Dissertation

Spezifikation und Entwicklung universitärer Lern- und Arbeitsumgebungen

Dipl. Wirt.-Inform. Alexander Roth

Schriftliche Arbeit zur Erlangung des akademischen Grades
doctor rerum politicarum (dr. rer. pol.)
im Fach Wirtschaftsinformatik

eingereicht an der
Fakultät für Wirtschaftswissenschaften der
Universität Paderborn

Paderborn, im August 2008

Gutachter:
1. Prof. Dr. Leena Suhl
2. Prof. Dr. Ludwig Nastansky

Für Anke und Henrike

Zusammenfassung

Universitäten der Wissensgesellschaft stehen derzeit unter immensen Druck, mehr Wissen und Handlungskompetenz schneller an eine größere und zunehmend inhomogener werdende Zielgruppe zu vermitteln. Daher müssen zum einen die Kernprozesse der Wissensorganisation effizienter und effektiver gestaltet werden, zum anderen müssen Lehr- und Lernprozesse die Schlüsselkompetenzen heutiger Wissensarbeiter fördern, um Studierende auf den Arbeitsmarkt vorzubereiten.

Geleitet von der Fragestellung, wie die daraus resultierenden neuen Anforderungen durch universitäre IT-Infrastrukturen unterstützt werden können, wird in dieser Thesis eine Alternative zu den klassischen monolithischen Lernplattformen aufgezeigt: Eine komponentenorientierte Systemarchitektur mit einem Kern zur Unterstützung zentraler Dienste, auf den neue Anwendungen sowohl für klassisches E-Learning als auch für selbstorganisierte und kooperative Szenarien medienbruchfrei aufgesetzt werden können.

Die vorliegende Arbeit fokussiert dabei die interdependente Spezifikation und Entwicklung der zentralen Plattform und der darauf basierenden Anwendungen. Dazu werden drei Ziele erarbeitet: (1) Definition einer kontrollierten Sprache (Normsprache) zur Verbesserung der hochschulweiten Spezifikation von Lern- und Arbeitsszenarien. (2) Konzeption einer komponentenorientierten Rahmenarchitektur, die für verschiedenste (verteilte) E-Learning-Anwendungen eine geeignete Standardplattform darstellt und die in die vorhandene universitäre IT-Infrastruktur integriert ist. (3) Konstruktion eines methodischen Vorgehens zur Realisierung von Standardplattform und darauf basierender E-Learning-Anwendungen unter Berücksichtigung des gewählten normsprachlichen Ansatzes und der konzipierten Rahmenarchitektur.

Zuletzt wird beschrieben, wie dieser Ansatz durch die ko-aktive Lern- und Arbeitsplattform koaLA an der Universität Paderborn erfolgreich in die Praxis umgesetzt wurde.

Abstract

The universities within today's knowledge society face enormous pressure to impart more knowledge and operational competence in less time to a target group that grows larger and more inhomogeneous. Therefore the core processes of knowledge organisation have to become more effective and efficient. In order to prepare students for the job market, also the methods of teaching and learning must encourage the core competencies that are necessary for today's knowledge workers.

Addressing the issue of how university IT-infrastructures can support the resulting new requirements, this thesis presents an alternative to the classic monolithic learning platforms: A component-oriented system architecture with a server core that not only facilitates central functions but which can also be complemented with new services for traditional e-learning and also for self-organised and cooperative scenarios. This thesis thereby focuses on the interdependent specification and development of this central platform and the belonging applications. For that purpose three objectives are identified: (1) Defining a controlled language that — within the whole university – improves specification of learning and working scenarios. (2) Conceptual design of a component-orientated architectural framework, which functions as a standard platform for a variety of shared e-learning applications and is also integrated into the existing university IT-infrastructure. (3) Developing a methodical approach for implementing the standard platform and the belonging applications in accordance with the chosen controlled language approach and the architectural framework.

The last part of this thesis describes the successful implementation of this approach in the cooperative learning and working platform koaLA at the university of Paderborn.

Danksagung

Die vorliegende Thesis entstand während meiner Zeit als wissenschaftlicher Mitarbeiter am DS&OR-Lab der Universität Paderborn, in der ich in verschiedenen Verbundprojekten und Arbeitsgruppen im Bereich neuer Medien in der Bildung sowie in Projekten zur IT-Infrastrukturentwicklung an Hochschulen mitgewirkt habe. Mein eigenes Projekt „Promotion“ wurde dabei durch die fachliche und freundschaftliche Unterstützung verschiedener Personen ermöglicht, denen ich an dieser Stelle dafür danken möchte.

Allen voran gilt mein Dank Prof. Dr. Leena Suhl für die Gelegenheit zur Promotion und für die Betreuung dieser Arbeit. Ihre grundsätzliche Unterstützung meines Vorhabens, die mir dazu überlassenen Freiräume und ihre stets förderlichen Anmerkungen haben entscheidend zum Gelingen beigetragen. Prof. Dr. Ludwig Nastansky danke ich für die freundliche Übernahme des Zweitgutachtens.

Ganz großer Dank gebührt Prof. Dr. Thorsten Hampel, der meine Arbeit lange Zeit durch anregende Diskussionen, Denkanstösse und auch darüber hinaus gefördert und geprägt hat. Die Zusammenarbeit mit ihm und seinem Team war mir stets ein Vergnügen.

Bedanken möchte ich mich auch bei allen Mitarbeitern des Locomotion-Projektes, die dazu beigetragen haben, die aus dieser Arbeit hervorgegangene koaLA-Lernumgebung erfolgreich zu einem hochschulweiten Angebot auszuweiten. Stellvertretend seien hier namentlich Dr. Gudrun Oevel, Andreas Brennecke und René Sprotte genannt.

In meinen Dank einschließen möchte ich auch meine aktuellen und ehemaligen Kollegen am DS&OR-Lab für die freundliche Atmosphäre und die ständige Bereitschaft zum Erfahrungsaustausch zum Thema Promotion. Vielen Dank an Dr. Markus Toschlaeger für seine wertvollen Tipps zum Aufbau der Arbeit.

Dr. Claus Biederbick und René Sprotte haben mich durch Korrekturlesen und überaus konstruktiven Anmerkungen bei der finalen Revision der Arbeit unterstützt. Für diesen nicht selbstverständlichen Freundschaftsdienst bedanke ich mich herzlich.

Besonderen Dank schulde ich meiner Familie. Meine Frau Anke hat hierfür nicht nur auf unzählige gemeinsame Stunden verzichtet, sondern mir darüber hinaus immer den nötigen Rückhalt gegeben und mich motiviert. Auch meine Mutter hat mich durch Korrekturlesen unterstützt.

Nicht nur meiner Mutter, sondern ebenfalls meinem Vater gebührt letztendlich mein größter Dank: Meine Eltern haben meine gesamte Ausbildung bis zur Promotion immerwährend und vorbehaltlos unterstützt. Ohne sie wäre all dies nicht möglich gewesen. Danke!

Paderborn, im August 2008 Alexander Roth

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AJAX	Asynchronous Javascript And XML
ANSI	American National Standards Institute
AOP	Aspektorientierte Programmierung
API	Application Programming Interface
ASP	Application Service Providing
BMBF	Bundesministerium für Bildung und Forschung
CBT	Computer Based Training
CDL	Component Definition Language
CEN	Centre de European Normalisation
CIM	Collaboration Information Management
CMI	Computer Managed Instruction
CMS	Content Management System
COM	Component Object Model
COAL	Client Object Access Layer
CoP	Community of Practice
CORBA	Common Object Request Broker Architecture
CQL	Common Query Language
CSCL	Computer Supported Cooperative Learning
CSE	Campus Source Engine
CSCW	Computer Supported Cooperative Work
CSS	Cascading Style Sheet
DCOM	Distributed Component Object Model
DIN	Deutsches Institut für Normung
DIN PAS	DIN Public Available Specification
DOM	Document Object Model
DRI	Digital Repositories Interoperability
ECTS	European Credit Transfer System
EJB	Enterprise Java Bean
ELF	E-Learning Framework
ELM	Essener-Lern-Modell
EML	Educational Modeling Language
ERM	Entity Relationship Model
EU	Europäische Union
EUR	Euro
F&E / FuE	Forschung und Entwicklung
FTP	File Transfer Protocol
GPRS	General Packet Radio Service

GSM	Global System for Mobile Communications
HDM	Hypertext Design Model
HIS	Hochschul-Informations-Systeme GmbH
HIS LSF	HIS-Portal für Lehre, Studium und Forschung
HTML	Hyptertext Markup Language
HTTP	Hyptertext Transfer Protocol
IAF	IMS Abstract Framework
ID/ISD	Instructional Systems Design
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFRS	International Financial Reporting Standard
IT	Informationstechnik
IKT / IuK	Informations- und Kommunikationstechnik
IP	Internet Protocol
ISO	International Organization for Standardization
J2EE	Java 2 Enterprise Edition
JDBC	Java Database Connectivity
JISC	Joint Information Service Committee
JNDI	Java Naming and Directory Interface
JRE	Java Runtime Environment
JVM	Java Virtual Machine
LD	Learning Design
LDAP	Lightweight Directory Access Protocol
LMOS	Learning Management Operating System
LMS	Learning Management System
LOM	Learning Object Metadata
LTSA	Learning Technology Systems Architecture
MIME	Multipurpose Internet Mail Extensions
MIT	Massachusetts Institute of Technology
MMOG	Massive Multiplayer Online Game
MOM	Message Oriented Middleware
MOO	MUD Object Oriented
MTBF	Mean Time Between Failures
MTTR	Mean Time to Repair
MUD	Multi User Dungeon
MVC	Model-View-Controller
NMB	Neue Medien in der Bildung
OAI	Object Action Interaction-Modell
ODBC	Open Database Connectivity
OKI	Open Knowledge Initiative
OO	Object Oriented
ORM	Objektrelationales Mapping
OSI	Open Systems Interconnection (Reference Model)

OSID	Open Source Interface Definition
PAPI	Public and Private Information for Learners
PKI	Public Key Infrastructure
PLE	Personal Learning Environment
QM	Qualitätsmanagement
QS	Qualitätssicherung
QTI	Question and Test Interoperability
REST	Representational State Transfer
RMI	Remote Method Invocation
RMM	Relationship Management Methodology
RPC	Remote Procedure Call
RSS	Real Simple Syndication
SaaS	Software as a Service
SCORM	Sharable Content Object Reference Model
SHDM	Semantic Hypermedia Design Method
SOA	Serviceorientierte Architektur
SOAP	Simple Object Access Protocol
SPL	Softwareproduktlinie
SQI	Simple Query Interface
SSL	Secure Socket Layer
SW	Software
SWE	Software-Engineering
TAOS	Terminologie-basierter Ansatz für die Objektorientierte Spezifikation
TBF	Time Between Failure
TTR	Time to Repair
UDDI	Universal Description, Discovery and Integration
UI	User Interface
UML	Unified Modelling Language
UMTS	Universal Mobile Telecommunications Systems
URL	Uniform Resource Locator
US-GAAP	Generally Accepted Accounting Principles in the United States
VoIP	Voice over IP
VORMS	Virtual Operations Research/Management Science
VSQL	Very Simple Query Language
WAP	Wireless Application Protocol
WBT	Web Based Training
WebDAV	Web Based distributed Authoring and Versioning
WebML	Web Modeling Language
WLAN	Wireless Local Area Network
WSDL	Web Service Description Language
WSDM	Web Site Design Method
WWW	World Wide Web
XML	Extended Markup Language
XQuery	XML Query

1. Einleitung

1.1. Szenario: Wissensgesellschaft und Hochschulbildung

Mit dem Ziel, gesellschaftlichen Wandel in große Zusammenhänge und langfristige Richtungen einzuordnen, werden in den Sozial- und Wirtschaftswissenschaften Zyklen und Megatrends beschrieben, die die treibenden Kräfte in einen gemeinsamen Kontext setzen. Nach der physischen Verkopplung menschlicher und maschineller Arbeit auf der Ebene der Elementarfaktoren (industrielle Revolution, seit 1790) sowie deren gesellschaftliche Verarbeitung (Postmoderne, seit 1940) ist in der *Informationsgesellschaft* seit 1980 eine Verkopplung des dispositiven Faktors mit der Informationstechnologie festzustellen (vgl. [Jän04a], S. 5f.): Zeitnah generierte Informationen in unternehmensinternen Datenbanken waren kennzeichnend für diese Epoche und hatten einen immer größeren Einfluss auf das Führen und Steuern von Organisationen.

Seit Mitte der 90er Jahre führt der omnipräsente Zugang zum weltweiten Datennetz Internet zu einer allgemeinen Informationsüberflutung (*quantitativen Schock* (s. [Lei01], S. 14). Beiträge aus unterschiedlichsten Quellen treten in einen internationalen, technisch vernetzten Pool von Informationsangeboten. Gerade die Idee, beliebige Wissensquellen logisch zu verknüpfen, begründet die Mächtigkeit und Qualität des Internets sowie seinen weltweiten Erfolg in vergleichsweise kurzer Zeit (vgl. [BLF99], auch [BL99]).

Durch die Aufhebung technischer Barrieren ist jeder mit Zugang zum Internet in der Lage zu Publizieren. Hierdurch wird wiederum die beschleunigte Generierung neuer, thematisch zunehmend spezialisierter Information durch eine unkontrolliert steigende Zahl von „Wissens-Schaffenden“ gefördert (vgl. [Lei01], S. 23). Während einst Intermediäre wie Hochschulen und Medien zwischen Informationsproduzenten und -konsumenten vermittelten, verschwimmen mit der *Inflation der Produzenten* die Grenzen zwischen den beiden Segmenten zunehmend. Dieses Phänomen markiert das Ende der Informationsgesellschaft, in der anfangs Informationen organisatorisch und technisch noch weitgehend isoliert und durch die Zugriffsberechtigten beherrschbar waren, und den Beginn der *Wissensgesellschaft*[1].

Die Wissensgesellschaft ist charakterisiert durch eine Aufwertung des Wissens im ökonomischen Kontext: Auf Unternehmensebene wird dieser „vierte Produktionsfaktor“ (s. [Jän04a], S. 129ff.) als die zentrale Voraussetzung für Wettbewerbsfähigkeit in einem zunehmend härter umkämpften internationalen Marktumfeld wahrgenommen (vgl. [Sch00b], S. 19, [NT01], S. 1, [Lei01], S. 6).

[1] Eingeführt wurde der Begriff der Wissensgesellschaft in der amerikanischen Soziologie als *Knowledge Society* zwar schon in den 60er Jahren (s. [Lan66], S. 650), jedoch erst in jüngster Zeit erlangte er die heutige Bedeutung. Seine Verwendung ist uneinheitlich. So wird häufig nicht zwischen Wissens- und Informationsgesellschaft differenziert, obgleich schon die Abgrenzung des Begriffs „Wissen“ gegenüber „Information“ die Problematik dieser Unschärfe deutlich macht (vgl. hierzu bspw. [PRR03], S. 15f.).

Als Ausgangspunkt ist eine *Spirale des Wissenswachstums* zu konstantieren: Gemäß der Erkenntnis, dass Wissen zu einem Wettbewerbsvorteil werden kann, erhöhen sich die Bemühungen, die neuen technischen Möglichkeiten der Kommunikation und Informationsverarbeitung zum Zwecke eines verbesserten Managements des Erfolgsfaktors Wissen auszunutzen. Dies ermöglicht ein produktiveres und effizienteres Management des Wissens, also dessen verbesserte Identifikation, Nutzung, Bewahrung, Verteilung etc.[2] Insbesondere die höhere Arbeitsteilung und Spezialisierung (vgl. [Lei01], S. 17) setzt eine große produktive Kraft frei, denn mit dem verbesserten Zugang zu Informationen durch das Fallen von Kommunikationsbarrieren positionieren sich Wissensproduzenten individueller. Der resultierende Quantitätsschub im Wissensbestand und der verfügbaren Informationen stellt wiederum einen Treiber dar für das stetige Bemühen, Organisation und Instrumente der Informationsverarbeitung zu verbessern, u.s.w. Diese Dynamik des Wissenswachstums senkt die zeitliche Dauer, in der Erkenntnisse noch als relevant und aktuell angesehen werden (verringerte *Halbwertszeit des Wissens*).

Daher ist die Spirale des Wissenswachstums im Unternehmensbereich gleichbedeutend mit einem radikal gestiegenen Innovationstempo und -druck. So wird behauptet, ein *Internetjahr* entspreche im Ergebnis drei Jahren in der Industriegesellschaft (vgl. [Lei01], S. 11). Dies impliziert verkürzte Produktzyklen und einen zunehmenden *Time-to-Market-Druck* im globalen Wettbewerb. Für größere US-Unternehmen ist eine durchschnittliche Rate von einem neuen Produkt pro Tag mittlerweile Standard (vgl. [Tap97], S. 60). Der Bedeutung von Wissen im Bereich Produktinnovation tragen auch die internationalen Bilanzierungsregeln (US-GAAP, IFRS) Rechnung. Hier sind die Forschungs- und Entwicklungsausgaben als Vermögenswerte zu bilanzieren. Seit 2005 ist dies auch für in Deutschland sitzende, börsennotierte Konzerne bindend, deren Ausgaben für FuE-Aktivitäten das Budget der öffentlich geförderten Forschung um ein Vielfaches übersteigen (vgl. [BMB04a], S. 175, [Lei01], S. 19).

Spezifisch für die Unternehmen im Zeitalter der Wissensgesellschaft ist weiterhin eine zunehmende *Prozessorientierung*. Der seit vielen Jahrzehnten (vgl. [Jän04a], S. 8) zu beobachtende Trend zur internationalen Verflechtung wirtschaftlicher Aktivität findet in elektronischen Netzwerken einen bedeutenden Katalysator: Die IT-gestützte, grenzüberschreitende Kommunikation ohne signifikanten Zeit- und Kostenaufwand reduziert die Transaktionskosten, insbesondere für Abstimmung, Absicherung, Planung und Abschluss von Geschäften. Die Informations- und Kommunikationstechnik (IKT) erleichtert es international wettbewerbsfähigen Unternehmen, sich auch in ausländischen Absatzmärkten zu positionieren.

Die Existenzberechtigung statischer Strukturen der Industriegesellschaft wird durch diesen Wandel in Frage gestellt (vgl. [Hau02], S. 6). Hiervon betroffen sind neben den Unternehmen auch die Prozesse innerhalb der Betriebe. Während in der Industriegesellschaft der eigenständigen Optimierung aller unternehmensinternen Abläufe noch eine zentrale Aufmerksamkeit zukam, sind Betriebe nun gezwungen, sich auf ihre Kernprozesse zu konzentrieren (vgl. [Jän04a], S. 1) und über Outsourcing günstigere Dienstleistungen und

[2] Nach den Kernprozessen des Wissensmanagements, vgl. [PRR03], S. 28.

Güter von Dritten in ihre Wertschöpfungskette zu integrieren (vgl. [Jän04a], S. 7). Die damit verbundene *Fragmentierung der Unternehmensprozesse* ist zu einem prägenden Attribut der Wissensgesellschaft geworden. Ergebnisorientierte *Meta-Information* zu einzelnen Prozessen (Zeit-, Mengen-, Qualitäts- und Kostenziffern für Input und Output) rücken in diesem Kontext für die Entscheidungsträger in den Vordergrund.

Diese Prozesssicht führt mehr und mehr zu einer *Abstraktion vom ausführenden Mitarbeiter*, dessen individuelle Arbeitsleistung austauschbarer wird (vgl. [FH96], S. 240). Zunächst nicht davon betroffen sind flexible, hochqualifizierte Angestellte (vgl. [FH96], S. 242). Die Folge ist ein Auseinanderklaffen der Schere zwischen solchen *Knowledge Workern* einerseits und ausführenden Mitarbeitern andererseits. Nach [Sch05], S. 1, sehen sich jedoch auch zunehmend Knowledge Worker dem Trend zum *Offshoring*[3] ausgesetzt: „offshoring moves higher up the skills ladder“. So sei mittlerweile auch zu beobachten, dass auch FuE-Leistungen – gemeinhin den klassischen Kernprozessen der Unternehmen zugerechnet – geographisch ausgelagert (Off-Shoring, z. B. [SAP02]) oder gar vollständig externalisiert und international eingekauft werden (Outsourcing, vgl. [Sch05], S. 5).

Die neuen Herausforderungen der Wissensgesellschaft verstärkend, wird durch die hohe Mobilität des Wirtschaftsfaktors Wissen der Wettbewerb der nationalen Standorte auch zu einem *Wettbewerb um soziales Kapital* (vgl. [Sch05], S. 6). Dies gilt insbesondere innerhalb des Kreises hoch entwickelter Industrienationen, wo niedrige Geburtenraten langfristig als Katalysatoren dieser Entwicklung wirken. Als Beispiel für die Folgen des Wettbewerbs um soziales Kapital kann der Exodus wissenschaftlichen Talents in Richtung USA angeführt werden (transatlantischer *Brain Drain*, vgl. [Bue01], S. 7).

Der Intensivierung des Wettbewerbs auf dem Arbeitsmarkt Rechnung tragend, werden individuelle und private – allgemein formuliert: dezentrale – Bildungsstrategien bedeutsamer, während die zentrale Definition von Curricula zunehmend schwieriger wird. Dies ist nicht zuletzt dadurch bedingt, dass sich das Generieren von Wissen wie beschrieben zunehmend von öffentlichen in den privaten Bereich verlagert. Diese Privatisierung der Bildung i. w. S. ist ein Katalysator für das Entstehen eines *Knowledge Gap* „zwischen wissensnahen (sozial bevorteilten) und wissensfernen (sozial deprivierten) Gruppen“, s. [Rog04], S. 54).

Als Gegensatz zur Kontinuität einstiger Erwerbskarrieren (Lebensberufe) in der Agrar- und Industriegesellschaft ist die Halbwertszeit eines typischen Berufsprofils in der Wissensgesellschaft ebenso kurz wie die Anforderung im Alltag des Berufsinhabers kurzlebig sind: Vor dem Hintergrund wissensbasierter Tätigkeiten wird der Mitarbeiter zum Wissensarbeiter, der in der Lage ist, in seinem Arbeitsumfeld seine Aufgaben selbst zu identifizieren und zu organisieren, relevantes Wissen zu kommunizieren und anzuwenden (vgl. [Hei03]). Während früher die Allgemeinbildung im Vordergrund stand, werden heute also die spezifischen Kompetenzen der Wissensarbeiter nachgefragt (vgl. [BLK03] und [Dav05], S. 25ff.):

[3] Der Begriff Offshoring steht für eine Form der Auslandsverlagerung unternehmerischer Funktionen und Prozesse aufgrund dort vorherrschender günstigerer Rahmenbedingungen wie Arbeitskosten.

