.gov/.

Vallerand, R. J. (1997). Toward a Hierarchical Model of Intrinsic and Extrinsic Motivation. In M. Zanna (Hrsg.), *Advances in Experimental Social Psychology (29)* (S. 271-360). New York: Academic Press.

van Harmelen, M. et al. (1997). Object models in user interface design: A CHI 97 workshop. *ACM SIGCHI Bulletin 29(4)*, 55-62.

van Harmelen, M. (Hrsg.), Wilson, S. (1997). *Object Modeling and User Interface Design: Designing Interactive Systems.* Boston: Addison-Wesley.

Venkatesh, V. & Morris, M. G. (2000). Why don`t men ever stop to ask for directions? Gender, social influence, and their role in technology acceptance and usage behavior. *MIS Quarterly 1(24),* 115-139.

Venkatesh, V. & Speier, C. (1999). Computer Technology Training in the Workplace: A Longitudinal Investigation of the Effect of the Mood. *Organizational Behavior and Human Decision Processes 79(1)*, 1-28.

Venkatesh, V., Morris, M. G., Davis, G. B. & Davis, F. D. (2003). User Acceptance of Information Technology: Toward a Unified View. *MIS Quarterly 27(3),* 425-478.

Vetter, M. (1998). *Aufbau betrieblicher Informationssysteme mittels pseudo-objektorientierter, konzeptioneller Datenmodellierung* (8. durchges. Aufl.). Stuttgart: Teubner.

visual rules (2007). Innovations: visual rules. Letzter Zugriff am 22.03.2007 unter http://www.visual-rules.de/00_home/home.html.

vom Brocke, J. (2006a). Serviceorientiertes Prozesscontrolling. Gestaltung von Organisation- und Informationssystemen bei Serviceorientierten Architekturen. Habilitationsschrift, Münster: Westfälische Wilhelms-Universität.

vom Brocke, J. (2006b). Design Principles for Reference Modelling. Reusing Information Models by Means of Aggregation, Specialisation, Instantiation, and Analogy. In P., Fettke & P. Loos (Hrsg.), *Reference Modelling for Business Systems Analysis* (S. 47-75). PA, USA: Idea Group.

von Rosenstiel, L. (2007). *Grundlagen der Organisationspsychologie* (6. Aufl.). Stuttgart: Schäffer-Poeschel.

Wandke, H. (2007). Mensch-Computer-Interaktion. In H., Schuler & K., Sonntag (Hrsg*.), Handbuch der Psychologie. Handbuch der Arbeits- und Organisationspsychologie* (Bd. 6, S. 203-216). Göttingen: Hofgrefe.

Wandmacher, J. (2002). *Einführung in die psychologische Methodenlehre.* Heidelberg: Spektrum.

Wedekind, H. (1976). *Systemanalyse: Die Entwicklung von Anwendungssystemen für Datenverarbeitungsanlagen.* München: Hanser.

Wedekind, H. (1981). *Datenbanksysteme I. Eine konstruktive Einführung in die Datenverarbeitung in Wirtschaft und Verwaltung* (2. völlig neu bearb. Aufl.). Mannheim: B. I. – Wissenschaftsverlag.

Wedekind, H., Görz, G., Kötter, R. & Inhetveen, R. (1998). Modellierung, Simulation, Visualisierung: Zu aktuellen Aufgaben der Informatik. *Informatik Spektrum 21(5),* 265-272.

Wedekind, H. & Ortner, E. (1980). *Systematisches Konstruieren von Datenbankanwendungen – Zur Methodologie der Angewandten Informatik.* München: Hanser.

Wedekind, H., Ortner, E. & Inhetveen, R. (2004a). Informatik als Grundbildung. *Informatik-Spektrum 27(2)*, 172-180.

Wedekind, H., Ortner, E. & Inhetveen, R. (2004b). Informatik als Grundbildung, Teil II: Bildung von Elementarsätzen. *Informatik-Spektrum 27(3),* 265-272.

Wedekind, H., Ortner, E. & Inhetveen, R. (2004c). Informatik als Grundbildung, Teil III: Gleichheit und Abstraktion. *Informatik-Spektrum 27(4)*, 337-342.

Wedekind, H., Ortner, E. & Inhetveen, R. (2004d). Informatik als Grundbildung, Teil IV: Objektsprache / Metasprache. *Informatik-Spektrum 27(5),* 459-466.

Wedekind, H., Ortner, E. & Inhetveen, R. (2004e). Informatik als Grundbildung, Teil V: Namengebung und Kennzeichnung. *Informatik-Spektrum 27(6)*, 551-556.

Wedekind, H., Ortner, E. & Inhetveen, R. (2005). Informatik als Grundbildung, Teil VI: Logik und Geltungssicherung. *Informatik-Spektrum 28(1)*, 48-52.

Weidenmann, B. (1999). Psychologie des Nichtverstehens, In J., Hennig & M., Tjarks-Sobhani (Hrsg.), *Verständlichkeit und Nutzungsfreundlichkeit technischer Dokumentation* (S. 34-49). Lübeck: Schmidt-Römhild.

Weill, P. & Ross, J. (2004). *IT Governance - How Top Performers Manage IT.* Harvard: Harvard Business School Press.

Wilde, T. & Hess, T. (2007). Forschungsmethoden der Wirtschaftsinformatik. Eine empirische Untersuchung. *Wirtschaftsinformatik, 49(4),* 280-287.