Handlungskompetenz Im Einzelnen umfasst die Handlungskompetenz die Teilkompetenzen Fach-/Methodenkompetenz, Personalkompetenz und Sozialkompetenz (vgl. [BLK01], S. 20, [Rau04], S. 8ff.), wobei die Fach-/Methodenkompetenz die Fähigkeit darstellt, erlerntes Fachwissen zielorientiert zu nutzen. Die Personalkompetenz hingegen formuliert die Ebene von ethischen Werten, Selbständigkeit und Kritikfähigkeit. Der Bereich der Sozialkompetenz umfasst die Fähigkeit, in einem sozialen Gefüge zu agieren, Verantwortung zu übernehmen und sich solidarisch zu verhalten. Somit bildet die Handlungskompetenz eine Grundlage für ein Individuum, in beruflichen, gesellschaftlichen und privaten Situationen handlungsfähig zu sein.

Medienkompetenz Eingebettet in den gesetzten Rahmen der übergeordneten Handlungskompetenz wird die Medienkompetenz beispielhaft von Bönkost für die Nutzung von digitalen Technologien durch vier Qualifikationen definiert: Medienkritik, Medienkunde, Mediennutzung und -gestaltung (vgl. [Bön04] und [Baa99], S. 34). Während die Medienkritik und die Medienkunde beide die Ebene der Vermittlung betrachten, fokussieren die Dimensionen der Mediennutzung und Mediengestaltung die Ebene der Zielorientierung im Rahmen der Lern- und Methodenkompetenz, die für die Initiierung und Ausführung von Handlungen der Individuen verantwortlich ist (vgl. [Baa99], S. 34).

Lernkompetenz Darüber hinaus bedarf es aber auch einer Handlungskompetenz für den Lernprozess, um das Lernen in einer Wissensgesellschaft langfristig gewährleisten zu können. „Lernen erfordert zum einen selbstgesteuerte, aktive Wissenskonstruktion und ist zum anderen ein sozialer, interaktiver Prozess“ [MK01], S. 10f. Daher ist neben der Sozialkompetenz und der Medienkompetenz die Kompetenz zur Selbststeuerung zentral. Hierfür sind vor allem metakognitive Fähigkeiten erforderlich, um (1) das Lernen vorzubereiten, (2) die Lernhandlung durchzuführen, (3) das Lernen zu regulieren, (4) die Lernleistung zu bewerten und (5) die Motivation und Konzentration aufrechtzuerhalten (vgl. ebenda, S. 11).

Kompetenz zum Wissensmanagement Die Kompetenz zum Wissensmanagement formuliert die Fähigkeit, die derzeit stetig wachsenden Informationsfluten nach Inhalt, Bedeutung und Nutzen zu selektieren, zu bewerten und für die Wissensbildung nutzbar aufzuarbeiten (vgl. [RRM97], S. 194f.). Durch eine Integration von Arbeits- und Lernprozessen schafft ein individuelles Wissensmanagement den Rahmen für eine pragmatische Wissensarbeit, die es dem Individuum ermöglicht, Expertise zu erlangen. Die Kombination von *Wissenswerkzeugen* und der Fähigkeit, diese zielorientiert einzusetzen, hat daher für den Einzelnen ein gewaltiges Potenzial (vgl. [Jän04a], S. 6; vgl. auch [HM97], S. 11 und [RRM97], S. 61).

Unsicherheit bezüglich dessen, was zukünftig darüber hinaus an spezifischen Kompetenzen gefragt sein wird, ist ein inhärentes Merkmal der Wissensgesellschaft. Diese Unsicherheit impliziert für den Einzelnen die permanente Notwendigkeit, den Status des Lernenden niemals zu verlassen und sich dem *lebenslangen Lernen* (vgl. [BLK01], [Lei01], S. 16) zu verschreiben. Gesellschaftliche Teilhabe und Chancengleichheit kann daher zukünftig

nicht mehr allein durch ein staatliches Bildungssystem gesichert werden; das Individuum selbst ist für den Erwerb der in der Wissensgesellschaft erforderlichen Kompetenzen verantwortlich (vgl. [RRM01], S. 11).

Trotzdem führt die Notwendigkeit des lebenslangen Lernens zu einem grundsätzlich veränderten Verständnis von institutioneller Bildung, da es zum Grundprinzip werden muss, an dem sich Angebot und Nachfrage in sämtlichen Lernkontexten ausrichten (vgl. [KdE00], S. 308, [Keh01], S. 125). Dabei muss Bildung, die im primären Bildungssektor der Schulbildung, aber vor allem im sekundären und tertiären Bildungssektor der Aus-, Fort-, und Weiterbildung gefördert wird, den Lernenden dazu befähigen, am öffentlichen Leben partizipieren zu können. Das Bildungsziel sollte also eine fachübergreifende Lernkompetenz sein (vgl. [MK01], S. 4). Dafür werden neue Lehr- und Lernformen und neue Wissensinhalte benötigt, die „stärker als je zuvor von externen Themen, Fragestellungen und Problemlösungsmöglichkeiten bestimmt sind und nicht mehr den konventionellen Paradigmen akademischer Disziplinen und Fachkulturen entsprechen“ [Keh01], S. 125. Dies impliziert auch konzeptuelle Veränderungen bei der Hochschulbildung, die in ihrer traditionellen Form oftmals zu statisch und unflexibel ist, um schnell auf neue Arbeitsanforderungen und veränderte Entwicklungen reagieren zu können (vgl. [KdE00], S. 308). In diesem Kontext wird der Hochschule als „[...] Teil der Aus- und Weiterbildung zum einen eine neue Bedeutung als Stätten der Wissensproduktion und -distribution beigemessen“ (vgl. [Keh01], S. 123), zum anderen sollen durch die Hochschulbildung in stärkerem Maße als zuvor übertragbare Fähigkeiten und Fertigkeiten vermittelt werden, die für unterschiedlichste Tätigkeitsbereiche Schlüsselkompetenzen herstellen. Diese neue Anforderung an Hochschulbildung geht deutlich über die bisherige Vermittlung von disziplinär strukturierten (Fach-)Wissen und den dazu gehörigen Methodenkenntnissen hinaus (vgl. [Keh01], S. 126).

Im bildungspolitischen Kontext greift die *Bologna-Erklärung*[4] das Thema „Lebenslanges Lernen“ explizit auf, wonach Universitäten im europäischen Bildungsraum offener gestaltet werden sollen: Allen Altersklassen und jedem Vorbildungsstand soll Zutritt zu universitärer Bildung ermöglicht werden. Dazu sollen bis zum Jahre 2010 neue Studienstrukturen durchlässiger und beweglicher als bisher gestaltet werden, im biographischen Bereich, wie auch im strukturellen Bereich[5].

[4] Auf der Grundlage einer Vereinbarung des Jahres 1998 (Sorbonne-Erklärung) zwischen den Bildungsministern Frankreichs, Deutschlands, Italiens und Großbritannien erwuchs ein Jahr später eine Erklärung der Bildungsminister, die von Vertretern aus 29 europäischen Ländern am 19. Juni 1999 in Bologna unterzeichnet wurde. Die Vorbereitung und Umsetzung dieser Erklärung wird als *Bologna-Prozess* bezeichnet.

[5] Im Einzelnen sollen sechs Aktionsschwerpunkte umgesetzt werden (vgl. [LW05], S. 7): (1) Einführung eines Systems leicht verständlicher und vergleichbarer Hochschulabschlüsse, (2) Einführung eines Systems, das sich im Wesentlichen auf zwei Hauptzyklen (Bachelor/Master) stützt, wobei bereits der erste Zyklus eine für den europäischen Arbeitsmarkt relevante Qualifikationsebene attestiert, (3) Einführung eines Systems zur Akkumulierung und zur Anrechnung/Übertragung von Studienleistungen (*European Credit Transfer Systems*, ECTS), (4) Mobilität von Studierenden, Lehrkräften und Forschern, (5) Zusammenarbeit bei der Qualitätssicherung und (6) eine europäische Dimension der Hochschulbildung.

Vor diesem Hintergrund entstehen völlig neue Anforderungen an Bildungseinrichtungen, denen es sich zu stellen gilt: Es muss mehr Wissen und Handlungskompetenz schneller an eine größere und zunehmend inhomogener werdende Zielgruppe vermittelt werden. Daher muss zum einen Wissen effizienter und effektiver vermittelt werden, zum anderen müssen Lehr- und Lernprozesse den o. a. Anforderungen gerecht werden.

1.2. Motivation der Arbeit

Neue Medien können die Hochschullehre nachweislich verbessern (vgl. [Bau03a] und [RB04]). Lange Zeit wurde ihr Einsatz mit einem Anspruch auf höchste Produktqualität verbunden, die Alltagstauglichkeit jedoch vernachlässigt. Beispielsweise wurde der in der E-Learning-Szene renommierte Wissenschaftspreis Medidaprix bislang fast ausschließlich an Projekte vergeben, die ein gutes didaktisches Modell medial sehr aufwändig umgesetzt haben. Ein Vergabekriterium war u. a. die isolierte Lauffähigkeit. Prozesse der Organisationsentwicklung an Hochschulen wurden nicht berücksichtigt (vgl. [BP04])[6]. Solche abgeschlossenen virtuellen Lernarrangements von besonders hoher Qualität befriedigen jedoch nicht die Bedürfnisse einer alltagstauglichen, unproblematischen Nutzung von Medien an deutschen Hochschulen, bei der das Kosten-Nutzen-Verhältnis und die Einbindung in die jeweils vorhandene IT-Infrastruktur im Vordergrund stehen.

Angebracht ist vielmehr eine Abkehr von der Produktorientierung einzelner Lehrstühle und Fachbereiche hin zu einer Dienstorientierung für eine bedarfsgerechte Mediennutzung für eine ganze Hochschule. Dies erfordert die Implementierung zentraler Unterstützungsprozesse und Basisdienste, die hochschulweit in fachbereichsspezifische Prozesse und Plattformen eingebettet werden können.

Neben den großen Strukturveränderungen, die durch die Bologna-Reform induziert sind, geht es derzeit und zukünftig also ebenfalls darum, „die Leistungsstrukturen von Hochschulen mit Hilfe eines bedürfnisorientierten Medieneinsatzes ebenso behutsam wie konsequent zum Nutzen aller Beteiligten sukzessive umzumodeln“ [Ede02], S. 19. Zumindest dann, wenn man die vorhandene didaktische Vielfalt nicht durch den Kauf einer zentralen Lernplattform eindämmen will, sondern E-Learning als Unterstützung individueller Lehr- und Lernprozesse versteht, das – wenn es gut umgesetzt wird – durchaus eine strategische Relevanz erlangen kann[7].

Aus technischer Sicht bedarf es dazu einer Infrastruktur, die es erlaubt, individuelle Lernwerkzeuge medienbruchfrei in die Kernprozesse der Wissensorganisation und der Modul- und Prüfungsverwaltung zu integrieren, um damit zum einen den Aufwand der Dozierenden für eine effektive Computerunterstützung der Lehre minimieren zu können, und zum anderen durch Systemkonvergenzen didaktische Mehrwerte zu schaffen, die durch isolierte Lernwerkzeuge nicht umgesetzt werden können. Die Lernwerkzeuge

[6] Der Medidaprix teilte sich erst zwischen 2005 und 2008 in zwei Preiskategorien „Digitale Medien in der Hochschullehre“ und „Hochschulentwicklung mit Digitalen Medien“. Für 2008 wurden die beiden Projektlinien zusammengeführt.

[7] Zum Beispiel der Einsatz von E-Learning zur Wissenschaftsentwicklung (E-Learning als Forschungsgegenstand), zur Schaffung eines flexiblen Zugangs zu hochschulischen Bildungsangeboten für Externe, zur Stärkung der (inter-)nationalen Wettbewerbsfähigkeit etc. (vgl. [Hop05a], S. 108ff.).

müssen dazu über einen gemeinsamen Dienstekern, der technischen Werkzeugen als Basis und Integrationsplattform dient, zentrale Funktionalität unterstützen (vgl. ebenda). Hierzu müssen Probleme der Wiederverwendbarkeit gelöst werden, die bei der Entwicklung monolithischer Lernplattformen bislang nicht angegangen wurden.

Darüber hinaus verlangt es die Loslösung von der an einzelnen und isolierten Systemen orientierten Herangehensweise in der Entwicklung von E-Learning-Komponenten. Diese sollte zwischen und innerhalb der einzelnen Funktionsbereichen (Lehre, Verwaltung, Forschung) abgestimmt sein, bspw. im Rahmen eines institutionalisierten hochschulweiten Architekturmanagements, welches nicht nur eine Technologiestrategie impliziert, sondern auch ein definiertes Vorgehensmodell für die Entwicklung und Erweiterung der IT-Infrastruktur durch E-Learning-Werkzeuge.

Die Transformation von einer verteilten kooperativen IT-Versorgung hin zu einer integrierten Informationsversorgung an Hochschulen bedingt demnach eine strukturierte Herangehensweise, die sowohl strategische als auch organisatorische und softwaretechnische Aspekte umfasst.

1.3. Problembeschreibung und Zielsetzung der Arbeit

Drei wichtige Herausforderungen hochschulweiter E-Learning-Infrastrukturen sollen an dieser Stelle näher beschrieben werden, um daran die konkrete Zielsetzung dieser Arbeit auszurichten.

Verbesserung der Kommunikation Eine Abstimmung der Medienentwicklung zwischen und innerhalb der Fach- und Funktionsbereiche einer Hochschule funktioniert nur dann gut, wenn die Kommunikation zwischen Anwendern, Softwareentwicklern, Mediendesignern und Didaktikern nicht gestört ist (vgl. bspw. [GS96], S. 3, [PR04], S. 288f., [FP04]). Eine häufig auftretende Störung liegt bspw. in den unterschiedlichen Blickwinkeln bzw. unterschiedlich interpretierten Begriffen, die von diesen Gruppen zur Formulierung von Anforderungen benutzt werden (vgl. [Ort95], S. 148f., [Hel97], S. 90f.). So kann zum Beispiel ein einfaches aber elementares organisatorisches Konzept wie eine „Veranstaltung“ an verschiedenen Fachbereichen oder in verschiedenen Prüfungsordnungen eine grundlegend unterschiedliche Bedeutung haben.

Doch nicht nur die unterschiedlichen Intentionen fachlicher Bezeichnungen bedeuten eine Störungsursache. Insbesondere der Bruch zwischen der präformalen Fachsprache der Anwender und der zur Spezifikation und Entwicklung genutzten formalen Sprachen stellt eine Quelle für Missverständnisse dar, da die vom Entwickler häufig verwendeten künstlichen Sprachen und Diagrammmethoden von den Anwendern nicht gelesen und ein gemeinsames Verständnis von Sachverhalten somit nur schlecht überprüft werden kann. Die Systemanalyse ist demnach „nur ein Durchgangsmedium, denn primäres Ausdrucksmittel unserer Gedankenwelt sind Begriffe“ [Hel97], S. 90. Das eigentliche Problem, das durch das Abbilden der Fachsprachen auf formale Beschreibungssprachen nach dem Konzept einer Interpretationssemantik (*Verbalisierung*, vgl. [Ort98]) entsteht, ist der Mangel an einer objektorientierten Spezifikation von Fachkonzepten auf der

Grundlage einer Rekonstruktion von allen gemeinsam verständlichen Fachbegriffen bzw. -konzepten. Entwicklungsergebnisse können somit zwar immer noch im Hinblick auf ihre Struktur überprüft werden, jedoch nicht mehr auf ihre Anforderungsgerechtigkeit (vgl. ebenda). Daraus resultiert eine im universitären E-Learning häufig anzutreffende Prototyp-orientierte Softwareentwicklung, die sehr zeitintensiv ist. Eine bessere Lösung stellt jedoch eine normsprachliche Softwareentwicklung dar, bei der bereits für die Anforderungsbeschreibung gemeinsam verstandene und rekonstruierte Fachbegriffe verwendet werden, die eine eindeutige Zuordnung zu den in der Entwicklung verwendeten Fachobjekten besitzen (vgl. [Hel97], [Sch97a], [Ort00], [HR05], [RH05]).

Erstes Ziel dieser Arbeit ist daher die Konstruktion einer kontrollierten Sprache zur Spezifizierung universitärer Lern- und Arbeitsumgebungen, bestehend aus einer Sammlung von Fachbegriffen (Terminologie) und einer eingeschränkten Grammatik (Satzbauregeln). Diese so genannte Normsprache muss alle notwendigen Aspekte der Anforderungsbeschreibung formal korrekt abdecken können.

Komponentenorientierte Entwicklung Die technischen Realisierungen eines integrierten Informationsmanagements an Hochschulen basieren bereits zunehmend auf dem Konzept einer serviceorientierten Architektur (vgl. [MMF07], S. 19). Erste Implementierungen fokussieren dabei insbesondere die Integration von Lernmanagement in institutionale Kontexte über offene Schnittstellen (siehe S. 46; vgl. hierzu auch [DEH+02], [BCG03], [XYES03], [RSS04], [RHG05], [GR07], [JHM07]). Dabei wird oftmals das softwaretechnische Prinzip der Refaktorierung (*Refactoring*, vgl. [FBB+99] u. [KZ02]) angewandt, also bestehende Systemstrukturen weiterentwickelt und z. B. vorhandene Schnittstellen so transformiert, dass sie den Standards der Komponentenarchitektur entsprechen und somit in die serviceorientierte Architektur integriert werden können[8]. Diese Lösung ist für den Übergang hin zu einer dienstorientierten IT-Infrastruktur sicherlich geeignet. Für neu zu entwickelnde virtuelle Lern- und Arbeitsumgebungen stellt dieser Weg – also die Entwicklung als monolithisches System und spätere Transformation von Schnittstellen – aber eine teure und schwer wartbare Lösung dar, weil eine systemübergreifende Semantik mit internen Systemstrukturen im Einklang gehalten werden muss.

Neue Lerntechnologien sollten dann besser von Grund auf komponentenorientiert entwickelt werden, um eine leichte Wiederverwendbarkeit in verschiedenen Szenarien gewährleisten zu können.

Dies bedingt allerdings neben dem Einhalten von Standards ein ingenieurmäßiges Vorgehen bei der Softwareentwicklung von Lehr-/Lernsystemen, dass im Allgemeinen bislang nicht üblich ist. Untersuchungen von Harrer und Martens haben diesbzgl. gezeigt, dass komplexere Lehr-/Lernsysteme häufig entsprechend der Fachdomäne oder orientiert

[8] Harrer diskutiert die Technik des Refaktorierens im Kontext von Lehr-/Lernsystemen in [Har03]. Bazijanec et al. beschreiben in [BGKT07], S.47ff., die damit verbundenen Herausforderungen aus organisatorischer Sicht. Demnach kann es den Verantwortlichen dezentral betriebener Systeme letztendlich sogar zu negativen Anreizen im Hinblick auf eine Integration kommen, wenn trotz dem Mehraufwand der Refaktorierung Funktionen einzelner Systeme wegfallen, wenn diese nicht in ein Gesamtkonzept übernommen werden.

an einem kognitiven/pädagogischen Ansatz konzipiert werden, ohne softwaretechnischen Anforderungen wie Architekturdetails, Systemschnittstellen oder Erweiterbarkeit Rechnung zu tragen (vgl. [HM05], S. 179). Diese werden zugunsten von kurzfristigen Projektzielen und isolierten Softwareprodukten vernachlässigt, was zu Problemen und Kosten bei der Entwicklung bzw. Weiterentwicklung führt (vgl. hierzu auch [Böl02], S. 10):

- Die Entwicklung von E-Learning-spezifischen Funktionen beginnt im Wesentlichen wieder bei Null. Funktionalität, die in anderen Lernsystemen bereits vorhanden ist, wird wieder neu implementiert. Die Folge sind lange Entwicklungszeiten und hohe Kosten.

- Der Beginn bei Null führt auch dazu, dass der Aufwand, der für die Neuimplementierung von Basisfunktionen eingesetzt werden muss, an anderen Stellen oftmals fehlt, z. B. für die Qualitätssicherung. Mangelhafte Dokumentation, wiederkehrende Fehler in Basisfunktionen und – wenn überhaupt – nur unzureichende Tests sind das Ergebnis, so dass die Qualität einer Software häufig erst in der zweiten oder dritten Version einen akzeptablen Stand erreicht.

- Im Lebenszyklus einer Software kann sich dessen Umfeld mehrmals verändern, so dass der Bedarf an Wartung bzw. Weiterentwicklung gegeben ist. Bei der Neuentwicklung eines Systems sollte also dessen leichte Wartbarkeit und Erweiterbarkeit ein wichtiges Kriterium sein. Dies erfordert jedoch einen höheren konzeptuellen Aufwand, der in der Praxis häufig zu Gunsten kurzfristiger Projektziele vernachlässigt wird. Hohe Kosten für Wartung und Weiterentwicklung sind dann „vorprogrammiert“.

Es stellt sich damit die Frage, warum die Entwicklung von Lernsystemen im universitären Kontext nicht „baukastenorientiert“ auf Basis von Fachkomponenten[9] erfolgt. E-Learning-spezifische Funktionen können – in Komponenten gekapselt – bei der Entwicklung neuer Lernumgebungen wiederverwendet werden, was Zeit und Kosten einer Neuentwicklung verringern und somit Ressourcen freigeben würde für Qualitätssicherung, konzeptuelle Vorüberlegungen und zur Erstellung eines domänenspezifischen Modells zur Kommunikation und Anforderungsanalyse. Die normsprachliche Entwicklung von Komponenten auf Grundlage eines domänenspezifischen, semantischen Modells garantiert darüber hinaus einen deutlich höheren Integrationsgrad[10].

In der Praxis scheint die Wiederverwendung von Systemen, Systemkomponenten oder Systemmodellen im E-Learning bislang jedoch kaum geglückt (vgl. [HM05]). Die Ursache hierfür kann zum einen daran liegen, dass es Kosten verursacht, Projektergebnisse parametrisierbar und flexibel zu halten. Insbesondere in universitären Organisationsstrukturen mit bislang häufig fehlenden Verrechnungsmöglichkeiten für intern erbrachte Dienstleistungen (vgl. [FB06], S. 66) scheint erst einmal unklar, wer für diese Kosten aufkommen soll.

[9] In der Literatur finden sich viele Bezeichnungen, z. B. *Business Objects*, Anwendungselemente oder *Application Objects*. Im weiteren Verlauf dieser Arbeit wird jedoch der Begriff Komponente oder Fachkomponente verwendet. Eine genauere Definition folgt in Kapitel 4.

[10] Zur Forderung nach *semantischer Angemessenheit* in Komponentenarchitekturen schreibt Frank in [Fra94], S. 28: „Keine Komponente sollte genötigt sein, die Semantik von [empfangenen] Daten mühsam und unter Risiko zu rekonstruieren. Je mehr ein System dieser Forderung genügt, desto höher ist sein Integrationsniveau“.

Zum anderen ist die Berücksichtigung der Wiederverwendbarkeit im Softwareentwurf generell auch eine konzeptuelle Herausforderung: Entscheidend ist die Wahl des richtigen Abstraktionsgrades einer Systemkomponente, damit sie für unterschiedliche Einsatzszenarien passend konfiguriert werden kann[11]. Darüber hinaus besteht in heterogenen Architekturen auch die Gefahr des so genannten *architectural mismatch*, wenn trotz des komponentenorientierten Ansatzes die gewählte Systemarchitektur mit der Architektur der zu nutzenden Komponente nicht harmoniert (vgl. [GAO95])[12]. Ein weiteres Hemmnis ist sicherlich das derzeitige Fehlen eines externen Komponentenmarktes.

Erklärtes zweites Ziel dieser Arbeit ist daher die Verbesserung des softwaretechnischen Vorgehens bei der Entwicklung universitärer Lern- und Arbeitsumgebungen. Zu diesem Zweck soll eine Rahmenarchitektur für komponentenbasierte virtuelle Lern- und Arbeitsumgebungen spezifiziert werden, die dem besonderen universitären Umfeld gerecht wird. Die zu spezifizierende Rahmenarchitektur muss dabei technologie- und plattformunabhängig sein, damit von ihr in der Praxis konkrete Architekturen abgeleitet werden können. Um einen Leitliniencharakter zu gewährleisten, sollen notwendige Komponenten der E-Learning-Infrastruktur, deren Aufgaben, Zusammenwirken und Systemgrenzen beschrieben werden sowie die generelle Integration in das universitäre Umsystem.

Strukturiertes Vorgehen Die Erweiterung einer konkreten komponentenorientierten Architektur um darauf aufbauende Lern- und Arbeitswerkzeuge ist eine wiederkehrende Aufgabe an einer Universität der Wissensgesellschaft. Insbesondere aus ökonomischen Gründen, aber auch im Sinne einer verbesserten Transparenz und Prozess- und Softwarequalität, sollte diese Aufgabe methodisch fundiert angegangen werden.

Aus diesem Grund ist das dritte Ziel dieser Arbeit die Konstruktion eines entsprechenden methodischen Vorgehens, das sowohl die o. g. normsprachliche Anforderungsbeschreibung als auch die komponentenorientierte Umsetzung im Sinne der zu spezifizierenden Rahmenarchitektur als ganzheitlichen Ansatz umfasst.

Abbildung 1.1 fasst die Nutzenpotenziale des hier skizzierten systematischen Ansatzes aus softwaretechnischer Sicht noch einmal zusammen.

[11] Der Abstraktionsgrad kann auf zwei Arten gewählt werden (vgl. [Böl02], S. 12): (1) Parameter und Abstraktionen können nach bestem Wissen spekulativ in die Komponenten und Frameworks eingebaut werden. Dabei besteht aber immer die Gefahr, wichtige Parameter und Abstraktionen vergessen zu haben bzw. zu viele unwichtige Parameter und Abstraktionen eingebaut zu haben, was eine Wiederverwendung verkompliziert. (2) Parameter und Abstraktionen können bei Bedarf eingebaut werden. Dieser Ansatz impliziert allerdings häufig umfangreiche Änderungen an der Architektur von Frameworks und Komponenten und kann daher zur raschen Degeneration führen.

[12] Durch die Öffnung von Komponentenschnittstellen als Dienst wird dieses Argument abgeschwächt. Allerdings kann es auch in SOAs zu Inkompatibilitäten kommen, z. B. durch die Verwendung unterschiedlicher Protokolle wie REST oder SOAP oder verschiedener Message-Bus-Systeme.

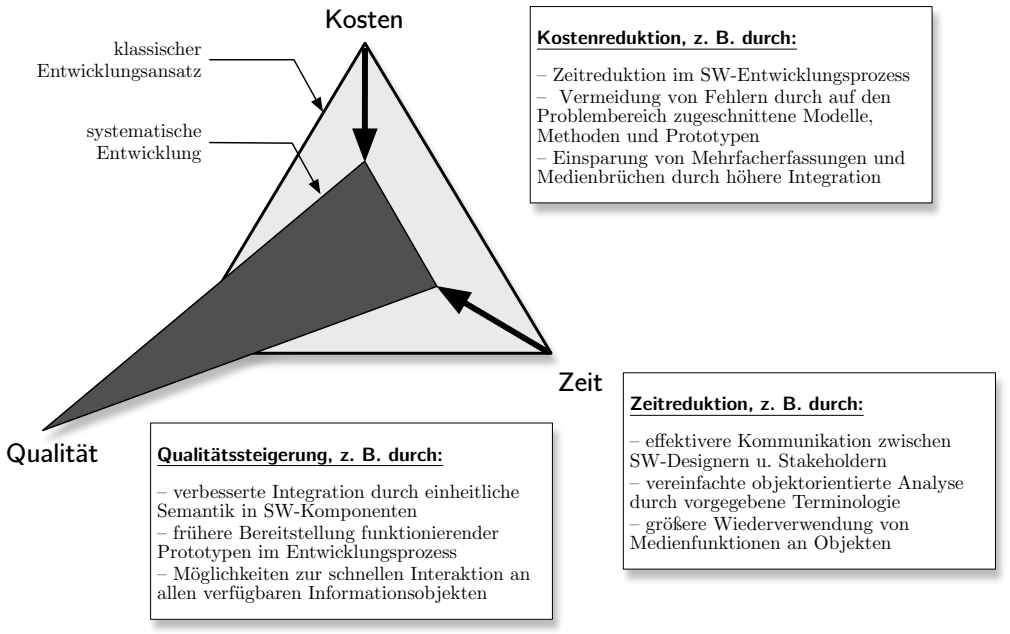

Abbildung 1.1.: Nutzenpotenziale eines systematischen Ansatzes in der Praxis

1.4. Aufbau der Arbeit

Der Hauptteil der Arbeit gliedert sich in acht Kapitel. Der Grundaufriss besteht aus einem Grundlagenteil (Kapitel 2), einem Hauptteil (Kapitel 3-5) und einem Schlussteil (Kapitel 6-7).

Im Anschluss an das vorliegende einleitende erste Kapitel wird im zweiten Kapitel der Problembereich und sein Umsystem näher erläutert. Dabei stehen das universitäre E-Learning zusammen mit der organisatorischen und technischen IT-Infrastrukturentwicklung an deutschen Hochschulen im zentralen Fokus. Ihre Einflussfaktoren sind zum einen die aktuellen Trends im E-Learning, zum anderen die Evolution des Internets („Web 2.0“). Beide Einflussrichtungen werden beschrieben, Trends werden herausgestellt und ihre Auswirkungen auf das universitäre E-Learning erläutert. Aus Gründen der besseren Lesbarkeit werden die theoretischen Grundlagen des Lösungsansatzes – terminologiebasierte Softwareentwicklung, komponentenorientierte Architekturen und Methoden-Engineering – nicht in Kapitel 2 mit aufgenommen, sondern in den entsprechenden Kapiteln vorgestellt, in denen sie auch weiter ausgeführt werden.

In Kapitel 3 soll das erste Teilziel erarbeitet werden: Nach einer theoretischen Fundierung zu Normsprachen und terminologiebasierter Softwareentwicklung wird die Metapher virtueller Wissensräume als semantisches Bezugssystem vorgestellt, auf das der in dieser Arbeit beschriebene Softwareentwicklungsprozess beruht. Die Nutzung der Terminologie zur Spezifizierung und Implementierung von Lehr-/ Lerntechnologien wird im dritten Unterkapitel erklärt.

Das zweite Teilziel, die Spezifizierung einer Rahmenarchitektur für komponentenbasierte Lehr-/Lernumgebungen, ist Thema des Kapitels 4. Zur theoretischen Fundierung wird zuerst der Architekturbegriff erklärt, um diesen dann im Kontext universitärer Informationsarchitekturen weiter auszuführen. Die gemeinsame Betrachtung von der Entwicklung wiederverwendbarer Komponenten und der Entwicklung von darauf basierenden konkreten Anwendungen in einem Vorgehensmodell wird aus technischer Sicht konkretisiert. Letztendlich stellt die Spezifizierung einer solchen Architektur für universitäres E-Learning den Hauptteil des Kapitels, indem benötigte Komponenten und das Zusammenspiel von Komponenten beschrieben werden.

Um eine Systematik bei der Softwareentwicklung sowohl auf Basis der Wissensraummetapher als auch auf Basis eines Produktfamilien-basierten Ansatzes gewährleisten zu können, wird in Kapitel 5 eine entsprechende Methode konstruiert. Im Wesentlichen wird ein Vorgehensmodell hergeleitet und das Metamodell der Methode sowie das dazu gehörende Rollenmodell und eine Techniksammlung beschrieben.

In Kapitel 6 wird die praktische Anwendung des in dieser Arbeit erarbeiteten Ansatz am Beispiel der an der Universität Paderborn hochschulweit eingeführten Lern- und Arbeitsplattform koaLA geschildert. Es werden beispielhaft wissensraumbasierte Komponenten beschrieben, wie auch deren Integration in die universitäre Informationsarchitektur oder externen Content-Netzwerken. Insbesondere wird auf die Umsetzung der in Kapitel 2.3 aufgezeigten neuen Einflüsse des Web 2.0 auf das E-Learning – speziell der Einbindung benutzergenerierten Contents und soziale Kontakte in Lernszenarien – eingegangen.

Die Arbeit schließt mit einer kritischen Würdigung der Arbeit und einen Ausblick.

1.5. Wissenschaftlicher Beitrag

Der Bereich E-Learning umfasst im Kontext der Wirtschaftsinformatik nicht nur den Prozess des Wissenserwerbs im engeren Sinn. Darüber hinaus bedürfen die Konstruktion, Kommunikation und Evaluation von Wissen einer angemessenen Unterstützung durch Informationssysteme. Speziell die Modernisierung universitärer IT-Infrastrukturen zur alltagstauglichen Unterstützung eines integrierten Informationsmanagements in Lehre und Verwaltung wird mit zunehmenden Leidensdruck durch steigende Studierendenzahlen und Serviceansprüche sowie der Einführung konsekutiver Studiengänge immer mehr zu einem neuen Anwendungsgebiet der Wirtschaftsinformatik[13].

Diesem Kontext ist auch die vorliegende Dissertationsschrift zuzuordnen. Es soll ein systematisches Vorgehen zu Entwurf und Entwicklung komponentenorientierter, integrierter Lern- und Arbeitswerkzeuge konzipiert werden, das durch eine auf die Problemstellung angepasste Softwarerahmenarchitektur unterstützt wird. Dadurch soll einerseits technologiestarken Lehreinheiten eine effiziente Entwicklung individueller Lernumgebungen zu Forschungszwecken und als Profilierungsmöglichkeit geboten werden, andererseits

[13] Dies belegen die steigenden Zahlen der wissenschaftlichen Veröffentlichungen in diesem Bereich und die Berücksichtigung von E-Learning und integrierten Campus-Management-Systeme auf Fachkonferenzen, bspw. der 9. Internationalen Tagung Wirtschaftsinformatik 2009 (Track 32 und 33).

denjenigen Lehreinheiten, die dies aus eigener Kraft respektive fehlenden Ressourcen nicht stemmen können, eine niedrigschwellige, konfigurierbare E-Learning-Umgebung zur Verfügung gestellt werden.

Dabei ergeben sich nach dem Kenntnisstand des Autors wissenschaftliche Neuerungen gleich an mehreren Stellen:

1. Das Konzept der normsprachlichen Entwicklung von Informationssystemen wurde bisher nicht auf das in dieser Arbeit behandelte Anwendungsgebiet angewandt.

2. Es gibt bislang kein Vorgehensmodell zur Entwicklung von Lernumgebungen, welches auf die Umsetzung mehrerer integrierter Werkzeuge ausgerichtet ist. Bisherige Ansätze fokussieren ausschließlich die Implementierung monolithischer Plattformen.

3. Ebenso gibt es bislang keine Rahmenarchitektur für hochschulweites E-Learning, die einen zentralen Dienstekern als Basis und Integrationsplattform vorsieht.

4. Die Umsetzung und die hochschulweite Einführung einer offenen Komponentenarchitektur für E-Learning ist ein im deutschsprachigen Raum bisher selten realisiertes Anwendungsbeispiel.

Nach Meinung des Autors handelt es sich demnach um eine sowohl theoretisch als auch praktisch höchst relevante Problemstellung, die es in den nächsten Kapiteln bestmöglich zu lösen gilt.

2. Beschreibung des Problembereichs und der Einflussfaktoren

In diesem Kapitel wird auf das in der Einleitung umrissene Problemfeld detaillierter eingegangen und gegenüber verschiedenen anderen Handlungsfeldern und Forschungsbereichen abgegrenzt. Dazu wird zunächst die Motivation zur Auseinandersetzung mit neuen Medien an Hochschulen erklärt, sowie die derzeitigen Rahmenbedingungen für ihren alltagstauglichen und effizienten Einsatz (2.1). Für das Verständnis des Problemfelds von entscheidender Bedeutung sind die aktuellen Entwicklungen im Bereich des computerunterstützten Lernens. Daher gibt Abschnitt 2.2 einen kurzen Gesamtüberblick über Begriffe und Konzepte des E-Learnings, um im Anschluss daran die derzeitigen Standardisierungsbestrebungen speziell im Bereich der Softwarearchitekturen von Lernumgebungen einzuordnen. Neue technologische und soziologische Trends des so genannten *Web 2.0* beeinflussen diese Entwicklungen ebenfalls sehr stark. Abschnitt 2.3 beschreibt diese Trends und zeigt daran Verbesserungspotenziale auf, die bei der (Re-)Konzeption universitärer Lern- und Arbeitsumgebungen angemessen berücksichtigt werden müssen. Eine Zusammenfassung schließt die Konkretisierung des Problemfelds ab (2.4).

2.1. E-Learning an deutschen Hochschulen

2.1.1. Nationale Bildungspolitik und Fördermaßnahmen

Um den Strukturwandel im deutschen Bildungsbereich voranzutreiben, der durch die Globalisierung und die IuK-Techniken induziert ist, initiierten der Bund und die Länder um das Jahr 2000 umfangreiche Forschungs-, Entwicklungs- und Pilotmaßnahmen. Lebenslanges Lernen sollte damit nachhaltig für alle Menschen gefördert und Bildungsstrukturen zukunftsorientiert angepasst werden. Dabei sollten einzelne Modellversuche und Pilotprojekte als *Leuchttürme* eine exemplarische Umsetzung des hochkomplexen Konzeptes des lebenslangen Lernens im Kleinen aufzeigen. Entsprechende Förderprogramme sollten diese neuen Erkenntnisse in die Breite tragen. Die schwerpunktmäßige Zielsetzung dieser Förderprogramme bestand dabei aus den drei Aspekten *Vernetzung*, *Infrastrukturausbau* und einer neuen *Funktionszuweisung für Schulen und Hochschulen*[1]:

[1] Im Weiteren dieser Arbeit wird der Fokus primär auf die Hochschulbildung gelegt.

Vernetzung Auf regionaler und überregionaler Ebene wurden bildungsbereichs- und -trägerübergreifende Kooperationsverbünde gefördert. Diese bestanden neben öffentlichen Schulen, Hochschulen und Bibliotheken auch aus kommerziellen Weiterbildungseinrichtungen und Herstellern von Lernsoftware und -inhalten. Somit sollte zum einen der Markt an Bildungsanbietern und -interessenten für alle transparent und stimuliert, zum anderen aber auch formale und informelle Lernorte miteinander verknüpft werden.

Infrastrukturausbau Zusammen mit der deutschen Telekom gründete das Bundesministerium für Bildung und Forschung (BMBF) den Verein „Schulen ans Netz“ mit dem Ziel, alle allgemein bildenden Schulen in Deutschland mit einem Internetanschluss auszustatten und die Medienkompetenz von Schülern und Lehrern zu fördern[2]. Im Hochschulsektor wurde an vielen Universitäten sowohl die interne Vernetzung durch Funknetz gefördert[3] als auch die Vernetzung der Universitäten untereinander durch das deutsche Forschungsnetz auf 100 Gigabit ausgeweitet.

Neue Funktionszuweisung an Hochschulen Mit dem Hochschulrahmengesetz von 1998 wurde die universitäre Weiterbildung als gleichrangig neben Forschung und Lehre gestellt (§2 Abs. 1 HRG). Gleichzeitig wurden die Hochschulen aufgefordert, ihre Angebote des lebenslangen Lernens zu verbessern und anzupassen[4]. Durch die Modularisierung der Angebote sowie die Ausweitung der Autonomie und Flexibilität der Hochschulen sollten die Möglichkeiten dafür geschaffen werden[5]. Die Umsetzung an den Hochschulen gestaltete sich jedoch als schwierig, konnten Dozierende ihre Weiterbildungsaktivitäten nicht auf das reguläre Lehrdeputat anrechnen lassen[6]. Als Konsequenz daraus entstanden an vielen Universitäten zentrale Weiterbildungseinrichtungen, welche die hochschulischen Angebote (zumeist durchgeführt durch für diese Zwecke gegründeten An-Institute) seitdem initiieren und koordinieren. Weiterhin wurden an vielen Universitäten mittlerweile die Möglichkeiten der Gasthörerschaft ausgeweitet und es werden Seniorenkollegs angeboten (vgl. [SSW+06], S. 89ff., S.313).

Bei der Förderung von Inhalten für Hochschulen setzte das BMBF auf sein Programm

[2] Ende 2001 waren alle Schulen mit einem Internetanschluss ausgestattet. Ende 2006 sind mehr als 20.000 Schulen (von rund 34.000) mit modernen Breitbandanschlüssen ausgestattet.

[3] Im Förderprogramm *Neue Medien in der Bildung* wurden vom BMBF 41 WLAN-Projekte (Fördersumme über 5 Mio. EUR, Laufzeit 4 Monate) und 25 Notebook-University-Projekte (Fördersumme ca. 26 Mio. EUR, Laufzeit 14-20 Monate) unterstützt.

[4] Allerdings wurden diese Forderungen nie weiter konkretisiert, womit das Problem und die Lösung der Umsetzung lebenslangen Lernens eigentlich nur von der politischen Ebene auf die der Hochschulen durchgereicht wurde (vgl. [Keh01], S. 129ff.).

[5] Erste Evaluationen haben jedoch ergeben, dass aufgrund der festgelegten Modulstrukturen und der strafferen Organisation zum einen die inhaltlichen Gestaltungsspielräume für Lehrende kleiner werden (also auch die Möglichkeit, auf neue Themen schnell und flexibel reagieren zu können), zum anderen die Möglichkeiten für Praktika, Nebentätigkeiten und Auslandsaufenthalte einschränken, sofern sie nicht laut Prüfungsordnung verbindlich vorgesehen sind (vgl. [JG07]).

[6] Kehm nennt in [Keh01], S. 130, noch weitere Gründe: Zum einen war eine Nachfrage nach universitärer Weiterbildung zu dem Zeitpunkt nicht hinreichend konkret, zum anderen hatte die wissenschaftliche Weiterbildung nur einen marginalen Stellenwert in der durch Forschung geprägten traditionellen Werthierarchie.

Neue Medien in der Bildung[7] (vgl. [BMB00]). Durch den Aufbau und der Nutzung multimedialer Informationsquellen für Dozierende und Studierende sollten die „Neuen Medien“ auf breiter Front Einzug in die Weiterbildung halten. Dazu wurden insgesamt 100 ausgewählte Projekte (E-Learning-Arrangements) gefördert, deren Aufgabe im Wesentlichen die Bereitstellung von Inhalten war (*Content-Projekte*) und die gleichzeitig die neue Lehr- und Lernkultur implementieren sollten: Informelles, selbstgesteuertes Lernen, in Verbindung mit Lernberatung und Motivierung.

Da die oben genannten Infrastrukturprojekte (WLAN- und Notebook-University) aber in erster Linie auf die Anpassung der technischen und organisatorischen Vernetzung der universitären Infrastruktur abzielten und weniger auf die Bereitstellung von Lernumgebungen für die neuen Inhalte, entstanden in den Content-Projekten sozusagen als Nebenprodukt sehr viele Individuallösungen[8], welche noch Jahre später den heutigen Heterogenitätsgrad der universitären IT-Landschaften stark beeinflussen. Dieser Effekt wurde noch durch mehrere Faktoren verstärkt: Erstens wurden die Content-Projekte nur als Verbundprojekte mit einer Spannbreite von zwei bis 17 Kooperationspartnern gefördert, was eine stattliche Zahl von 540 Einzelprojekten ergab (vgl. [Bau03a], S. 5). Dies führte innerhalb der beteiligten Universitäten zu einem unsystematischen Nebeneinander von Aktivitäten im selben Feld, da es zweitens nur in seltenen Fällen hochschulweite Strategien und Pläne gab. Als dritter verstärkender Faktor wirkte zu diesem Zeitpunkt das Nicht-Vorhandensein von Standards für Lernumgebungen und -inhalte[9]. Nicht zuletzt befand sich die Forschung zu diesem Thema auf ihrem Höhepunkt, so dass selbst innerhalb von Verbundprojekten die beteiligten Partner verschiedene Lösungen konzipierten, implementierten und evaluierten (s. [RS05], S. 146ff.).

Bei Nachhaltigkeitsüberlegungen zu einer hochschulweiten Ausweitung der E-Learning-Angebote auf alle Lehrenden und Studierenden ergab sich durch die ansteigende Zahl der Nutzerinnen und Nutzer ein Skalierungsproblem[10], das mit den damaligen isolierten Strukturen nicht mehr bewältigt werden konnte: Interne Strukturen und Aufgaben müssen dazu entsprechend angepasst und übergreifend aufgebaut werden[11] (vgl. [Kub04], S. 2). Ein für den nachhaltigen Einsatz notwendiger Schritt ist dabei der Aufbau einer organisatorischen und technischen Vernetzung in den Hochschulen und den in ihrem Umfeld etablierten Einrichtungen (vgl. ebd., vgl. [Sch04a], S. 1).

Da die initialen Fördermaßnahmen hierzu nicht ausreichten, wurde der Förderschwerpunkt „Neue Medien in der Bildung“ um „Maßnahmen der Strukturentwicklung“ im Jahr 2004 ausgeweitet, die bis zum Jahr 2008 die „systematische und professionelle Produktion und Nutzung digitaler Lehrmaterialien jenseits von Drittmitteln finanzierten

[7] Förderinitiative „Einsatz Neuer Medien in der Hochschullehre“, Fördersumme ca. 180 Mio. EUR.

[8] Aus dem Audit-Bericht: „Fast alle Projekte haben Lehr-/Lerneinheiten produziert (94 %), während der Anteil derjenigen Projekte, die (auch) Produkte im Bereich der Wissensressourcen und Tools entwickelt haben, etwas mehr als die Hälfte ausmacht (55 bzw. 52 %)“, [Bau03a], S. 8.

[9] „Learning Management Systems have evolved so far, but they have each developed independently and despite the similarities in their feature sets, no interoperability exists between them. Furthermore, they do not integrate with other third-party systems“ [BCG03].

[10] Sowohl organisatorisch als auch technisch, denn die meisten Anwendungen waren zumeist auf einen lehrstuhlinternen Einsatz ausgerichtet (Datenmodell, didaktisches Modell).

[11] Aufgaben wie Betrieb, technischer Service und Support, Qualifizierung und pädagogischer Support, Content-Entwicklung, curriculare Integration, Verwaltungsintegration, Marketing sowie Qualitätssicherung und -entwicklung müssen hochschulweit organisiert werden (vgl. [Kub04], S. 2).

und zeitlich befristeten Projekten und über das Engagement von einzelnen Pionieren hinausgehend“ etablieren soll [Sch04a]. Erstmalig liegt dabei der Fokus nicht mehr nur auf dem Lehren und Lernen, sondern auf der Umsetzung von Integrationspotenzialen und Prozessverbesserungen an den jeweiligen Universitäten[12] (vgl. ebd.).