Wilde, T., Hess, T. & Hilbers, K. (2008). Akzeptanzforschung bei nicht marktreifen Technologien: typische methodische Probleme und deren Auswirkungen. In M., Bichler et al. (Hrsg.), *Multikonferenz Wirtschaftsinformatik 2008* (S. 1031-1042). Berlin: GITO-Verlag,

Wimmer, R. (2003). Wie kann man Sprachkritik begründen? In A., Linke, H. Ortner & P. R., Portmann-Tselikas (Hrsg.), *Sprache und mehr. Ansichten einer Linguistik der sprachlichen Praxis* (S. 417-450). Tübingen: Max Niemeyer.

Winand, U. (2000). *Entwurfsprinzip Alltagstauglichkeit – Konzepte zur Gestaltung von medien- und kommunikations- und informationstechnisch gestützten Anwendungssystemen und Diensten.* Arbeitsbericht 27, Informatik der Universität Kassel.

Winograd, T. (1997). The Design of Interaction. In P. J., Denning & R. M., Metcalfe (Hrsg.), *Beyond Calculation* (S. 149-167). New York: Springer.

Wirfs-Brock, R. (1990). Responsibility-Driven Design. Letzter Zugriff am 25.10.2007 unter http://www.wirfs-brock.com/Design.html.

Wirtz, M. & Caspar, F. (2002). *Beurteilerübereinstimmung und Beurteilerreliabilität. Methoden zur Bestimmung und Verbesserung der Zuverlässigkeit von Einschätzungen mittels Kategoriensystemen und Ratingskalen*. Göttingen: Hofgrefe.

Wittlage, H. (1993). *Methoden und Techniken praktischer Organisationsarbeit* (3. überarb. u. erw. Aufl.). Berlin: Herne.

Wittgenstein, L. (1977). *Philosophische Untersuchungen.* Frankfurt a. Main: Suhrkamp.

Wittgenstein, L. (1984). *Logisch-philosophische Abhandlung. Tractatus logico-philosophicus* (1. Aufl., 13. Nachdruck, 2000). Frankfurt a. Main: Suhrkamp.

Wittgenstein, L. (2001). *Philosophische Untersuchungen* (Kritisch-genetische Edition) von J., Schulte, H., Nymyn, E., v. Savigny & G. H., v. Wright (Hrsg.). Frankfurt a. Main: Suhrkamp.

Wixon, D. R. & Ramey, J. (Hrsg.) (1996). *Field Methods Casebook for Software Design*. New York: John Wiley.

Wunderer, R. & Mittmann, J. (1995). *Identifikationspolitik. Einbindung des Mitarbeiters in den unternehmerischen Wertschöpfungsprozess.* Stuttgart: Schäffer-Poeschel,

Wurch, G. (1983). Erfassung und Darstellung von Bürogeschehen. In P., Wißkirchen, T., Kreifelts, F., Krückeberg, G., Richter & G., Wurch (Hrsg.), *Informationstechnik und Bürosysteme.* Stuttgart: Teubner.

XRules (2005). XRules. Letzter Zugriff am 17.11.2008 unter http://www.xrules.org.

Yom, M. & Wilhelm, T. (2006). Intranet-Usability – Das Rezept für ein nutzungsfreundliches Intranet. Letzter Zugriff am 13.11.2006 unter http://www.contentmanager.de/magazin/artikel_1131_-usability_nutzerfreundlich_intranet.html.

Zachman, J. (1987). A framework for information systems architecture. IBM Systems Journal 26(3), 277-293. Letzter Zugriff am 10.07.2008 unter http://www.research.ibm.com/journal/sj/263/-ibmsj2603E.pdf.

Zachman J. (2008). Concise definition of the Zachman Framework, Zachman International, Letzter Zugriff am 29.09.2008 unter http://zachmaninternational.com.

Zelewski, S. (1999), Grundlagen. In C., Corsten & M., Reiß (Hrsg.), *Betriebswirtschaftslehre* (3. Aufl., S. 1-140). München: Oldenbourg.

Zelger, J. (2000). Parallele und serielle Wissensverarbeitung: Die Simulation von Gesprächen durch GABEK. In J. Zelger & R., Buber (Hrsg.), *GABEK II – Zur Qualitativen Forschung* (S. 31-91). Innsbruck: Studien Verlag.

Zelger, J. & Buber, R. (Hrsg.). (2000*). GABEK II – Zur Qualitativen Forschung.* Innsbruck: Studien Verlag.

Zelger, J., Raich, M. & Schober, P. (Hrsg.) (2008). *GABEK III – Organisationen und ihre Wissensnetze.* Innsbruck: Studien Verlag.

Zillmann, D. (2004). Emotionspsychologische Grundlagen. In R., Mangold, P., Vorderer & G., Bente (Hrsg.), *Lehrbuch der Medienpsychologie* (S. 101-128). Göttingen: Hofgrefe.

Zimbardo, P. G. & Gerrig, R. J. (2004). *Psychologie* (16. aktual. Aufl.). München: Pearson.

Zimmermann, H. (1997). Homonymanalyse, maschinelle. In H. J., Schneider (Hrsg.), *Lexikon Informatik und Datenverarbeitung* (4. aktual. u. erw. Aufl., S. 389-390). München: Oldenbourg.

Zuser, W., Grechenig, T. & Köhle, M. (2004). *Software Engineering mit UML und dem Unified Process* (2. überarb. Aufl.). München: Pearson.

Zwicky, F (2005). Fritz Zwicky Stiftung. Letzter Zugriff am 11.10.2005 unter http://www.zwicky-stiftung.ch/.

Index