2.1.2. Zielgerichteter Einsatz von IuK-Technologien an Hochschulen

Die Erwartungen, die von Politik, Gesellschaft und Wirtschaft an Hochschulen gerichtet werden, umfassen unter anderem die Etablierung einer den Problemstellungen der Wissensgesellschaft angemessenen Lehr-/Lernkultur sowie eine Verbesserung der Effizienz der Hochschulen bei der Wahrnehmung ihrer (Weiter-) Bildungsaufgaben[13].

In Abschnitt 2.1.2.1 wird die Vielfältigkeit der Anforderungen an konstruktivistisch geprägte virtuelle Lern- und Arbeitsumgebungen theoretisch aufgezeigt und den idealtypischen Potenzialen des Einsatzes von Multimedia und IuK-Technologie in der Lehre gegenüber gestellt.

Ein zielgerichteter IKT-Einsatz in der Hochschullehre trägt darüber hinaus die Potenziale in sich, gerade durch die Vernetzung von Technologien über die Grenzen verschiedener Einsatzfelder hinweg, wie E-Learning, E-Science, E-Government und der Digitalen Bibliothek, Synergieeffekte zu erzielen und zur Effizienzverbesserung von Lehre und Verwaltung beizutragen. Abschnitt 2.1.2.2 zeigt diese Potenziale auf.

Zur Implementierung und zum Betrieb einer IT-Infrastruktur, die den gestiegenen Anforderungen gerecht wird, bedarf es an den meisten Hochschulen auch weitreichender Änderungen organisatorischer Art, um die neuen Angebote kostendeckend und alltagstauglich nachhalten zu können (Abschnitt 2.1.2.3). Entsprechend wichtige informationstechnische Maßnahmen werden in Abschnitt 2.1.2.4 vorgestellt.

2.1.2.1. Neue Medien in der Hochschullehre

In Deutschland kann man rückblickend zwei große „Wellen“ von technologischer Unterstützung des Lehrens und Lernens ausmachen. Die erste bildungstechnologische Welle setzte bereits Mitte der 1960er Jahre ein. Zu der Zeit wurde versucht, mit so genannten *Lernautomaten* bessere Lernerfolge zu realisieren. Dabei wurden die behavioristischen lernpsychologischen Konzepte von B. F. Skinner angewandt, um über computergesteuerte Rückmeldungen in Frage-Antwort-Mustern das Verhalten des Lernenden respektive seine Antworten zu beeinflussen. Da diese Programme eine vorprogrammierte Sequenz von Lernschritten abarbeiten, werden solche Anwendungen auch als *programmierte Instruktion* bezeichnet[14].

[12] In einer zweiten Förderlinie wird zwar auch die Verstetigung von E-Learning-Angeboten in hochschulübergreifenden Netzwerken gefördert. Aber auch diese sind – trotz ihrer Verankerung an einzelnen Instituten und Fachbereichen – letztlich auf das *Backbone* der gesamten Hochschulstruktur angewiesen (vgl. [EKW05], S. 1).

[13] Kehm geht in [Keh01] genauer auf die Erwartungen an Hochschulen in diesem Kontext ein und zählt neben den hier genannten Aspekten auch die Schaffung eines flexiblen und offenen Zugang zur Hochschulbildung, Modularisierung und Nachfrageorientierung hinzu (S. 131).

[14] N. Crowder führte später Verzweigungen in den Sequenzen ein, um in Abhängigkeit von der Art des Fehlers nicht den gleichen Lerninhalt erneut, sondern alternative Darstellungen anzubieten.

Diese Methodik stieß jedoch auf viel Kritik: Die stereotype Aneinanderreihung von Informationseinheiten und Prüfungsfragen ist für Lernende oft demotivierend und führt nur zu mäßiger Akzeptanz, wenn nicht sogar Ablehnung. Zudem kann ein tieferes Verständnis der Lerninhalte kaum erschlossen werden, weswegen die Anwendungen vielfach auf reines Faktenwissen beschränkt blieben (vgl. [Ker01], S. 65).

Die zweite Welle wurde in den 1990er Jahren initiiert, als die Bearbeitung von Grafiken und die Wiedergabe von Audio und Video sowie Animationen auf dem Computer für jedermann möglich wurden und Zugänge zum Internet allgemein vorhanden waren[15]. Im Vergleich zu traditionellen Medien verfügen digitale Medien über erweiterte Darstellungs-, Kommunikations- und Aktualisierungsmöglichkeiten. Dadurch wurde ihnen gerade für die Hochschulbildung neue Qualitäten des Lehrens und Lernens sowie Kosteneinsparungen zugewiesen, was zunächst an idealtypischen Potenzialen festgemacht wurde[16]:

„Richness“ Reichhaltigkeit in Darstellungsformen (Text, Bild, Ton, Animation, Video etc.), Navigationsmöglichkeiten (Hyperlinks, Sitemaps, Themenlisten, History, Bookmarks etc.), Hilfe-Angebote (Hilfe-Seiten, Glossar, Weblinks, integrierte virtuelle Tutoren etc.), Vermittlungsmethoden (individuelles, problemorientiertes, selbstgesteuertes, entdeckendes Lernen, Gruppenarbeit etc.) und Einsatzgebiete (zur Motivation, ergänzend als Informationsquelle, in Selbstlernphasen, zur Vor-/Nachbereitung, zur Vertiefung etc.)

„Communication“ Interaktivität zwischen Nutzern und Anwendung (verschiedene Formen der Benutzungsschnittstelle), mit multimedialen Inhalten (dialektisches Verknüpfen von Kognition und Aktion) und soziale Interaktion (synchrone und asynchrone Kommunikation, soziale Vernetzung, Awareness)

„Suitableness“ Adaptivität mit den Komponenten, Adaption an individuelle Voraussetzungen und Veränderungen (Anpassung der Lerninhalte an Lernvoraussetzungen, Lerntyp, Lerntempo etc.), Adaptierbarkeit der Inhalte des Systems (sinnvolle, spezifische Einbindung der Darstellungsformen) sowie Adaptierbarkeit an organisatorische Besonderheiten (Anpassung an das Curriculum, das Lernumfeld, die Zielgruppe, den aktuellen Bedarf etc.)

„Independence“ Ortsunabhängigkeit und zeitliche Unabhängigkeit (Selbstbestimmung der Lernzeiten, Anpassung an den eigenen Lernrhythmus, Selbsteinteilung der Stoffmenge etc.). Die Ortsunabhängigkeit (*Alokalität*) von Lernen mit digitalen

[15] In der Zwischenzeit fanden Versuche statt, durch so genannte *intelligente tutorielle Systemen* den individuellen Kompetenz- und Wissensstand von Lernenden zu diagnostizieren und mit einem „idealen Modell der Expertise“ zu vergleichen, um Inhalte und Art der Vermittlung für den Lernprozess vorzuschlagen. Dieser Ansatz ist jedoch bis heute noch nicht in vertretbarem Aufwand allgemein, d. h. domänenunspezifisch, umzusetzen.

[16] Obwohl diese Potenziale in der Literatur oftmals zusammen genannt werden, schließen sie sich in der Praxis u. U. wechselseitig aus. Ein Beispiel ist die Zeitabhängigkeit bei sozialen und kommunikativen Szenarien. Eine Übersicht über die Limitierungen bei computerunterstütztem Lernen bietet [Kal03], S. 58ff.

Medien wird mit Begriffen wie „*in-placE-Learning*“ oder „*self-placed-learning*“ umschrieben

„Broadcast“ Globalität und Wirtschaftlichkeit bei der Informationsbereitstellung (weltweite Bereitstellung, gleichzeitige Nutzung, Wissensmanagement etc.) und den Informationszugriff.

Diese Potenziale der neuen digitalen Medien entfachten in der Hochschuldidaktik die Diskussion über alternative Lehr-/Lernformen und regen – auch noch mehr als zehn Jahre später – zur Entwicklung und Erprobung neuer didaktischer Konzepte an (vgl. [Sch01b], [SW04], [BD05], [Bau06b], [Dür06], [O'H07]).

Im Fokus stehen dabei selbstorganisiertes Lernen und konstruktivistisch geprägte Lernformen, bei denen die eigenständige Erarbeitung der Inhalte durch den Studierenden (Konstruktion) und dessen kognitive bzw. konstruktive Aktivität im Mittelpunkt didaktischer Überlegungen stehen. Studierende sollen dabei insbesondere zum selbstgesteuertem Lernen angeleitet werden. Sie sollen Methoden und Fähigkeiten beherrschen lernen, um kontinuierlich zu lernen, eigene Wissens- und Kompetenzlücken zu identifizieren, Möglichkeiten zu nutzen, diese zu füllen und eigene Fortschritte zu verfolgen.

Um den im vorigen Abschnitt skizzierten Strukturwandel in der Hochschulbildung voranzutreiben und um parallel dazu die hier skizzierten idealtypischen Potenziale neuer Medien für Bereiche des Lehrens und Lernens in der Breite umzusetzen, förderte das BMBF in den Jahren 2001-2003 im Rahmen des Förderprogramms „Neue Medien in der Bildung“ einhundert Projektverbünde (mit insgesamt 541 Einzelprojekten) an deutschen Hochschulen. Sie sollten virtuelle Studienangebote, virtuelle Lernumgebungen und entsprechende Lern- und Betreuungswerkzeuge für verschiedene Fachbereiche entwickeln und evaluieren (vgl. [BMB00], S. 8ff.). Hochschulpolitische Ziele waren dabei u. a. die Verbesserung der Qualität der Hochschullehre, die Erhöhung der Selbst- bzw. Fernstudienanteile und die Entwicklung neuer Lehr-/Lernformen, die Präsenz- und Online-Lehre kombinieren sollten[17].

Diese Förderinitiative hat zwar keine Revolution der Hochschullehre ausgelöst, aber grundlegende Änderungen aufgezeigt, mit denen das Postulat „Lehrverbesserung“ eingelöst werden konnte (vgl. [Bau03a] und [RB04]). Grundsätzlich wurde durch die neuen Lernangebote, -umgebungen und -werkzeuge eine kritische Masse an praxistauglichen Medienprodukten erzeugt (s. [BMB04b]). In einer empirischen Studie über die Nutzung dieser Medienprodukte wurde festgestellt, dass ihr Einsatz vorwiegend in so genannten *Blended Learning*-Szenarien – auch als *Crossmedia-Lernen* bzw. *Hybrides Lernen* oder *Methoden-Mix* bezeichnet – erfolgte und somit zu einer Koexistenz traditioneller und innovativer Lehrstrukturen führte: Neue Inhalte und Methoden ergänzen in diesen Fällen Althergebrachtes (*Anreicherungskonzept*[18]). Erklärtes Ziel ist es dabei, durch eine didaktisch sinnvolle Verknüpfung von traditionellen Lernkonzepten und -medien mit virtuellem bzw. Online-Lernen ein möglichst optimales Lernarrangement zu schaffen.

[17] Weitere hochschulpolitische Ziele waren die Schaffung neuer Fern- und Weiterbildungsangebote sowie die Herstellung internationaler Wettbewerbsfähigkeit (vgl. [BMB00], S. 15).

[18] Weiterführende Quellen zu den Konzepten des Blended Learnings sind [RR03], [Vol04], [SSB04] und [Rei05a].

Zum Teil kann aber auch von einer qualitativen Verbesserung der Lehre im Sinne einer neuen Lehr-/Lernkultur gesprochen werden. Neben einer überwiegend darstellungsorientierten Nutzung neuer Medien wurden im geringeren Umfang auch offenere, explorative Lehr-/Lernformen realisiert. Diese stellen die Bedürfnisse der Studierenden stärker in den Mittelpunkt und unterstützen ein selbstorganisiertes Lernen und z. T. auch explizit Partner- und Gruppenarbeit.

Die Zunahme der Partner- und Gruppenarbeit im Vergleich zu traditionellen Veranstaltungsformen steht in den Förderprojekten offensichtlich in Zusammenhang mit der Nutzung telemedialer Anwendungen, die neue Formen der Interaktivität zwischen Lehrenden und Lernenden aber auch zwischen Lernenden untereinander unterstützen. In Kombination mit multimedialen Anwendungen können unterschiedliche Veranstaltungsformen und Lehr-/Lernmethoden[19] virtualisiert werden. Eine Übersicht über die am häufigsten eingesetzten Typen multimedialer Anwendungen und ihre Haupteinsatzgebiete ist in Tabelle 2.2 gegeben.

In einem resümierenden Audit des Förderprogramms (vgl. [Bau03a]) wurde neben einer generell positiven Resonanz – die E-Learning-Community in Deutschland wurde gestärkt und neue Medien in der Hochschullehre in die Breite getragen – auch einige Kritik geübt.

So wurde von den Experten mehrfach bemängelt, dass große Potenziale neuer Medien in den im Förderprogramm gestalteten Lehr- und Lernsituationen ungenutzt blieben. Die meisten Medienprodukte boten oft nur darstellende Lernformen an und setzten die Integration in pädagogisch wichtige Interaktions- und Kommunikationsprozesse nur lücken- oder mangelhaft um (vgl. auch [RB04], S. 436). Informationstechnisch gesehen bieten digitale Medien das Potenzial, Informationsobjekte zu Objekten der Manipulation zu machen und dadurch den neuen Anforderungen nach mehr selbstbestimmten, aber auch kooperativ organisierten Wissenserwerb in Lernszenarien nachzukommen. Große Einschränkungen bzgl. der Kommunikation und Interaktion traten in den im Förderprogramm neu geschaffenen Lernsysteme aber auf Ebene der Informationsträger auf: Viele Systeme betten die Inhalte ein und begrenzen somit durch die feste Verbindung von Information und Träger Zugriffe und Manipulationen. Die Benutzer können also nur in dem durch das System vorgegebenen Rechte- und Funktionsrahmen auf die Inhalte zugreifen. Klassische Lernsysteme, die dabei ein festes Rollenmodell von Autoren und Lernenden implementieren, erlauben zumeist keine direkte Manipulation von Inhalten durch Lernende. Wenn die somit gesetzten Grenzen für bestimmte Benutzergruppen nur die reaktive Betrachtung, nicht aber die aktive Manipulation zulassen, können weitere Bearbeitungsschritte nur nach dem Transfer der Inhalte in ein neues Werkzeug realisiert werden (vgl. [Now05], S. 62). Dadurch werden Medienbrüche erzwungen, die dem Benutzer die selbstbestimmte Interaktion mit Materialien erschweren. Analog dazu wird die Kommunikation über Inhalte in kooperativen Szenarien aufwändiger und dadurch auch komplexer, wenn sie nicht an den Informationsobjekten direkt, sondern über nebenste-

[19] Die Studie von Rinn und Bett zählte im Jahr 2004 Skripte, Übung, Problembasiertes Lernen, Vortrag/Vorlesung und den Dialog zwischen Lehrenden und Lernenden zu den am häufigsten virtualisierten Szenarien. Danach folgten Seminar, Fallstudie, Projektarbeit, Praktikum, Handapparat/ Bibliothek, Planspiel und die Betreuung von Arbeiten (vgl. [RB04], S. 435).

hende, in sich eigenständige Technologien, wie Chat, Mailsysteme und Foren, stattfinden muss. Die Interaktion und Kommunikation wird dann unnötigerweise gebremst, da für Inhalt und Kommunikation jeweils getrennte Kommunikationskanäle verwendet werden müssen. Keil charakterisiert diese Situation als „langsame Interaktion“ (s. [KS05a], S. 16).

Weiterhin kritisierten die Experten, waren die geschaffenen Angebote sehr von organisatorischen Strukturen und der Prozesssicht der Lehrenden geprägt. Informelle Strukturen (bspw. veranstaltungsübergreifende Lerngruppen) und Anforderungen von Studierenden wurden oftmals nicht berücksichtigt.

Der zweite große Kritikpunkt war die mangelnde Sorge um die Nachhaltigkeit der entwickelten Medienprodukte in den einzelnen Projekten. Die von der Bildungspolitik geschürten Erwartungen an einen Markt für universitäre Lerninhalte wurden enttäuscht. Die Nachfrage von Externen oder von anderen Dozierenden gegen ein Entgeld blieb über den Zeitraum der Förderung gering, so dass viele Projekte nach ihrem Wegfall ihre Angebote nicht mehr kostendeckend betreiben konnten.

Auch wurde die Ausnutzung von Skaleneffekten durch die häufig mangelnde Übertragbarkeit der entwickelten bzw. eingesetzten Werkzeuge (Software) auf andere Fachbereiche verhindert. Die Ausgaben für Betrieb, Wartung und Weiterentwicklung der geschaffenen Lösungen mussten daher oftmals im vollen Umfang von den Lehrstühlen allein getragen werden. Zu den Betriebs- und Wartungskosten der eigentlichen Software hielt zusätzlich der eigenverantwortliche Betrieb von Basisdiensten wie Webserver, Mailserver, Chatserver, Verzeichnisdienst, Datenbank und -sicherung sowie Betriebssysteme die Kosten hoch, wenn keine Integration in die universitäre IT-Infrastruktur erfolgt war. Daneben bedeutet auch eine solche „Nicht-Integration“ für den Betrieb immer Doppelerfassungen von Daten durch Medienbrüche.

Die Expertenkommission regte im Audit des NMB-Förderprogramms daher ein anschließendes Förderprogramm an, in welchem die Rahmenbedingungen für eine nachhaltige mediengestützte Lehre an Hochschulen weiterentwickelt werden kann. Dies betrifft sowohl die Vermeidung von Medienbrüchen im gesamten Lehr- und Verwaltungsprozess der Hochschulen als auch die Integration der mediengestützten Angebote in die strategische Profilbildung der Hochschulen, Länder und der gesamten Bundesrepublik Deutschland.

2.1.2.2. Heterogene Prozess- und Systemstrukturen

Das BMBF schrieb daraufhin 2004 eine neue Förderung aus zur „Entwicklung und Erprobung von Maßnahmen der Strukturentwicklung zur Etablierung von E-Learning in der Hochschullehre im Rahmen des Förderschwerpunkts *Neue Medien in der Bildung*“ (s. [Sch04a]). Eine erfolgreiche und IT-gestützte Modernisierung von Lehren, Lernen und Prüfen an Hochschulen und die systematische Erschließung des diesbezüglich in den Hochschulen vorhandenen Innovationspotenzials sind organisationsneutral, d. h. ohne Strukturveränderungen, nicht zu erreichen. Mit dem Förderprogramm soll eine „Anschubfinanzierung“ zur beschleunigten Entwicklung und Erprobung von Organisationsmodellen geleistet werden, die – insbesondere mit den Medienentwicklungskonzepten der

einzelnen Hochschulen – zu einer verstärkten Nutzung von E-Learning und E-Teaching führen und somit die Qualität und Effizienz von Lehren, Lernen und Prüfen zu erhöhen. Dabei sei die hochschulweite Integration von E-Learning als strategische Aufgabe für die Hochschulentwicklung insgesamt anzusehen.

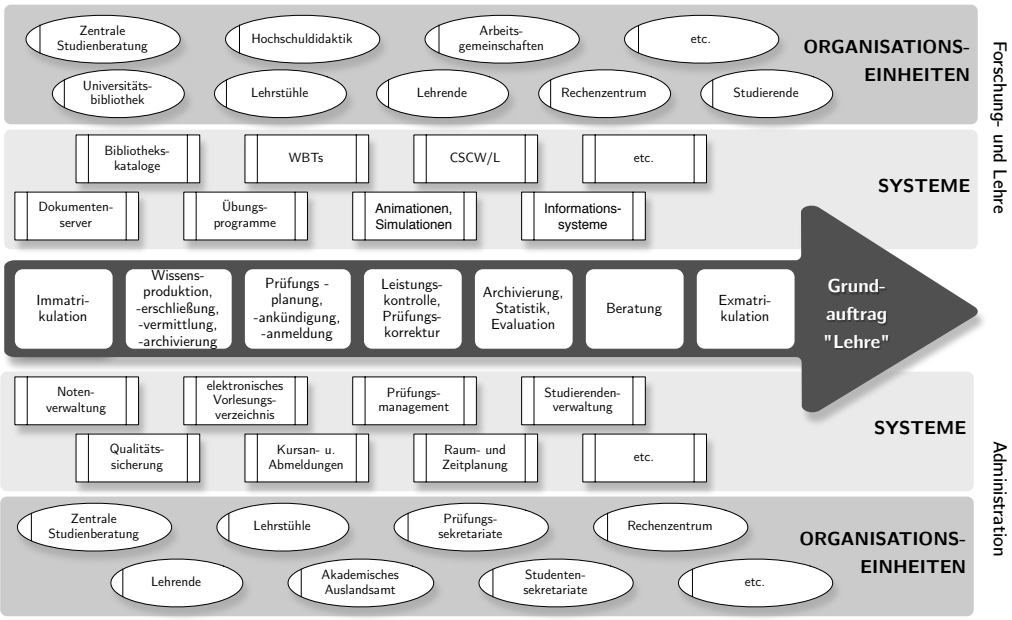

Abbildung 2.1.: Das Hochschulstudium, hier dargestellt als Prozess, ist mit seinen beiden Kernbereichen Wissensorganisation und Modul- und Prüfungsverwaltung sowohl organisatorisch als auch informationstechnisch stark fragmentiert.

Im Folgenden wird die Hochschullehre als Prozesskette gesehen, die durch die Bereiche Kernprozesse (Lehre und Forschung), direkte Supportprozesse (z. B. Raumplanung, Prüfungsabwicklung) und indirekte Supportprozesse (z. B. Schlüsselverwaltung, Haushalts- u. Personalprozesse) unterstützt wird (vgl. [BGKT07], S. 44ff.)[20]. In diesen beiden Bereichen werden heute eine Vielzahl multi- und telemedialer Anwendungen eingesetzt, die zum größten Teil Insellösungen darstellen. Der Einsatz dieser Systeme erfolgt durch verschiedene Organisationseinheiten, die in unterschiedlichen Verantwortungsbereichen auf unterschiedlichen Ebenen angesiedelt sind. Diese Situation führt dazu, dass die Transparenz des Gesamtprozesses stark erschwert wird und der gesamte Prozess durch eine hohe Anzahl Schnittstellen zwischen den Beteiligten, den Produkten und Organisationseinheiten gekennzeichnet ist. Abbildung 2.1 beschreibt diesen Sachverhalt grafisch.
Innerhalb von Hochschulen wird immer noch differenziert nach Verwaltung, Forschung

[20] Vgl. hierzu Abbildung 2.1: Die Prozessschritte laufen dabei typischerweise nicht sequentiell, sondern in starkem Maße parallel und studienbegleitend ab.

und Lehre sowie nach Fachbereichen, Querschnittszentren und Dienstleistern. Weick führt in [Wei76] an, dass zwischen diesen einzelnen Organisationseinheiten zwar lockere Verbindungen bestehen, sie sich aber auch durch einen hohen Autonomiegrad auszeichnen[21]. Dieser Umstand zieht zum einen ineffiziente Prozesse nach sich, da die in der Administration zuständigen Ansprechpartner für – aus Sicht der Lehrenden zusammengehörende – Abläufe stark verteilt sind (vgl. [DEH+02], S. 3f.). Für Studierende einer Hochschule ergeben sich dieselben Probleme im verschärften Maße. Auch sie sind mit mehreren Ansprechpartnern konfrontiert und müssen sich nach deren Kommunikationsanforderungen richten, die ggf. nach den Traditionen des jeweiligen Faches bzw. einzelner Lehrstühle variieren.

Zum anderen bedeutet die starke organisatorische Fragmentierung im Zusammenspiel mit einem hohen Autonomiegrad für Strukturveränderungen eine geringe Steuerbarkeit und Berechenbarkeit der Einzelelemente, aber auch des Gesamtsystems. Steuerungsversuche bezüglich einzelner Organisationseinheiten wirken sich nicht notwendigerweise auf die gesamte Organisation aus (vgl. [Kub04], S. 9ff.). Technische Innovationen und Prozessänderungen werden dann dadurch gehemmt, dass jedes mal auf technische, pädagogische und verwaltungsbezogene Aspekte der einzelnen Subsysteme Rücksicht genommen werden muss (vgl. ebd. S. 9ff.).

Aus informationstechnischer Sicht ergeben sich durch die an Hochschulen vorherrschenden heterogenen IT-Infrastrukturen große Effizienzpotenziale bzgl. der Unterstützung aller administrativen Prozesse, aber auch aller im Wissenschaftsbetrieb anfallenden Prozesse der Verarbeitung von Wissen, sei es in Lehr- und Lernprozessen, in der Forschung und Entwicklung sowie bei der Kommunikation und Publikation: Studierenden- und Prüfungsdaten, Lehrveranstaltungen und Räume sowie deren Belegungen werden oftmals redundant gehalten und müssen in verschiedenen Systemen mehrfach erfasst und manuell abgeglichen werden. Auch der Austausch und die gemeinsame Bearbeitung von Dokumenten und Daten, der Zugriff auf Online-Publikationen etc. unterliegt an den meisten Hochschulen noch immer den Restriktionen von fachbereichsspezifischen Rollen, Rechten und Systemgrenzen. Zudem existieren für verschiedene Systeme häufig unterschiedliche Benutzer-IDs zur Authentifizierung, so dass nicht nur ein Mehraufwand seitens der Dozierenden vorhanden ist, sondern auch die Studierenden mit einer Vielzahl an Logins und Passwörtern, Benutzungsschnittstellen und Mehrfacherfassungen von Daten[22] für und in verschiedenen Systemen konfrontiert sind.

In der Anwendung der Dienste bedeutet dies sowohl für das Personal als auch für die Studierenden einer Hochschule einen unnötigen Mehraufwand und ungewollte Verzögerungen bei alltäglich wiederkehrenden Nutzungsszenarien. Um die Prozesse zu optimieren und somit gleichzeitig auch geeignete Rahmenbedingungen für den nachhaltigen Einsatz von neuen Medien in der Hochschullehre zu schaffen, muss die bestehende

[21] Cohen et al. bezeichnen ein solches lose gekoppeltes System autonomer Einheiten in [CMO72] als *garbage can*: Dadurch, dass viele Leute verschiedene Dinge hinein tun, ergibt sich für jeden Betrachter immer wieder ein anderes Bild. Als Ergebnis einer solchen „organisierten Anarchie“ sind den Mitarbeitern die Ziele und Verfahren des Gesamtsystems oft unklar, bzw. müssen erst häufig in Folge von Handlungen (re-)konstruiert werden, statt sie a priori setzen zu können (vgl. [CMO72], S. 2).

[22] Bspw. Benutzerprofile und Kursbuchungen und -abmeldungen.

Fragmentierung in den beiden Kernbereichen sowohl organisatorisch als auch informationstechnisch beseitigt werden.

2.1.2.3. Organisatorische Maßnahmen

Mit der Einführung und dem nachhaltigen Betrieb neuer Technologien in der Hochschullehre ist eine Menge an Aufgaben und Ressourcen verbunden. Darunter fallen beispielsweise Angebote zur kostendeckenden Produktion, Nutzung und Verarbeitung von Medien, sowie zur Kompetenzentwicklung, pädagogischem Support und Projektmanagement. Nicht zuletzt bedarf es – motiviert durch eine Defragmentierung der Systemlandschaft – einer entsprechend großen Umstellung respektive Erweiterung der IT-Infrastruktur und des technischen Betriebs und Supports. Eine Skalierung solcher integrierten Dienste und Services auf einen hochschulweiten Einsatz bedingt neue Formen der Organisation und Zusammenarbeit: Eine enge Kooperation und Koordination aller Dienste-Erbringer und Serviceleister ist eine Voraussetzung, die zunächst einmal schrittweise entwickelt und aufgebaut werden muss[23]. Aus diesem Grund ist ein umfassendes *Change-Management* erforderlich, um die existierenden fragmentierten Akteurskonstellationen hin zu effizienteren Organisationsstrukturen zu überführen, mit denen ein breites Spektrum an kunden- und serviceorientierten IuK-Angeboten und -diensten für neue Medien in der Lehre zur Verfügung gestellt werden kann. Kubicek führt in [Kub04], S. 16f., die möglichen Optionen auf:

- ➢ Gründung einer neuen Einrichtung
- ➢ Erweiterung des Aufgabenbereichs bestehender Einrichtungen
- ➢ Auslagerung bzw. Ausgliederung aus der Hochschule
- ➢ Nutzung externer Dienstleister oder übergreifende Kompetenzzentren
- ➢ Institutsübergreifende Definition von Arbeitsabläufen und Verantwortlichkeiten.

Innerhalb des akademischen Bereichs gilt es, die an einer Hochschule vorhandenen unterschiedlichen didaktischen und technischen Ansätze in ein Gesamtkonzept zu integrieren (vgl. [AMM06], [Bre04], S. 14, [Deg04], S. 3, [EKW05], S. 1 u. 3f., [Hop05a], [Wan07]). Bewährt hat sich dabei die Verschränkung von Top-Down-Elementen mit Bottom-Up-Initiativen (vgl. [AMM06]). Weiterhin müssen Anreizstrukturen geschaffen werden, um sowohl das Engagement von Pionieren und didaktische Innovatoren beizubehalten als auch die Änderungsbereitschaft an der Basis zu stimulieren und andere Dozierende als neue Akteure zu gewinnen[24].

[23] Neben der o. g. geringen Steuerbarkeit autonomer Organisationseinheiten zählt Fischer in [FB06], S. 66, noch eine fehlende Internalisierung von Kundenorientierung und des Service-Gedankens bei vielen an der Hochschule Beschäftigten zu den Herausforderungen, sowie fehlende Verrechnungsmöglichkeiten für erbrachte Dienstleistungen und untransparente Leistungsspektren beteiligter Einheiten. Degkwitz und Schirmbacher führen in [DS07], S. 21, darüber hinaus noch die hohe Spezialisierung von Hochschulangehörigen im akademische Bereich mit einem wissenschaftlichen Anspruch an, der eine bereichsübergreifende Prozesssicht einschränkt.

[24] Beispielsweise kann die Leistung einzelner öffentlich anerkannt und den jeweiligen Projekten eine Vorbildfunktion zugesprochen werden. Darüber hinaus bieten hochschulinterne Förderprogramme oder Pilotprojekte finanzielle Anreize.

Top-down Maßnahmen zur Integration von dezentral betriebenen IT-Diensten in eine universitäre Informationsarchitektur mit einer zentralen Technologie- und Architekturstrategie und definierten Architekturprinzipien[25] würden hier nicht nur einen Effizienzgewinn bedeuten. Sie würden auch eine große Handlungssicherheit für die Akteure herstellen, welche im Rahmen ihrer Lehre auch Individualsoftware und fachspezifischen Lösungen einsetzen, die nicht zentral betrieben werden können/sollen. Solche Leitungsentscheidungen zur strategischen Ausrichtung würden E-Learning-Aktivitäten an der „Basis“ rahmen und fokussieren (zum sog. *Gegenstromverfahren* s. [Hop05a], S. 167; vgl. auch [Ker04a]) und unterstützen somit den Pioniergeist von Entwicklern und didaktischen Innovatoren, wenn sie ihre Errungenschaften systematisch mit anderen Akteuren vernetzen können[26].

2.1.2.4. Informationstechnische Maßnahmen

Aus informationstechnischer Sicht bedeutet die Zusammenführung der verschiedenen IT-Anwendungen zu einem verbundenen Informationsangebot den Übergang von einer verteilten kooperativen IT-Versorgung hin zu einer integrierten Informationsversorgung. Dabei bedarf es einer Integration von Anwendungen, ohne die vorhandene Vielfalt an didaktischen Arrangements einzuschränken.

Die Grundlage dazu bildet ein übergreifendes Identitäts- und Rechtemanagement (vgl. [HSWH07]). Dies konsolidiert die oftmals dezentral und redundant vorgehaltenen Benutzerverwaltungen und bildet sie in einem zentralen System ab. Sämtliche Aufgaben, die mit dem Anlegen, Ändern und Sperren von Benutzerkonten in Verbindung stehen, können somit zusammengefasst werden. Dadurch können sowohl Mehrfacherfassungen von Benutzerdaten vermieden, Login-Routinen von Benutzern reduziert, aber auch die Gefahr von Dateninkonsistenzen verringert werden[27]. Bei diesem Integrationsschritt kann nach zwei Strategien vorgegangen werden (vgl. [Pus04], S. 89f.):

- ➢ Aufbau einer zentralen Benutzerverwaltung über ein einheitliches Mitarbeiter- und Organisationsdatenmanagement und Ablöse aller dezentralen Datenbestände[28]. Hierbei muss jedoch ein universitätsweit einheitliches Kriterium zur Identifikation der Benutzer eingeführt werden.

[25] Die *Technologiestrategie* legt Leitlinien für die Entwicklung der IT-Basisinfrastruktur fest und hat daher direkten Einfluss auf alle zu entwickelnden IT-Architekturen. Die *Architekturstratgie* beschreibt den Zielzustand inkl. des Weges zu seiner Erreichung, und unter *Architekturprinzipien* versteht man alle architekturbezogenen Grundsätze und übergreifende Standardisierungen, die für alle Weiterentwicklungen gelten (vgl. [Der03]).

[26] Zur Schaffung von Anreizstrukturen für das Engagement von Hochschullehrenden s. [ES04], [EKW05], S. 4, [ZR05], S. 45 und [AMM06]. Darüber hinaus beschreibt Moog in [Moo05], S. 84ff., mögliche Konzepte der IT-Versorgung.

[27] Puschmann führt in [Pus04], S. 81f., noch weitere Gründe an: Weniger Passwortänderungen, Reduktion des Administrationsaufwandes bei der Wiederherstellung von Zugängen aufgrund verlorener oder vergessener Passwörter etc.

[28] Als Standard für die Speicherung hat sich der OSI-Standard X.500 etabliert. Eine Benutzerkennung wird in einem solchen Verzeichnis als Objekt mit zugeordneten Attributen in einer Baumstruktur organisiert (vgl. [KL03], S. 15ff.). Als Übertragungsprotokoll von Benutzerdaten hat sich das *Lightweight Directory Access Protocol* (LDAP) etabliert und als Quasi-Standard durchgesetzt ([KL03], S. 29ff.).

- ➢ Aufbau eines zentralen Systems zum Verwalten eines Mappings zwischen den unterschiedlichen Benutzernamen und Passwörtern der verschiedenen Systeme. Dabei wird an einer dezentralen Benutzerverwaltung festgehalten.

Die Einführung eines Verfahrens zum Single Sign-On sorgt in diesem Rahmen für eine systemübergreifende Authentifizierung. Somit ist für den Anwender nur eine einmalige Interaktion zur Anmeldung erforderlich. Beim Wechsel zwischen verschiedenen Systemen übernimmt das Verfahren über eine bestimmte Zeitspanne alle notwendigen Authentifizierungsschritte gegenüber anderen Systemen selbständig. Es sorgt also dafür, dass eine bestätigte Identität automatisch an andere Systeme weitergegeben wird. Diese Systeme vertrauen auf die bestätigte Identität des Single Sign-On-Systems und führen damit keine erneute Authentifizierung durch, sondern evaluieren in einem nachgelagertem Autorisierungsprozess nur noch die Zugriffsberechtigungen. Auch hier existieren unterschiedliche Mechanismen:

- ➢ Der Benutzer meldet sich auf einem sicheren Kommunikationskanal an einem zentralen Dienst (Ticketsystem) an und erhält ein Ticket mit begrenzter Lebensdauer, das ihm den Zugang zu allen Applikationen garantiert. In den meisten Fällen muss in das zu integrierende System noch eine zusätzliche Softwareschicht zur Verifikation von Tickets eingebaut werden.

- ➢ Ein weiteres Verfahren, das hauptsächlich bei der Integration von Backend-Systemen eingesetzt wird, ist die zertifikatsbasierte *Public Key Infrastructure* (PKI) für die sichere und verbindliche Kommunikation und Transaktion über offene Netze. Zwischen den integrierten Systemen wird eine Vertrauensstellung durch Austausch der öffentlichen Zertifikate hergestellt. Die Überprüfung der Identitäten erfolgt nach dem Prinzip der digitalen Signatur (vgl. [Ert01], S. 93ff.).

Auf dieser Basis lassen sich die Rollen und Rechte aller Personen auf dem Campus festlegen, d. h. wer in welchen Bereichen welche Rechte besitzt. Die Rechte sind an die Anwendungen und Werkzeuge weiterzugeben, die in diesem Rahmen eingebunden sind. Somit kann eine solche Diensteinfrastruktur als „organisatorischer Kit“ zur Einbettung mediengestützter Lehr- und Lernprozesse in einen institutionellen Kontext dienen (vgl. [Ker04b], [KS05a], S. 25).

Ausschlaggebend für die Nutzerakzeptanz und nachhaltige Nutzung von Werkzeugen für die mediengestützte Lehre ist dabei ihre Anbindung an Systeme der Hochschulverwaltung zur Lehr- und Veranstaltungsplanung sowie der Prüfungsverwaltung. Nur mit einer geeignet integrierten Lösung, die ein manuelles Abgleichen der Kurs-, Belegungs- und Strukturdaten vermeidet und dadurch zur Verbesserung der Wirtschaftlichkeit der eigenen Verwaltung als auch des Service für Studierende beiträgt, kann die Organisation der Lehre erleichtert und so Kapazitäten des Lehrpersonals für die Lehrinhalte und deren didaktische Aufbereitung freigesetzt werden (vgl. [DEH+02], S. 1ff., [Kub04], S. 12f., [RS05], [MJ05], S. 21). Dabei sind die Daten der Veranstaltungsverwaltung zu übernehmen bzw. die Erstellung des kommentierten Vorlesungsverzeichnisses zu unterstützen, sowie die Anbindung an Prüfungsverwaltungs- und Abrechnungssysteme, um damit die Erstellung von Statistiken zu erleichtern (vgl. [Kub04], S. 13). Solch eine

Kopplung von Verwaltungssystemen und Lernsystemen kann auf verschiedenen Kommunikationsparadigmen beruhen, welche den Komplexitätsgrad der Schnittstellenimplementierung stark beeinflussen[29].

Ebenso wie die Kopplung von Lernsystemen mit Veranstaltungssystemen muss auch die Kopplung von Lerntechnologien untereinander erfolgen, um Medienbrüche in Lernprozessen selbst aufzulösen. Kerres unterstreicht in [KNW03], S. 92f., dass Lehren und Lernen komplexe Aktivitäten sind, für die in einem Lernsystem unterschiedliche Technologien benötigt werden, um diese Aktivitäten angemessen unterstützen zu können. Softwarelösungen im Bereich des computerunterstützenden Lernens implementieren zumeist bestimmte lerntheoretische bzw. didaktische Paradigmen und sind dadurch auf bestimmte Szenarien des Lernens und Arbeitens hin ausgerichtet. Die Lern- und Arbeitsszenarien an Hochschulen sind jedoch vielfältig, und analog dazu sind es auch die Anforderungen an ihre Softwareunterstützung: „Virtual classrooms should be more flexible than their physical counterparts rather than less so. Do you teach art history? Then you need an image annotation tool. But probably a different one than the image annotation tool needed to teach histology. Foreign language teachers may want voice discussion boards to check student accents. Writing teachers should have peer editing tools. History teachers should habe interactive maps. And so on“ [FM06].

Die Idee, dass eine virtuelle Lernumgebung nicht als eine geschlossenes System, sondern als eine Menge gekoppelter Technologien konzipiert werden muss, die sich (nur) für ihre Benutzer als „eine“ Plattform darstellt, ist inzwischen in der Scientific Community weit verbreitet (vgl. bspw. [Ker04b], [Sch04b], [SW04], [RHS05], [FM06], [Lam07], [Kei07]). Dieser Ansatz beeinflusst daher sehr stark die aktuellen Entwicklungen neuer Architekturmodelle für integrierte virtuelle Lernumgebungen, auf die in Abschnitt 2.2 näher eingegangen wird.

Eine weitere wichtige Maßnahme ist die Integration von Bibliotheksdiensten für eine medienbruchfreie und durchgängige Verwaltung digitaler Informationsstrukturen. Die Anbindung von Lerntechnologien an eine digitale Bibliothek bedeutet in diesem Fall eine Menge an Systemkonvergenzen: Dokumente können über die Lernsysteme innerhalb der digitalen Bibliothek gefunden und in Kurskontexte und Lernszenarien eingebettet werden, fertige Kursbestandteile können dann ohne erzwungene Medienbrüche in der digitalen Bibliothek archiviert und eigens erstellte Dokumente dort publiziert werden (vgl. [BHH+06]).

Nachdem die vorhandenen Werkzeuge auf Basis eines übergreifenden Identitäts- und Rechtemanagements in eine gemeinsame Infrastruktur integriert wurden, können einzelne Funktionen mit in eine gemeinsame Darstellungsschicht aufgenommen werden, die dadurch als zentraler Einstiegs- und Informationspunkt dienen kann. Zwar haben die meisten deutschen Hochschulen bereits zentrale Webseiten, auf der Dienste sowie wichtige Informationen für bestimmte Zielgruppen wie Studierende, Lehrende, Verwaltungsmitarbeiter und externe Besucher gebündelt angeboten werden. Auf Grundlage der beschriebenen informationstechnischen Maßnahmen lassen sich darüber hinaus jedoch

[29] Die einzelnen Integrationsebenen und dieser Komplexitätszusammenhang wird in dieser Arbeit auf S. 118f. beschrieben.

personalisierte Hochschulportale realisieren, die Benutzern eine persönliche, individualisierte Sicht auf die Dienste und Informationen bieten können, für die sie als Person und durch die ihnen zugeordneten Rollen autorisiert sind (vgl. [A+07]).

2.1.3. Zusammenfassung

Der effiziente und effektive Umgang mit den quantitativ stark gewachsenen Informationsmengen fordert von den Berufstätigen in der heutigen Wissensgesellschaft Kompetenzen und individuelle Bildungsstrategien, die das staatliche Bildungssystem bislang nur ungenügend unterstützen konnte. Die darauf bezogenen aktuellen Bildungsreformen implizieren konzeptuelle Veränderungen bei der Hochschulbildung, die grundsätzlich sind für die forschungsorientierten und disziplinären Paradigmen, aus denen das an Universitäten traditionell vermittelte Wissen generiert und in denen es organisiert ist. Neben einer neuen Bedeutungszuweisung der akademischen Forschung zur Generierung neuen Wissens und einem als gleichrangig neben Forschung und Lehre gestellten universitären Weiterbildungsauftrag werden Hochschulen insbesondere dazu angehalten, die für die Wissensarbeit notwendigen Kompetenzen durch neue Lehr- und Lernformen sowie praxisrelevantere Fragestellungen und Lösungsansätzen zu vermitteln.

Hierzu wurden u. a. Neue Medien in die Hochschullehre eingeführt, denen durch erweiterte Darstellungs-, Kommunikations- und Aktualisierungsmöglichkeiten neue Qualitäten des Lehrens und Lernens zugesprochen wurden. Gezielte Fördermaßnahmen des Bundes sollten diese in die Breite tragen. Dabei führte die Schwerpunktsetzung auf hochschulübergreifende Verbundprojekte in Kombination mit fehlenden hochschulinternen E-Learning-Strategien und fehlenden Lerntechnologiestandards jedoch an vielen Universitäten zu heterogenen Infrastrukturen, die aus vielen Insellösungen und unkonsolidierten, fragmentierten Prozessabläufen bestehen.

Ein flächendeckender effizienter und alltagstauglicher Einsatz von E-Learning an Präsenzhochschulen erfordert daher eine Ressourcenallokation sowohl aus organisatorischer als auch aus informationstechnischer Sicht. Die folgenden drei Ziele finden sich aus diesem Grund in den meisten aktuellen Medienentwicklungsstrategien deutscher Hochschulen wieder:

- Realisierung effizienter Organisationsstrukturen, mit denen ein breites Spektrum an kunden- und serviceorientierten IuK-Angebot und -diensten für neue Medien in der Lehre zur Verfügung steht

- Optimierung des personellen und finanziellen Ressourceneinsatzes durch Bündelung von Ausstattungen, Investitionen und Kompetenzen sowie Schaffung von Freiräumen für Innovationen und nachhaltige Weiterentwicklung durch höhere Kosteneffizienz und Synergie aller laufenden IuK-Prozesse

- Entwicklung einer attraktiven Infrastruktur für multimediales Lehren und Lernen, die den Anforderungen der Wissensgesellschaft genügt sowie national und international wettbewerbsfähig ist.

Eine für multimediales Lehren und Lernen attraktive Infrastruktur muss dabei Medienbrüche auf zwei Ebenen aufheben, ohne die vorhandene Vielfalt an didaktischen Arrangements einzuschränken (vgl. [RH07]):

1. Die Administrations- und Dispositionssysteme in der Hochschulverwaltung müssen Schnittstellen anbieten, um Einsparungen zusätzlichen Aufwands für den Betrieb der in der Lehre eingesetzten Informationssysteme und Werkzeuge zu ermöglichen

2. unterschiedliche Elemente und spezialisierte Einzelwerkzeuge müssen auf einfachem Weg zu einer hybriden Lernumgebung integriert werden können, welche verschiedenste, individuelle pädagogische und organisatorische Anforderungen von Lehrveranstaltungsformaten und selbstorganisiertem Lernen angemessen unterstützen können.

Im Gegensatz zu Hochschulverwaltungssystemen, die hierzu nur Schnittstellen zu einer kleinen Untermenge ihrer Funktionen nach Außen öffnen brauchen, impliziert dieser Trend für Lerntechnologien jedoch die Umstellung ihrer Softwarearchitektur vom monolithischen System hin zur offenen Architektur und somit einen Fokuswechsel vom Funktionsumfang zur Integrationsfähigkeit. Des Weiteren muss eine gezielte konzeptionelle und technische Konvergenz von Lernwerkzeugen dahingehend erreicht werden, dass Benutzern die selbstbestimmte Interaktion mit Inhalten über Systemgrenzen hinweg ermöglicht wird.

In Abb. 2.2 sind noch einmal die in diesem Abschnitt erarbeiteten Handlungsfelder aufgezählt, in denen an Universitäten aktiv an der Schaffung einer attraktiven Infrastruktur für E-Learning gearbeitet werden muss.

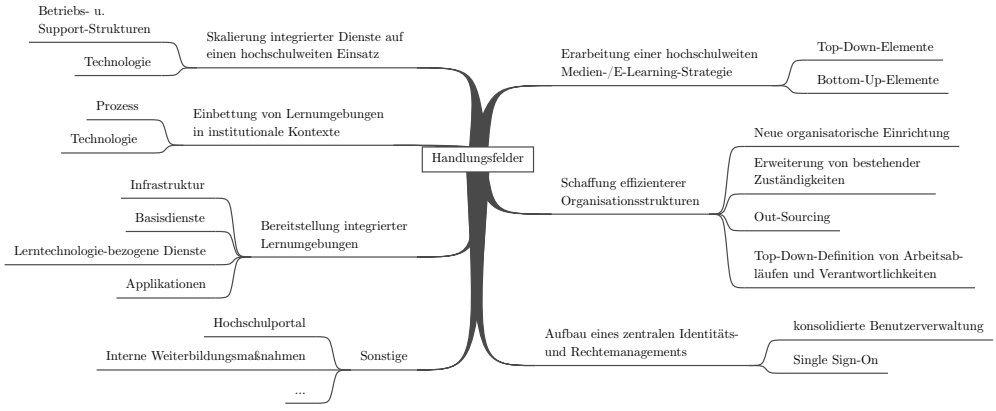

Abbildung 2.2.: Reorganisation und IT-Unterstützung zur Schaffung einer attraktiven universitären Infrastruktur für E-Learning

2.2. Das elektronisch unterstützte Lernen

2.2.1. Begriffliche Einordnung

Bis etwa 1995 fällt E-Learning praktisch mit *Computer-based-Training* (CBT)[30] zusammen. Kennzeichnend für ein CBT ist, dass das Lernsystem und das Kursmaterial lokal auf dem Rechner des Anwenders installiert ist (vgl. [BBSS01], S. 289).

Multimediale Elemente, die durch die zunehmende Leistungsfähigkeit der Computer in den 1990er Jahren ermöglicht wurden, markieren eine Ausbaustufe (*Multimedia-CBTs*). Aber selbst in diesem Szenario ergänzt die neue Lernform die konventionelle, papierbasierte Fernlehre (*Distance Learning*, s. u.) lediglich auf einer medialen Ebene und nicht in einer grundsätzlichen Art und Weise. Weiterhin gilt die These, dass mit dem Ansprechen vieler Sinnesorgane (Hören und Sehen) durch Multimedia generelle Erfolgsverbesserungen beim Lernprozess erzielt werden, mittlerweile als widerlegt (vgl. [Reg04]).

Trotz anfänglicher Euphorie um das CBT leidet das Konzept in der Praxis oftmals an mangelnder Akzeptanz und Motivation bei den Anwendern (s. [MG05], S. 161). In Verbindung mit der administrativen Ineffizienz der Datenträgergebundenheit, der dadurch erschwerten Distribution lerninhaltlicher Aktualisierungen sowie der zwangsläufigen sozialen Isolation der Lernenden[31] ergibt sich, dass CBTs in der reinen Form von einer ehemaligen Dominanz auf dem E-Learning Markt im Weiterbildungssektor zu einer Randerscheinung geworden sind (vgl. [SM02], S. 26 u. [SKRS01], S. 21).

Das *Web-based-Training* (WBT), ein unmittelbar auf der Internettechnologie aufsetzendes Konzept, adressiert wichtige Schwachstellen des CBTs. Es handelt sich bei einem WBT um ein über Server bereitgestelltes und administriertes Lernsystem, mit dem der Anwender über eine zumeist browserbasierte Benutzungsoberfläche in Interaktion tritt. Inhaltlich profitieren WBTs von einer größtmöglichen Aktualität und von einem Ausbruch aus der Isolation des Lerners in eine Welt der „explorativen und kommunikativen“ Erfahrung (vgl. [MG05], S. 189):

- Die *kommunikative Ebene* wird durch die Einbindung von internetbasierter Mensch-zu-Mensch-Interaktion erschlossen, klassischerweise über Foren, Email und Chat[32].

- Die *explorative Erfahrung* wird z. B. durch hypermediale Lernsysteme als Spezialform des WBT gewährleistet. Anstelle der festen Definitionen eines vorgegebenen Lernweges durch die Inhalte nutzen sie das Hypertextkonzept, um dem Anwender

[30] Der Mitte der achtziger Jahre eingeführte Begriff CBT wurde im Allgemeinen als Umschreibung für die Wissensvermittlung per Computer verstanden (vgl. [Bau06a], S. 11). Mit dem WWW hat die Wissensvermittlung eine neue Dimension erreicht. Somit dient der Ausdruck CBT mittlerweile der Abgrenzung gegenüber webbasierter Wissensvermittlung. Als Synonyme werden die Begriffe *Computer Aided/Assisted Learning* (CAL) oder *Computerunterstütztes Lernen* (CUL) verwendet.

[31] Neben dem fehlenden Kontakt zu weiteren Lernenden fehlt auch oftmals eine individuelle Betreuung durch Lehrende.

[32] Kritisch ist hierzu anzumerken, dass der Terminus WBT heute noch überwiegend mit der Mensch-System-Kommunikation assoziiert wird. Dies liegt daran, dass der Ebene der Kommunikation in den Anfangszeiten der WBTs einen geringeren Stellenwert eingenommen hat. Dies ändert sich jedoch spätestens seit der Fokussierung auf benutzergenerierte Inhalte und Kooperationsunterstützung.

ein eigenständiges und flexibles „Erforschen“ zu ermöglichen. Die damit einhergehende mangelnde Sequenzialität bedingt andererseits auch höhere Ansprüche an den Lernenden und birgt die Gefahr der kognitiven Überlastung (*cognitive overload*) und Desorientierung (*Lost in Hyperspace*, vgl. [Ter02]).

E-Learning, Synonym für multimediales Lernen und Telelernen, wird in der Regel als Oberbegriff für alle Varianten von Lernangeboten verwendet, in denen neue Medien eine Rolle spielen. Die große Abdeckung des Begriffs wird durch die folgende Definition veranschaulicht: „E-Learning ist eine durch Informations- und Kommunikationstechnologie unterstützte Form des Lernens“ [BBSS01], S. 35. Es umfasst somit sowohl das Online-Lernen über das Intra- oder Internet in Form von Web-Based-Trainings als auch die Computer-Based-Trainings. Zudem können auch noch differenziertere Technologien und Paradigmen eingesetzt werden, die in verschiedenen Formen des E-Learnings münden (s. nächster Abschnitt).

Zum *Distance Learning* zählen alle Bildungsmaßnahmen, die über eine räumliche Distanz abgehalten werden. Distance Learning ist gleichzusetzen mit dem deutschen Begriff Fernunterricht und umfasst sowohl das E-Learning, das Online-Lernen (WBT) und das CBT als auch traditionelle papierbasierte Formen und Bildungsfernsehen.

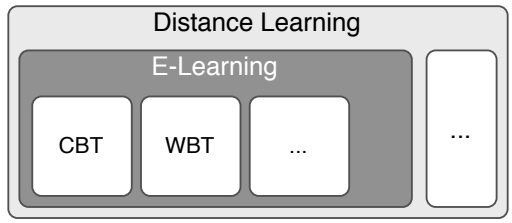

Abbildung 2.3.: Untermengen des Distance Learnings

2.2.2. Formen des E-Learnings

In Theorie und Praxis sind unzählig viele Ausprägungen von E-Learning zu finden. Bislang hat sich in der Literatur jedoch noch kein einheitlicher Ordnungsrahmen zur Kategorisierung dieser Varianten durchgesetzt. Autoren ziehen zur Unterscheidung die verschiedensten Strukturmerkmale heran, wie beispielsweise die Notwendigkeit einer Verbindung zum Internet (vgl. [Kol96], S. 50), die Merkmale Raum und Zeit (vgl. [Alb03], S. 36f.), Funktionen der Medien und Sicht des Lernenden (vgl. [RR03], S. 33ff.), oder das Maß an Selbstregulation[33]. Ley und Schachner unterscheiden in [LS04] anhand des

[33] Die Selbstregulation bezeichnet das Ausmaß an Freiheitsgraden des Lerners auf mehreren Merkmalsdimensionen, wie z. B. Ausmaß der Vertiefung, Ort und Dauer des Lernens, Art der Medien und Sequenz der Inhalte, Auswahl von Beispielen und Inanspruchnahme von Hilfen sowie Lerntechniken und -strategien (vgl. [Nie01], S. 117ff.).

Paradigma	Fokus	Gestaltungsaspekte	Beispiele
E-Training	Content	Instruktionales Design	CBT, WBT
E-Mentoring, E-Tutoring	Feedback	Interaktion Lernender-Lehrender	Hybrid Learning
E-Collaboration	Interaktion	Gestalten von Kommunikationsräumen	Communities of Practice
Ad-hoc-Lernen	Prozess	Prozessanalyse, Lernbedarf	Just in time-Lernmodule
Wissensmanagement	Dokumente	Learning Objects, Suche, Metadaten	Learning by Google
Personal Publishing	Experten	Publikationsplattform	Wiki, Weblog

Tabelle 2.1.: Unterscheidung von E-Learning-Paradigmen nach dem Merkmal der Selbstregulation (aus: [LS04])

mehrdimensionalen Kriteriums der Selbstregulation sechs Paradigmen, die im Folgenden als Beispiel eines möglichst vollständigen Ordnungsrahmens skizziert werden (s. auch Tabelle 2.1).

2.2.2.1. E-Training

Im Mittelpunkt des E-Trainings stehen Inhalte (Content), die weitestgehend Grundlagenwissen darstellen und mittels Instruktion vermittelt werden sollen. Die Durchführung kann isoliert oder als Bestandteil eines Blended Learning-Mixes erfolgen. Der Lernende interagiert dabei ausschließlich mit dem Medium, wobei der Grad der Interaktionsfreiheit je nach System variieren kann.

E-Trainings werden klassischerweise durch CBTs und WBTs implementiert. Bestandteile der Anwendungen sind Interaktionen in Form von Fragen und Feedback, Lernmodule zum Vermitteln von Inhalten und Testmodule zur Evaluation des Lernfortschritts (vgl. [SM02], S. 26). Als konkrete Ausprägungen sind „Drill&Practice Software“ und „Tutorielle Systeme“ zu nennen, bei denen der Lernende eigenverantwortlich und zu individuell gewählten Zeiten lernen kann (asynchrones Lernen).

2.2.2.2. E-Teaching

Dieses Paradigma schließt die betreuende Begleitung von Lernenden durch eine Lehrperson als soziale Komponente mit in den Lernprozess ein. Auf diese Weise kann der Betreuer als reiner Ratgeber oder motivierend agieren, indem er nur auf Anfragen von Lernenden reagiert oder die Lernenden zu einem Dialog auffordert (vgl. [SM02], S. 44f.). Diese Betreuung kann synchron (z. B. Chat) oder asynchron (z. B. E-Mail) erfolgen. Entsprechende Technologien sind in die Lernumgebung mit einzubinden.

2.2.2.3. E-Collaboration

Soziale Lerntheorien gehen davon aus, dass Lernen ein selbstorganisierter und selbststeuernder Prozess von Personengruppen ist, in dem zwar ein für das selbststeuernde Lernen förderndes Umfeld geschaffen werden kann, Lerngruppen aber nicht von Außen organisiert werden können oder sollen (vgl. [NRP00], S. 5). Bei dieser Form des E-Learnings arbeiten die Anwender daher in kleinen Gruppen zusammen an gemeinsamen Zielen und Lernen dabei von- und miteinander durch gemeinsames Erschließen von Wissen. Die externe Anleitung nimmt hierbei ab, wohin dagegen soziale Aspekte stärker gefördert werden sollen wie kulturelle Voraussetzungen, soziale Kontakte und die Anerkennung von Selbstreferenz und -reflexion. Konzepte wir *Knowledge Communities* oder *Communities of Practice* (CoPs, vgl. hierzu [AP00], [Wen98b] und [WS01]) verfolgen diesen kooperativen Ansatz. Technisch kommen dabei CSCW/L-Werkzeuge (*Computer Supported Cooperative Work/Learning*) zum Einsatz, die vor allem das Bearbeiten von Dokumenten in Gruppen, Versionierung und Kommunikation unterstützen.

2.2.2.4. Ad-hoc-Lernen

Zentrales Merkmal dieses Paradigmas ist die Bedarfsorientierung: Kenntnisse und Fähigkeiten werden erst dann neu angeeignet bzw. vertieft, wenn diese benötigt werden (vgl. hierzu *Learning on Demand* bzw. *Just-in-Time-Training* in [SM02], S. 76f.). Der Lernprozess ist dabei so zu unterstützen, dass er sich an individuell gewählten Lernzielen flexibel ausrichten kann[34]. Das dabei zu berücksichtigende Spektrum erstreckt sich dabei von der kontinuierlichen Suche nach Informationen (bspw. Nutzung von Suchmaschinen) hin zu klassischen Schulungen (vgl. [LL04]).

2.2.2.5. Wissensmanagement

Ausgangspunkt eines persönlichen Wissensmanagements ist die Reflexion individueller Denkweisen und Handlungen in dem persönlichen Arbeits- oder Lernbereich, die zur Verbesserung der eigenen Effizienz und der partizipierenden Mitarbeitenden oder -lernenden führen können. Methodische Ansätze des Wissensmanagements implementieren dabei mit Planungs-, Durchführungs- und Kontrollelementen oftmals einen Management-Regelkreis. Die Kernprozesse des Wissensmanagements sind nach [PRR03], S. 28ff., die Identifizierung, der Erwerb, die Entwicklung, (Ver-) Teilung, Nutzung und Bewahrung von Wissen[35].

Beim persönlichen Wissensmanagement erfolgt das Lernen weitestgehend autonom und selbstgesteuert. Neben der Informationssuche im Intra-/Internet („*Learning by Google*“) können in diesem Zusammenhang vor allem Instrumente eingesetzt werden, welche Kreativitätstechniken (*Mind-Maps/ Topic-Maps*) und/oder semantisches Arrangieren von Informationen unterstützen.

[34] Im betrieblichen Umfeld bedeutet dies beispielsweise, dass die Vorgänge zur Weiterbildung von Mitarbeitern an Stellen zu koppeln sind, also idealerweise arbeitsintegriert stattfinden.

[35] Coenen hat in [Coe02], S. 19ff., die Bausteine des Wissensmanagements an die Besonderheiten des Wissensmanagements von Lernenden an einer Universität angepasst und erweitert; insbesondere hat er dabei einzelne Bausteine detaillierter betrachtet und mit den beiden zusätzlichen Bausteinen Wissensdemonstration und -diagnose universitäre Prozesse wie Leistungsprüfung und Zertifizierung abgebildet.

2.2.2.6. Personal Publishing

Neue Werkzeuge wie Weblogs und Wikis entfernen technische Barrieren, um online zu Schreiben und zu Veröffentlichen. Ein jeder kann heutzutage über Web-Dienste und *Personal Publishing*-Systeme wie Weblogs eigene Inhalte generieren und (ver-)teilen, mit Metadaten auszeichnen und darüber auch soziale Kontakte und Netzwerke aufbauen (vgl. [Mos06], S. 99). Im Mittelpunkt des Personal Publishing stehen Wissensträger, die ihr eigenes Wissen, ihre eigenen Erkenntnisse und Ansichten als so genannten *Microcontent*[36] verfassen und publizieren. Durch Formen der Kommentierung können diese Inhalte in einer Gemeinschaft aktiv diskutiert und semantisch strukturiert werden. Somit bleibt das Wissen nicht träge, sondern kann sich im Dialog entwickeln. Diese und weitere lerntheoretische Aspekte werden in dieser Arbeit ab S. 79ff. behandelt.

2.2.2.7. Blended Learning

Eine Vermischung der hier vorgestellten Paradigmen, Formen und Konzepte des computerunterstützten Lernens untereinander, aber auch mit Methoden des Präsenzlernens ist Ausgangspunkt für das *Blended Learning*. Als eine Form des hybriden Lernens[37] steht es für einen Methodenmix, in dem sämtliche didaktische Instrumente – Fernunterricht oder Face-to-Face, elektronisch oder nicht-elektronisch, individuell oder kooperativ etc. – zu einem integrierten pädagogischen Gesamtkonzept zusammengefügt werden können (vgl. [BBSS01], S. 217, [Rep02]). Somit soll der Lernprozess an sich begünstigt sowie verschiedenen Lerntypen und Lebenssituationen individuell gerecht werden. Dabei können auch solche Lernformen Berücksichtigung finden, die zunächst antiquiert erscheinen, etwa statische Ressourcen wie reine Texte oder CBTs (vgl. [BBSS01], S. 224).

2.2.3. Technologien virtueller Lernumgebungen

Informationsarchitekturen virtueller Lernumgebungen, wie sie in Hochschulen eingesetzt werden, bestehen aus verschiedensten Technologien. Diese können in drei Klassen Basistechnologien, Lerntechnologien und Lernsystemen eingeordnet werden, die aufeinander aufbauen (vgl. [BBSS01], S. 208ff.). Dabei ergibt sich die Funktionalität der höheren Klasse jeweils durch die Integration und Nutzung von Technologien der darunter liegenden Klassen[38].

Auf der untersten Ebene befinden sich Informations- und Kommunikationssysteme, die bzgl. ihrer Anwendung und Integration grundsätzlich indifferent sind und nicht auf den Bereich elektronisch vermittelten Lernens ausgerichtet. Diese so genannten *Basistechnologien* bilden die Grundlage übergeordneter Technologien und Anwendungssystemen (vgl. [BBSS01], S. 208, [Nie04], S. 254ff.). Zu ihnen zählen hauptsächlich Technologien

[36] Mosel beschreibt den Begriff in [Mos06] wie folgt: „Important (formal) aspects of microcontent are that it is referable, can be machine-readable through metadata [...] and is generally focused on one or few single ideas or topics“.

[37] Ebenfalls eine Bezeichnung für die Kombination unterschiedlicher Lernformen, wobei sich ein optimaler Mix auf unterschiedliche Perspektiven beziehen kann, z. B. Technologie-Mix, Methoden-Mix etc. (vgl. [SM02], S. 23f.).

[38] Das Konzept dieser Mehrschicht-Architekturen wird ausführlicher in Kapitel 4 vorgestellt.

für die Funktionsbereiche Kommunikation (bspw. E-Mail/Chat/VoIP), Informationsbeschaffung (z. B. Suchmaschinen), Administration, Produktion und Evaluation. Aber auch hardwareseitige bzw. hardwarenahe Technologien wie bspw. Internetzugang und Systemsoftware sind auf dieser Ebene einzuordnen.

Die Funktionen von *Lerntechnologien* sind speziell für den Lernbereich entwickelt und orientieren sich an bestimmten Lernprozessen. Hierbei werden Basistechnologien auf die Domäne des elektronisch unterstützten Lernens angewendet und stellen beispielsweise Hilfs- und Kommunikationswerkzeuge, Tools zur Kooperation oder Kursadministration, spezielle Suchmaschinen und Programme zur Medienproduktion (z. B. Grafik-, Audio- und Video-Bearbeitungsprogramme) dar. Die Menge an Anwendungsgebieten scheint unendlich groß, so dass die in der Tabelle 2.2 aufgeführten Typen multimedialer Lerntechnologien nur einen kleinen, aber wichtigen Ausschnitt davon abbilden.

Während die Integration von Basistechnologien zu einer Lerntechnologie durch die Festlegung auf einen spezifischen Anwendungsbereich bestimmt wird, ist die Integration von Lerntechnologien zu einem *Lernsystem*[39] vor allem durch die Konkretisierung des Anwendungskontextes mit Hilfe von Daten und Inhalten gekennzeichnet, die sowohl Inhalte beinhalten, als auch didaktische Konzepte und Methoden implementieren (vgl. [Nie04], S. 253). Jede in der Praxis eingesetzte E-Learning-Anwendung respektive Kombination von Anwendungen stellt somit in sich und mit seiner Umwelt ein Lernsystem dar.

2.2.4. Standards und aktuelle Integrationsansätze

Ein virtuelles Lernsystem respektive eine virtuelle Lernumgebung besteht aus einer oder mehr Lerntechnologien, die im Sinne einer bestimmten Didaktik kombiniert und genutzt werden können. Unter dem Gesichtspunkt der Interoperabilität haben sich im E-Learning bestimmte Spezifikationen, Normen und Standards[40] durchgesetzt, die beschreiben, wie verschiedene Lerntechnologien sowohl innerhalb virtueller Lernumgebungen als auch umgebungsübergreifend miteinander operieren können. Erst unter diesen Voraussetzungen rentiert sich bspw. die mit hohen Kosten verbundene Erstellung von WBTs[41].

Im Folgenden werden Spezifikationen, Normen und Standards klassifiziert, die auf die Entwicklung von Lerntechnologien und -systemen angewendet werden können. Mittlerweile weiten sich die Standardisierungsbemühungen auf den Bereich der didaktischen und methodischen Gestaltung von virtuellen Lernszenarien aus (vgl. [AKTZ04], S. 217).

[39] In dieser Arbeit wird der Begriff *Lernumgebung* als Synonym zum Lernsystem verwendet.

[40] Spezifikationen spiegeln dabei erste Harmonisierungen in kleineren Anwendungsgemeinschaften und können als Vorstufen der Normen gesehen werden, welche wiederum Spezifikationen innerhalb größerer Gemeinschaften vereinbaren. Eine Norm ist in der deutschen Sprache eher ein Regelwerk, während ein Standard eher eine Qualitätsaussage darstellt, also bspw. wie eine Norm umzusetzen ist (vgl. [Lin04], S. 341ff.). Das British Standards Institute liefert auf ihrer Webseite eine klarere Definition eines Standards: „document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievment of the optimum degree of order in a given context."

[41] Die Herstellungskosten für eine Stunde interaktiven, didaktisch aufbereiteten Content reichen von 2.000 bis zu 20.000 Euro und mehr, je nach Grad der Multimedialität und des Themengebietes (vgl. [Häf02]).

Anwendungstyp	**Merkmale**	**Haupteinsatzgebiete**
Übungsprogramm	sequentielle Abfrage von Bekanntem, bzw. Feststellung von Defiziten mit Feedback	Übungs- u. Anwendungszwecke (deklarative Lernziele)
Animation & Simulation	Abbildung komplexer dynamischer Zusammenhänge in einem (interaktiven) Modell, u. U. mit der Möglichkeit, die Auswirkung von Parameter- u. Variablenmanipulation zu beobachten	exploratives Lernen, learning by doing (prozedurale Lernziele), 3D-Modelle
computerunterstützte Planspiele	meist in Form von Wirtschafts-, Sozial- oder Natur-Simulationen, in denen der Teilnehmer selbst Teil der Simulation ist; bis hin zu Mikrowelten, in denen der Lernen einen Realitätsbereich selbst modelliert	Unterstützung eines Verständnisses für Zusammenhänge (vernetztes Denken); Aufbau und Handelssimulation
Hypermediale Infosysteme	elektronisches Handbuch (Indexsystem, Lexikon) mit Suchfunktion und oft hypermedialen Elementen	geordnete Informationspräsentation (deklarative Lernziele)
Tutorials, guided tours	angeleitetes Lernen durch im Programm festgelegte Lernwege, die Reihenfolge der Lerninhalte kann jedoch innerhalb eines gewissen Rahmens selbst bestimmt werden	Einführung in neue Wissensgebiete und in die Bedienung von Programmen
intelligente tutorielle Systeme	interaktive Unterstützung des Lernenden beim Aneignen von Neuem durch a) Feedback, dass über „falsch/richtig“ hinausgeht, b) systemgesteuerte Auswahl des als nächstes zu präsentierenden Inhalts in Abhängigkeit von der Lösung der vorausgehenden Aufgabe durch den Lerner sowie sonstiger Einschätzung des Lernenden (z. B. hinsichtlich Vorwissen, Fähigkeiten)	Vertiefung des Wissens (deklarative Lernziele), adaptive Informationssysteme, Datenbanken
WBTs ohne/mit direkter sozialer Interaktion	Lern- und Informationssysteme, die ggf. eine Vernetzung mehrerer Lerner ermöglicht. Zur Verfügung stehen können Chat, Whiteboard, Application Sharing und Video-Conferencing	soziale Interaktion in virtuellen Seminaren, netzbasierte globale Zusammenarbeit (CSCL/W), virtuelle Klassenräume

Tabelle 2.2.: Beispiele für Anwendungstypen multimedialer Lernsysteme (in Anlehnung an [Kal03], S. 66)

2.2.4.1. Klassifikation von Spezifikationen und Standards

Euler und Seufert haben in [ES05], S. 458, vier Klassen von Standards bzw. Spezifikationen unterschieden in Technologie-, Prozess-, Lerntechnologie- und QM-/QS-Standards. Diese Kategorisierung wird in diesem Abschnitt noch ergänzt um eine Klasse, die spezifizierte Frameworks und Referenzmodelle enthält, die zum Softwareentwurf von Lernsystemen und -technologien verwendet werden können (vgl. Abb. 2.4):

- Bei den *Technologiestandards* handelt es sich um Standards für Basistechnologien, auf die bei der Entwicklung und beim Betrieb von Lerntechnologien zurückgegriffen werden kann. Diese Kategorie umfasst unter anderem Webtechnologien, Protokolle oder Übertragungsformate wie *Extensible Markup Language* (XML) oder *Real Simple Syndication* (RSS), aber auch Audio- und Videoformate.

- *Lerntechnologiestandards/-spezifikationen* betreffen die Komponenten computergestützter Lernszenarien (z. B. Inhalte, Methoden, Benutzermodelle) und zielen auf Interoperabilität und Wiederverwendbarkeit von Lernumgebungen und -inhalten ab. Soweit sie auf quelloffenen Dateiformaten basieren, verhindern sie die Bindung von Nutzern an einzelne Plattformen (*Locked-In*) und sind Voraussetzung für einen Wettbewerb unter den Systemherstellern und für wettbewerbsfähige Open-Source-Produkte.

- *Architekturspezifische Frameworks und Referenzmodelle* zielen auf ein gemeinsames Verständnis bzgl. der Informations- und Softwarearchitektur von Lernsystemen und -technologien ab. In diesem Rahmen werden von verschiedenen Standardisierungsgremien aktuell die in der E-Learning-Domäne geläufigsten Prozesse und Dienste definiert, abstrakte (formale) Modelle vom Anwendungsverhalten und gemeinsame Datenstrukturen festgelegt, sowie gemeinsame Schnittstellendefinitionen und Infrastrukturen verabschiedet. Das Ziel dabei ist zum einen eine Kostenreduktion bei der Modellerstellung beim Entwurf und der Implementierung von E-Learning-Software durch Wiederverwendung, zum anderen die generelle Beschreibung von Lernsystemen und -technologien als Standard.

- Nach [ES05] bilden *QM- und QS-Standards* den Rahmen für den Einsatz von Standards in Organisationen, jedoch haben sich im Bereich von Qualitätsmanagement und -sicherung bisher noch keine Standards durchsetzen können[42]. Internationale Standardisierungsbestrebungen stellen aber bereits ein Rahmenwerk für die individuelle Qualitätssicherung bereit. Von deutscher Seite wurde ein Referenzmodell und Referenzkriterien eingebracht (DIN PAS 1031-1, vgl. [DIN04a] und [EGHP05], S. 59), welche Prozesse der Planung, Entwicklung, Durchführung und Evaluation von Bildungsprozessen und Bildungsangeboten unter besonderer Berücksichtigung von E-Learning beschreiben.

[42] Zwar gibt es Normen wie die ISO 9000:2000 (`http://www.quality.de/lexikon/iso_9000_2000.htm`, letzter Zugriff: 31.07.08), dennoch ist diese nicht für alle Qualitätsziele respektive alle Organisationsarten geeignet und eben kein Standard (vgl. [EGHP05], S. 57).

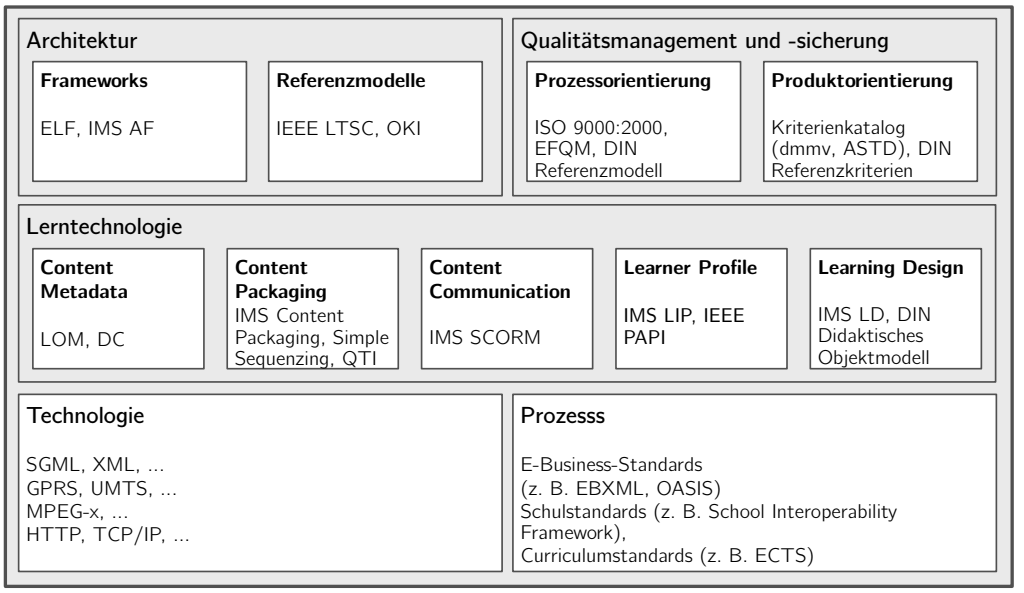

Abbildung 2.4.: Spezifikationen und Standards im E-Learning

➢ Lernumgebungen werden zumeist in bestimmten organisatorischen Kontexten betrieben, wie in Hochschulen, in Unternehmen oder zur Unterstützung kommerzieller Weiterbildung über Internetportale. Insofern sind auch Standards relevant, die für verwandte Prozesse und Systeme entwickelt wurden. Dazu gehören *Prozessstandards* bestimmter Bildungssysteme[43], aber auch aus dem Bereich des E-Business[44].

Auf den folgenden Seiten werden die beiden für das Forschungsumfeld dieser Arbeit besonders interessanten Kategorien *Lerntechnologiestandards* und *Architekturspezifische Frameworks und Referenzmodelle* noch einmal ausführlicher behandelt.

2.2.4.2. Lerntechnologiestandards

Die für den Bereich des E-Learnings spezifischen Lerntechnologiestandards, die die Sicherstellung der Interoperabilität und Wiederverwendbarkeit virtueller Lernressourcen für Lehrende, Lernende und Entwickler fokussieren, werden durch ein Kooperationsnetzwerk von Standardisierungsgremien und -projekten spezifiziert. Verschiedene Varianten dieser Spezifikationen werden von Normungsinstitutionen wie der internationalen

[43] Eine Spezifikation zu diesem Zweck ist die *Schools Interoperability Framework Implementation Specification* (SIF), die Bereiche der Prüfungsverwaltung, Studentenadministration und Studenteninformation koordiniert (vgl. [SIF07]).

[44] Zum Beispiel die *Electronic Business using XML* (ebXML, s. `http://www.ebxml.org/`), deren Ziel eine Standardisierung von Datenübertragungsschnittstellen im gegenseitigen elektronischen Datenaustausch im Kontext verschiedenster Geschäftsprozesse darstellt.

ISO/IEC/JTC1[45], der europäischen CEN/ISSS[46] oder dem Deutschen Institut für Normung e. V. (DIN) harmonisiert, autorisiert und herausgegeben.

Die wichtigsten Standardisierungsgremien und -projekte sind:

IEEE LTSC Das *Learning Technology Standards Committee* der IEEE entwickelt auf Basis einer Systemarchitektur *Learning Technology Systems Architecture* (LTSA, siehe S. 44) abstrakt-konzeptuelle Spezifikationen für verschiedene Teilbereiche des E-Learnings. Sie besteht aus Arbeitsgruppen, die – aufgeteilt auf verschiedene Themenbereiche – Spezifikationen der verschiedenen Konsortien prüfen und Standards verabschieden bzw. Vorschläge für amerikanische und internationale Standards bei ANSI oder ISO einreichen.

IMS Seit seiner Aufstellung als Educom-Projekt in 1997 hat das stark auf amerikanische Bildungsverhältnisse ausgerichtete *Instruction Managed Systems Global Learning Consortium* Zulauf von den größten Lerntechnologielieferanten, -verlagen und -endnutzern bekommen. Es stellt z. Zt. im Bereich der Interoperabilität das fortgeschrittenste Projekt dar. Die praxis- und anwendungsorientierteren Spezifikationen des *Instructional Management Systems*-Projekt orientieren sich größtenteils an den Modellen der LTSC, beinhalten jedoch auch Lösungsvorschläge zur Implementierung sowie detaillierte Vorgehensrichtlinien (vgl. [Paw01], S. 97). IMS-Konzepte wie das Learning Design (S. 43) sind zwar – Stand heute – noch in wenigen bis keinen Produktivsystemen integriert, werden aber von vielen Herstellern aufmerksam verfolgt und im akademischen Bereich[47] experimentell verwendet (vgl. [WK07]).

ADLNet Die Anwendungsgemeinschaft *Advanced Distributed Learning Network* sollte – gesponsort von der Amerikanischen Regierung – im großen Umfang die Entwicklung von dynamischer und rentabler Lernsoftware vorantreiben sowie deren Hersteller dazu bewegen, die Anforderungen der Belegschaft von Militär und staatlichen Einrichtungen an Erziehung und Training zu berücksichtigen. Das ADLNet ist bekannt für seine Empfehlung des *Sharable Content Object Reference Model* (SCORM, S. 42), mit denen die US-Militärs untereinander alle Lerninhalte nutzen, austauschen, nachverfolgen und wieder verwenden können.

[45] Die *International Standards Organization* ist ein Netzwerk von nationalen Standardisierungsinstituten aus 140 Ländern, das mit internationalen Organisationen, Regierungen, der Industrie, Herstellern und Verbrauchern zusammenarbeitet. In der *International Electrotechnical Commission* (IEC) arbeitet das *Joint Technical Committee 1, Subcommittee 36* (JTC1/SC36) an internationalen Normen im Umfeld E-Learning, Erziehung und Training.

[46] Das *Centre de European Normalisation/the Information Society Standardisation System* (CEN/ISSS) erhielt im Jahre 1999 von der Europäischen Union das Mandat, einen Arbeitsplan für Interoperabilität im E-Learning für Europa zu erarbeiten und ist somit Träger der Normierungsaktivitäten in Europa. Das erklärte Ziel ist dabei die Sicherstellung, dass jeder Standard europäische Anforderungen berücksichtigt.

[47] Vorreiter ist hierbei die Open University Netherlands (OUNL), aus dessen entwickelter *Educational Modeling Language* (EML, vgl. [Kop01]) bei der IMS das Trägerformat Learning Design (IMS LD) erwachsen ist.

ARIADNE Die *Association of Remote Instructional Authoring and Distributon Networks for Europe* entstand aus einem Förderprogramm der Europäischen Union und hat einen europäischen Wissenspool virtueller Lernressourcen aufgebaut und entsprechende Entwicklungswerkzeuge zur Verfügung gestellt. Neben der Entwicklungsarbeit nimmt das Projekt auch Einfluss auf Standardisierungsbestrebungen[48].

AICC Das *Aviation Industry CBT Committee* stellt Richtlinien für die Entwicklung und Evaluation von E-Learning-Technologien im Bereich der Luftfahrt auf. Diese *AICC Guidelines and Recommendations* (AGRs) beschreiben die Anforderungen an Struktur und Interoperabilität in Bezug auf Bildungsprodukte mit definierten Lernzielen und werden an die Arbeitsgruppe 11 des IEEE LTSC für *Computer Managed Instruction* (CMI) weitergereicht.

Die Spezifikationen und Standards, die von den o. g. Konsortien und Projekten erarbeitet wurden, fokussieren jeweils unterschiedliche Ebenen der Standardisierung. In Anlehnung an [ES05] lassen sich demnach Standards bezogen auf Inhalteverwaltung und -distribution, Lernmanagement, Lernerprofile und Didaktik unterscheiden, die im Folgenden kurz skizziert werden.

Content Metadata *Inhaltliche Standards* haben das Ziel, die system- und anwendungsunabhängige Sicherstellung der Wiederverwendbarkeit, Anpassbarkeit und Rekombinierbarkeit von Lernressourcen zu gewährleisten. Dazu werden die Lernressourcen mit Hilfe von Metadaten beschrieben. Um passende Lerninhalte aufzufinden und diese in anderen Lernszenarien wieder zu verwenden, können die Beschreibungen von Entwicklern oder Lehrenden durchsucht werden. Des Weiteren können Lernressourcen dadurch in Weiterbildungsdatenbanken strukturiert abgelegt werden, was neue Vertriebswege und Marketingkanäle ermöglicht (vgl. [ES05], S. 459). Der in diesem Bereich am meisten verbreitete Standard ist die *Learning Object Metadata* (LOM). Die Wiederverwendbarkeit auf Basis von Metadaten hängt maßgeblich davon ab, ob die Lernressourcen angemessen mit Klassifikationen und Schlagworten beschrieben wurden. Weitere Standards: Dublin Core zur Beschreibung von Dokumenten und anderen Online-Ressourcen, SQI (*Simple Query Interface*) als Schnittstellenspezifikation zur Konfiguration und Übertragung von Anfragen an Lernrepositories.

Content Packaging Spezifikationen zum *Content Packaging* beschreiben Kursstrukturen in standardisierter Form (*Packages*), die ihrerseits wieder Lerninhalte referenzieren können. Die Packages umfassen neben den Dateien der Lerninhalte auch ein Manifest, welches Metadaten, die Struktur einzelner Einheiten und den Verweis auf die Ressourcen enthält. Weiterhin umfassen die Spezifikationen auch Regeln für die Auslieferung der Inhalte und Aspekte der Evaluation. Beispiele: IMS Content Packaging, IMS Simple Sequenzing.

[48] In diesem Zusammenhang hat der Verband sein entworfenes Metadaten-Schema zur Beschreibung von Lerninhalten mit den Spezifikationen der IMS abgestimmt und zusammen an die IEEE weitergeleitet. Seitdem ist die Arbeitsgruppe 12 des LTSC zuständig für weitere Spezifikationen der *Learning Object's Metadata* (LOM).

Learning Management *Management-Standards* behandeln die Interoperabilität administrativer Systeme (z. B. Lernmanagementsysteme) und Lernressourcen. In diesem Zusammenhang ist SCORM hervorzuheben, das als Referenzmodell zur Integration unterschiedlicher Lerntechnologiestandards zunehmend an Verbreitung gewinnt (vgl. [ES05], S. 463)[49]. SCORM besteht seit der Version „SCORM-2004“ aus drei wesentlichen Komponenten:

Content Aggregation Model (CAM) beschreibt die Verwendung von Ressourcen als *Packages*, sowie ihre strukturisierte Ablage in *Organisations*. In Analogie zum Content Packaging (s. o.) werden die Ressourcen und das Paket via Metadaten im XML-Format beschrieben.

Runtime Environment (RTE) Um die Definition von Lernzielen zu ermöglichen, Lernfortschritte zu dokumentieren und eine Adaption von Darstellung und Inhalten an bestimmte Lernpräferenzen des Benutzers zu unterstützen, stellt die RTE eine Schnittstellendefinition zwischen dem Lernmanagementsystem und einzelnen Lerneinheiten zur Verfügung. Diese enthält einen Startmechanismus (*Launch*), eine gemeinsam zu nutzende Funktionsschnittstelle (API) zur Kommunikation zwischen LMS und Lerninhalt, sowie eine Spezifikation des Datenaustauschs zwischen LMS und Lerninhalt.

Sequenzing & Navigation (SN) Die SN-Spezifikation beschreibt so genannte Aktivitätenbäume, die alle möglichen Sequenzen zur Darstellung von Lerninhalten in Abhängigkeit von Aktionen des Benutzers definieren.

Eine maßgebliche Schwäche von SCORM ist die noch fehlende Integration didaktischer Konzepte. Trotz dessen kann die Verwendung von SCORM zu einer Verbesserung der Investitionssicherheit an Hochschulen führen (vgl. [ES05], S. 465). Weitere Beispiele in dieser Kategorie: Die Spezifikation *Computer Managed Instruction* (CMI) zum Austausch und zur Kombination und Administration von Kursen über Lerntechnologien hinweg soll zentrale Administrations- und Steuerungssysteme ermöglichen. Ein weiterer Standard ist die *Question and Test Interoperability* (QTI), welcher sich mit der Austauschbarkeit von Lernerfolgsüberprüfungen befasst.

Learner Profile *Aktorenorientierte Standards* unterstützen die Erfassung und konsistente Nutzung von Daten über Lernende und Lernprozesse. Diese Daten können dann zur Anpassung einer Lernumgebung, oder aber auch zur Unterstützung organisatorischer Prozesse (Klausuranmeldungen, Einreichung von Bewerberprofilen, Anlegen von Personalstammdaten) genutzt werden (vgl. [ES05], S. 461). Auf dem Gebiet der aktorenorientierten Standards gibt es zwei konkurrierende Ansätze: Das *Learner Information Package* (IMS LIP) und *Public and Private Information for Learners* (IEEE PAPI). Durch die Etablierung des LIP als nationalen Standard in Großbritannien zeichnet sich

[49] Ein Grund dafür ist sicherlich, dass sich die maßgeblichen Standardisierungsgremien (u. a. IEEE, IMS und AICC) an der Entwicklung dieses Standards mit beteiligt haben.

eine höhere Verbreitung und Akzeptanz für diesen Standard ab. IMS LIP umfasst unterschiedliche Kategorien, mit denen die Zielsetzungen, Aktivitäten, formale und informale Lernerfahrungen, Präferenzen oder auch Zugangsmöglichkeiten eines Lernenden beschrieben werden können. Daten über den Lernenden können somit sukzessive ergänzt und fortgeschrieben und somit auch für informelles Lernen im Rahmen des lebenslangen Lernens genutzt werden[50].

Didaktische Standards Zielsetzung didaktischer Standards ist eine Beschreibung didaktischer Konzepte und Methoden, die die Entwicklung und Implementierung von Lerntechnologien betreffen. Insbesondere adressieren sie die Darstellung didaktischer Methoden sowie die Zuordnung von Lernressourcen zu didaktischen Kontexten, die durch die derzeitigen Metadatenstandards wie LOM nicht adäquat repräsentiert werden (vgl. [Bau06b]).

Auf Basis der *Educational Modeling Language* (EML) wurde die Spezifikation des *Learning Designs* (IMS LD, vgl. [IMS03b]) entwickelt, die bereits international viele Anhänger gefunden hat. Sie beschreibt ein Aktivitäten-orientiertes Metamodell zur pädagogischen Modellierung von Lernumgebungen, das die Ausstattung und den Ablauf von didaktischen Szenarien umfasst. Da es nur Informationen auf Ebene der Prozesssteuerung enthält und keine über die ihnen zugrunde liegenden pädagogischen Ansätze, ist IMS LD in diesem Sinne didaktisch neutral (vgl. [Bau06b], S. 241). Trotzdem stößt die Modellierung didaktischer Konzepte auch auf Widerstände. Das liegt daran, dass didaktische Standards oft als „... Vereinheitlichung der Didaktik oder Einschränkung der didaktischen Freiheit“ [ES05], S. 466, verstanden werden.

Eine weitere Spezifikation ist die DIN PAS 1032-2:2004, welche ein didaktisches Objektmodell zur Modellierung und Beschreibung didaktischer Szenarien definiert (vgl. [DIN04b]).

2.2.4.3. Architekturspezifikationen

Ein *Framework* liefert ein Vokabular, mit dem sich wiederkehrende Konzepte und integrierende Umgebungen modellieren lassen. Analog zum *Entwurfsmuster-Konzept* in der Softwareentwicklung (vgl. [GHJV96]) ist der vorrangige Fokus von Architekturframeworks daher nicht technischer Art, sondern auf ein gemeinsames Verständnis von architektonischen Sachverhalten auf der Basis eines gemeinsamen Vokabulars ausgelegt. Architekturframeworks im E-Learning zielen also nicht auf die Entwicklung eines generischen Lernsystems oder -technologie ab, sondern sie bieten allgemeine Definitionen von Komponenten an, die im Rahmen unterschiedlicher organisatorischer Ziele und Bedingungen dazu genutzt werden können, passende Systeme und Technologien für die E-Learning-Domäne zu entwickeln. Auf der Basis einer Auswahl von in einem oder mehreren Frameworks definierten Komponenten beschreibt eine Referenzarchitektur eine bestimmte Klasse von Lernsystemen oder -technologien idealtypisch. Weiterhin umfasst eine Referenzarchitektur Beschreibungen oder Regeln über das idealtypische Zusammenspiel dieser Komponenten im Hinblick auf die speziellen Anforderungen dieser Klasse.

[50] Hierbei ist allerdings zu beachten, dass sich solche Standards langfristig erst dann in der Breite durchsetzen werden, wenn entsprechende Sicherheitskonzepte bezüglich der Nutzerdaten bereitgestellt werden (vgl. [ES05], S. 463).

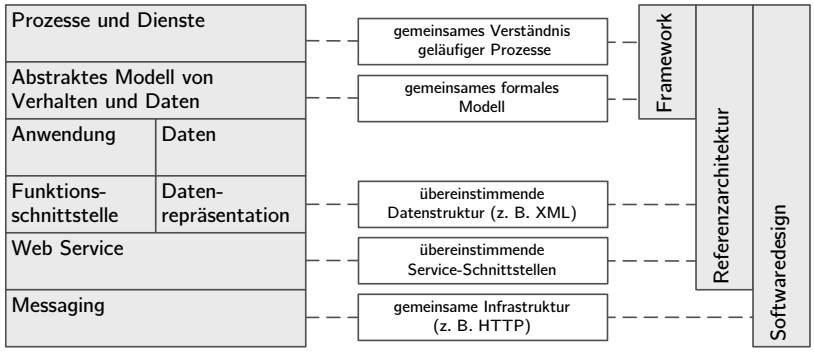

Abbildung 2.5.: Die Rolle von Frameworks und Referenzarchitekturen in Spezifikationsvereinbarungen, in Anlehnung an [WBR04], S. 15.

Durch diese Referenzfunktion können von einer allgemeinen Referenzarchitektur spezielle Entwurfsmodelle, ein so genanntes *Softwaredesign*, für die konkrete Umsetzung eines Systems oder einer Technologie abgeleitet werden. Auf den folgenden Seiten werden mehrere Architekturmodelle vorgestellt, die für die Domäne E-Learning wegweisend sind, da an ihnen mehrere Standardisierungsprojekte mitgewirkt haben[51].

Learning Technology Systems Architecture (LTSA) Die LTSA wurde innerhalb der IEEE LTSC durch die Arbeitsgruppe 1 entwickelt. Die Spezifikation beschreibt Lernsysteme auf einem sehr abstrakten Niveau, das weder Vorschläge zur Implementierung umfasst, noch von pädagogischen Konzepten abhängig ist. „This Standard (1) provides a framework for understanding existing and future systems, (2) promotes interoperability and portability by identifying critical system interfaces, and (3) incorporates a technical horizon (applicability) of at least 5-10 years while remaining adaptable to new technologies and learning technology systems“ ([IEE01], S. 7). Das Architekturframework umfasst demnach keinerlei Implementierungsrichtlinien und keine bestimmten Ausprägungen seiner Komponenten, womit die Nutzung als Referenzarchitektur ausgeschlossen ist. Es bietet allerdings einen Bezugsrahmen für weitere Standardspezifikationen und Referenzarchitekturen von Lerntechnologien, wie bspw. von Tyler in [Tyl03] skizziert.

Die LTSA definiert fünf Architekturebenen, von denen nur die dritte Ebene für den Standard maßgeblich ist und im Rahmen der Spezifikation näher beschrieben wird. Sie beschreibt aktive Systemkomponenten/Teilsysteme (*process*), Informationsflüsse zwischen Komponenten (*flows*) sowie Funktionen zur Persistierung von Informationen (*stores*) (vgl. [IEE01], S. 21ff.). Andere Ebenen wie Entwurfsfunktionen, Anwendersichten und Implementierungsvorschläge bleiben unspezifisch.

[51] Andere Arbeiten in diesem Bereich werden aufgrund ihrer Spezialisierung auf bestimmte didaktische Konzepte – insbesondere auf intelligente tutorielle Systeme – an dieser Stelle nicht weiter berücksichtigt (z. B. [IM94], [MTH98]), [HD02], [HM05] etc.).

E-Learning Framework (ELF) Das *E-Learning Framework* (ELF) des englischen *Joint Information Service Committee* (JISC) und des australischen *Department of Education, Science and Training* (DEST) basiert auf einer Sammlung verteilter Dienste, die zur Umsetzung von Lernsystemen, E-Learning-Portalen und anderen, insbesondere dienstorientierten Informationsarchitekturen unterstützend eingesetzt werden können: „The ultimate aim of the Framework is, for each identified service, to be able to reference an open specification or standard that can be used to implement the service, and also to be able to provide open-source implementation toolkits such as Java and C# code libraries to assist developers“ [JIS08].

Jeder Dienst wird dabei mit Hilfe einer Servicespezifikation beschrieben, die mindestens folgende Information bereitstellen (vgl. [CET05], ServiceSpecification):

1. Eine Beschreibung des Dienstes und seine Rolle innerhalb des Frameworks
2. Eine Menge an Schnittstellendefinitionen oder Empfehlungen entsprechender Spezifikationen
3. Eine Menge an Datentyp-Definitionen oder Empfehlungen entsprechender Spezifikationen
4. Unterstützung der Implementierungstechnologie wie z. B. Java-Schnittstellen und -Klassen für J2EE, COM-Schnittstellen und -Klassen für .net oder WSDL-Schnittstellen und XML-Schemata für eine Web-Service-Implementierung.

Die Dienste sind in vier Schichten unterteilt:

User Agents interagieren direkt mit dem Anwender, wie beispielsweise Autorenwerkzeuge und Portale. User Agents können entweder sehr kompakt sein und sich auf einen bestimmten Bereich fokussieren, oder mehrere Prozesse umspannen, die in einem einheitlichen Arbeitsfluss dargestellt werden und so homogen erscheinen.

Application Services beinhalten Funktionen, die von User Agents benötigt werden. Beispiele dafür sind das Abfragen von Benutzerinformationen oder die Speicherung von Daten. Die Hauptanforderungen an Application Services ist einerseits die Offenlegung ihrer Funktionalität, um von beliebigen User Agents oder anderen Application Services wieder verwendet werden zu können. Und andererseits die Bereitstellung einer Standardschnittstelle für die Wiederverwendbarkeit.

Common Services bieten nicht-bildungsspezifische Funktionen, die jedoch Grundlage für die Ausführung der Application Services und der User Agents sind und ursprünglich gemeinsame Funktionen sowie Daten anderer Services beinhalten.

Infrastructure beschreibt die zugrunde liegenden Ressourcen, die für eine Implementierung vorausgesetzt werden. Diese werden jedoch im Rahmen der Konzeption des ELF nicht weiter definiert.

Da das E-Learning-Framework für viele Dienste auch Funktionsschnittstellen und Datenrepräsentation spezifiziert und auch Implementierungen und Werkzeuge bereit hält, hat es – im Gegensatz zur LTSA – nach obiger Definition viele Qualitäten einer Referenzarchitektur.

IMS Abstract Framework (IAF) Das IAF stellt für das IMS Global Learning Consortium einen definierten Kontext dar, innerhalb dessen es Lerntechnologien spezifizieren kann. Es enthält neben einer technischen Beschreibung einer serviceorientierten *Multi-Tier-Architektur*[52] eine abstrakte Darstellung von den Diensten[53], mit denen Lernsysteme im weitesten Sinne konstruiert werden können (vgl. [A+03]). Genau wie das ELF legt auch das IAF den Fokus auf ein gemeinsames Verständnis von architektonischen Sachverhalten auf Basis eines gemeinsamen Vokabulars, insbesondere auf Infrastrukturen für *verteilte* Lernsysteme (vgl. ebd.). Insoweit deckt das Framework die mögliche Bandbreite an Architekturen verteilter Lernsysteme ab, die mit dem Satz an spezifizierten Diensten entwickelt werden können.

Ein abstraktes Modell von Verhalten und Daten wird durch die Verweise auf bestimmte IMS Spezifikationen zu Diensten und Datenstrukturen aufgebaut, die zur Detaillierung des Frameworks genutzt werden. Des Weiteren werden Lösungsvorschläge zu Implementierungsaspekten gemacht, z. B. zu Transportmechanismen zum Austausch von XML-Dokumenten.

Genau wie beim ELF muss zur Ableitung einer Referenzarchitektur eine Auswahl von spezifizierten Diensten auf eine Anwendungsklasse bezogen adaptiert werden (*domain profiling*). Anderson et al. zählen dazu in [A+03] (1) die Identifikation anforderungsgerechter Spezifikationen, (2) die Verfeinerung der Spezifikation durch anwendungsbedingte Einschränkungen des Datenmodells und anwendungsbezogene Erweiterungen der Spezifikation und (3) die Validierung und Zertifizierung der abgeleiteten Referenzarchitektur gegenüber den originalen Spezifikationen.

Open Knowledge Initiative (OKI) Die *Open Knowledge Initiative* (OKI) ist ein Projekt zur Spezifikation eines Referenzmodells für universitäre Lernsysteme im amerikanischen Raum, das ursprünglich vom Massachusetts Institute of Technology (MIT) initiiert wurde[54]. Das Ziel ist die Definition von offenen Schnittstellen (*Open Source Interface Definitions*, OSIDs) zu fundamentalen Diensten der universitären Informationsarchitektur, durch die sowohl Lerntechnologien untereinander als auch mit universitären Administrationssystemen und anderen akademischen Systemen (z. B. Bibliothekssystemen und Digitalen Repositories) kommunizieren können. Eine OSID gilt dabei als zweiseitige Vereinbarung zwischen Dienstanbietern (Server) und Dienstnutzern (Clients). Implementieren beide Parteien diese Schnittstellenspezifikation, kann eine Interoperabilität bzw. eine Portabilität von Anwendungsfunktionen gewährleistet werden.

[52] Strukturierungsprinzip für Softwarearchitekturen, wobei einzelne Aspekte des Systems konzeptionell in einer Schicht (engl. *tier*) zugeordnet werden.

[53] In Analogie zum ELF sind die Dienste auf die Schichten *Application Layer*, *Application Services Layer*, *Common Services Layer* und *Infrastructure Layer* verteilt.

[54] Die Spezifikation werden von Herstellern wie beispielsweise Apple Computer, Cisco Systems, Microsoft, Moodle, IMS Global Learning Consortium und Universitäten wie das MIT, die California State University und die Open University of Catalonia adaptiert. Mehr Informationen unter `http://www.okiproject.org`. Letzter Zugriff: 31.07.2008.

Setzt sich die Referenzarchitektur auf dem amerikanischen Ausbildungsmarkt durch[55], können neue Komponenten sehr leicht in die dortigen universitären Informationsarchitekturen eingebunden werden, wobei ein initialer Installations- und Konfigurationsaufwand nicht notwendigerweise reduziert wird; die Kosteneinsparungen sollen sich jedoch im Betrieb durch verringerte Wartungs- und Weiterentwicklungskosten ergeben (vgl. [BCG03]).

Campus Source Engine (CSE) Die deutsche Open Source-Initiative CampusSource[56] entwirft auf Basis einer eigenen Referenzarchitektur (vgl. [SVS03] und [GMS03]) seit 2003 eine Softwaredesign namens *CampusSource Engine* (CSE) zur Koppelung ihrer Plattform- und Werkzeugbörse an die universitäre Diensteinfrastruktur. Die Referenzarchitektur besteht auch in diesem Beispiel aus der Spezifikation einer mehrschichtigen[57], dienstorientierten Architektur, sowie aus einer abstrakten Beschreibung von Diensten[58]. Die CSE wurde unter zwei Zielsetzungen entworfen (vgl. [S$^+$04]). Zum einen zur Definition und Implementierung einer einheitlichen Schnittstelle der bei CampusSource verfügbaren E-Learning-Plattformen zu Datenbeständen von Verwaltungssoftware der Firma Hochschul-Informations-Systeme GmbH (HIS)[59]. Zum anderen, um eine zielgerichtete Konvergenz der verfügbaren Systeme und Werkzeuge in CampusSource zu erlangen, damit für Anwender die Möglichkeit besteht, aus einer Vielzahl von Funktionen eine auf ihre individuellen Bedürfnisse abgestimmte Plattformen zusammenzustellen (vgl. ebd.).

Zum Zeitpunkt der Fertigstellung dieser Arbeit existiert bereits ein Teil der CSE als ereignisgesteuerte, Nachrichten-orientierte Integrationsplattform, die Schnittstellen zum Datenabgleich zwischen HIS LSF[60] und den CampusSource-Plattformen[61] implementiert.

[55] Stand August 2006 gab es bereits 31 unterschiedliche Anbieter und Implementierungen, die diese Spezifikationen berücksichtigen und somit interoperabel sind (vgl. [OKI06]).

[56] CampusSource wurde 1999 zur Verstetigung quelloffener Lernplattformen gegründet. Ihr Portfolio enthält derzeit 19 E-Learning-Anwendungen, sowohl umfangreiche Lernmanagementsysteme als auch kleinere Werkzeuge (siehe `http://www.campussource.de`).

[57] Six et al. unterscheiden hier Web-Schicht, Applikations-Schicht, lokale Datenhaltungsschicht und zentrale Basisdienste (vgl. [SVS03]).

[58] Es werden *Allgemeine Dienste* (Security Service, Directory Service und UI Service) und *spezielle Dienste* (Course Service, Exercise Service, Exam Service, StudyRecord Service) unterschieden. Die Aufzählung wird aber ausdrücklich als beispielhaft und nicht vollständig deklariert (vgl. [GMS03], S. 11f.).

[59] Die Software der HIS ist im Bereich der Verwaltungen der Hochschulen in NRW als Standard etabliert.

[60] Das Modul *Lehre, Studium, Forschung* (LSF) dient als Studieninformations-, Studienberatungs- und Planungssystem für verschiedene Nutzerkreise (Studierende, Lehrpersonal, Administratoren, Raumplaner).

[61] Beim Stand der Fertigstellung dieser Thesis sind dies ILIAS, CLIX, EWS II, metacoon, Moodle, OpenUSS und Stud.IP (vgl. [Pos07]).

2.2.5. Zusammenfassung

Das computerunterstützte Lernen hat durch die ständig zunehmende Vernetzung und Nutzung von Informations- und Kommunikationssystemen neue Gestaltungsmöglichkeiten hinzugewonnen. Stellte Ende der 1990er Jahre noch die durch WBTs verbesserte Distribution von Lerninhalten und die explorative Erfahrungen mit Hypermedia das Maß der Dinge dar, sind in den letzten Jahren neue E-Learning-Paradigmen hinzugekommen, die bspw. soziale Aspekte bei der Kooperation, eine bessere Integration von Lern- in Arbeitsprozesse, die Selbstorganisation beim Wissensmanagement oder den direkten Austausch mit Experten in einen neuen Fokus rücken.

Die bisher getrennten Präsenz- und Fernunterrichtsformen konvergieren durch Blended Learning Ansätze zu hybriden Unterrichtsmodellen, ebenso verschmelzen Theorien von Bildung und Wissen zu neuen pädagogischen Anwendungsfeldern von Wissensmanagement (vgl. [Rei05b]). Diese Aufweichung von Grenzen bisher getrennter Konzepte schlägt sich auch in der Nutzung von Technologien für das Lernen nieder; zu den bislang spezialisierten Lerntechnologien (vgl. Tabelle 2.2) werden zusätzlich Basistechnologien zur synchronen und asynchronen Kommunikation und Kollaboration verwendet. Dies mündet in neuen Anforderungen zur Integration von Lernumgebungen in heterogene Informationsarchitekturen, um Lern- und Arbeitsprozesse medienbruchfrei über System- und Technologiegrenzen hinweg zu unterstützen.

Dieser neue Fokuswechsel, von den funktionsmächtigen aber monolithischen Lernsystemen hin zu dienstorientierten, offenen IT-Infrastrukturen mit interoperablen Lernwerkzeugen lässt sich auch an den aktuellen Standardisierungsbestrebungen im E-Learning belegen: Wurden zunächst hauptsächlich Standards zur Inhalteverwaltung und -distribution erarbeitet, zielten die Bestrebungen der letzten Jahre schwerpunktmäßig darauf ab, Lerner- und Lernprozessdaten zu spezifizieren, um die Kombination von Lerntechnologien innerhalb eines Lernsystems zu erleichtern[62].

Besonders klar erkennbar lässt sich der Wechsel an den neueren Spezifikationen von Referenzarchitekturen für Lernumgebungen ablesen, die allesamt komponentenbasierte, dienstorientierte Multi-Tier-Architekturen beschreiben. Erste Implementierungen von OSIDS oder die CampusSource Engine zeigen den erfolgreichen Einsatz speziell im universitären Bereich, da somit der Übergang von einer verteilten kooperativen IT-Versorgung hin zu einer integrierten Informationsversorgung gefördert wird, ohne die an Hochschulen vorhandene Vielfalt an didaktischen Arrangements einzuschränken.

[62] Zwar zielen die Standards immer noch auf formale, institutionalisierte Lern- und Bildungsprozesse ab, werden aber nach Klebl nicht an Bedeutung verlieren, da auch das informelle Lernen auf Kompetenzen und ein bestimmtes Maß an Grundkenntnissen in einem Wissensbereich beruht, die in formalen Bildungsprozessen erworben wurden (vgl. [WK07]).

2.3. Neue technologische und inhaltliche Trends im World Wide Web

Neben aktuellen Trends im E-Learning hat die Evolution des World Wide Webs einen großen Einfluss auf die computergestützte Hochschullehre, sowohl aus technologischer als auch aus didaktischer Sicht. Im Folgenden sollen diese Weiterentwicklung und ihre Potenziale näher beschrieben werden.

2.3.1. Der Begriff des Web 2.0

Das Zerplatzen der Dot-Com-Blase im Herbst 2001 bedeutete nicht das Ende des World Wide Web, sondern einen Wendepunkt. Üblicherweise markiert eine solche Marktbereinigung eine Phase, in der sich neue Technologien durchgesetzt haben und nun aus der ökonomischen Perspektive betrachtet werden (*Slope of enlightenment*, vgl. [FL05]). Drei Jahre später wurde der Begriff „Web 2.0“ allgemein eingeführt, um die Konzepte und Anwendungen zu beschreiben, welche das Web in der Zeit nach diesem Wendepunkt wichtiger machten als zuvor[63]. Danach beschreibt der Begriff keine technologische Innovation oder Revolution webbasierter Anwendungen, sondern vielmehr einen neuen Evolutionsabschnitt des Internets. Obwohl dieser Abschnitt durch technische Neuerungen fixiert werden kann, liegt die Wurzel der Veränderung in einer neuen Verhaltensstruktur der Internetbenutzer, die sich über die Jahre der Onlinepräsenz auf einer gesellschaftlichen Entwicklungsebene gebildet hat.

Nach Tim O'Reilly nutzen Angebote des „neuen“ World Wide Webs daher die „kollektive Intelligenz“ ihrer Nutzer (vgl. [O'R05]): Suchresultate können durch Berücksichtigung von Benutzeraktivitäten verbessert werden, kollaborative Spam-Filter lernen aus den Einstufungen ihrer gesamten Benutzer und funktionieren somit besser als solche, welche nur die Nachricht an sich analysieren. Das gemeinsame Verschlagworten („Taggen“) von Inhalten mit teils überlappenden Assoziationen führt zu Möglichkeiten, diese entlang logischer Wege wieder aufzufinden.

So genannte *Social Software* unterstützt im Web zum einen das Publizieren von eigenen Inhalten, zum anderen unterstützt sie den Aufbau vernetzter sozialer Strukturen durch Nutzerbeteiligung. Auf den folgenden Seiten werden diese neuen Werkzeuge und ihre Nutzungsmöglichkeiten vorgestellt, sowie die Implikationen, die sie für das computergestützte Lernen haben.

[63] Zwar wurde der Begriff Web 2.0 bereits 1999 in einem Aufsatz von Darcy DiNucci verwendet, in dem er aktuelle technologische Entwicklungen des Internets beschreibt (vgl. [DiN99]), wirklich geprägt wurde er mit seiner heutigen Bedeutung jedoch erst im Jahre 2005 von Tim O'Reilly mit seinem Artikel „What is Web 2.0“ (vgl. [O'R05]).

2.3.2. Benutzergenerierte Inhalte und soziale Strukturen

2.3.2.1. User Generated Content – Vom Konsumenten zum Produzenten

Weblogs Weblogs (oder kurz: Blogs) sind die fundamentalen Vertreter der sozialen Software, die aktuell das Erscheinungsbild des Internets revolutionieren und einen wichtigen Eckpfeiler des heutigen World Wide Webs ausmachen. Unter dem Begriff Weblog wird in seinen verschiedensten Formen nach eine Ausprägung von Internet-Publikation verstanden, deren inhaltliche Bandbreite von persönlichen Gedanken und privaten Dingen des Lebens bis hin zu journalistischen Beiträgen professioneller Nachrichtendienste reicht. Grundsätzlich steht der Begriff Weblog für die drei folgenden zentralen Ideen:

- Die Erzeugung und das vereinfachte Einstellen von Inhalten mit Hilfe eines Content-Management-Systems (CMS)
- Eine zeitliche Einordnung durch eine chronologische Darstellung, sowie die Zusammenfassung ähnlicher Inhalte durch Kategorisierung
- Einbezug der Leser durch Kommentar- und Verknüpfungsfunktionen direkt am Beitrag selbst. Durch Automatismen kann auf Einträge anderer Weblogs, die sich auf diesen Beitrag beziehen, am Beitrag selbst verwiesen werden.

Heute existieren viele Blog-Hosting-Anbieter, über die sich kostenlose Weblogs in wenigen Schritten und ohne jegliche Kenntnisse in Web-Design oder -Programmierung einrichten lassen. Professionelle, einfach zu betreibende Weblog-Systeme sind auch durch die Open Source-Community entstanden. Auf die freie Verfügbarkeit dieser Systeme und ihrer leichten Bedienung begründet fanden Weblogs im World Wide Web eine starke Verbreitung (vgl. Abb. 2.6). Durch zahlreiche thematische und soziale Verknüpfungsmöglichkeiten[64] zwischen einzelnen Weblogs untereinander ist mittlerweile ein stark interaktives, soziales Netzwerk entstanden. Seit Ende der 1990er wächst diese so genannte Blogosphäre[65] exponentiell an.

Wikis Ebenfalls zur sozialen Software gehören Wikis, die gegenüber klassischen Technologien des Informationsmanagements ein Potenzial für soziale Prozesse bereitstellen. Das erste dokumentierte Wiki wurde im Frühjahr 1995 durch Ward Cunningham, einem Software-Designer aus Portland, betrieben, um zusammen mit Entwicklern aus aller Welt eine Datenbank für Entwurfsmuster aufzubauen[66].
In Analogie zu Weblogs basieren Wikis ebenfalls auf einem spezialisierten CMS. Der Fokus liegt hierbei auf der kooperativen Erstellung von Hypertexten, wobei keinerlei Kenntnisse in Web-Design oder -Programmierung benötigt werden. Mit Hilfe einer einfachen Syntax kann jeder Benutzer – meistens ohne vorherige Anmeldung am System

[64] Verknüpfungen zwischen Weblogs können über die Trackeback/Ping-Funktion (s. S. 53) geschaffen werden, als auch über eine Auflistung der vom Autor eines Weblogs selbst gelesenen Weblogs, der so genannten Blogroll.
[65] Die Blogosphäre kennzeichnet die Gesamtheit der Weblogs. Es ist ein Kunstwort aus der Abkürzung Blog und Sphäre (Gesellschafts- bzw. Wirkungskreis).
[66] Das Wiki ist unter `http://c2.com/cgi/wiki` online. Letzter Zugriff: 31.07.2008.

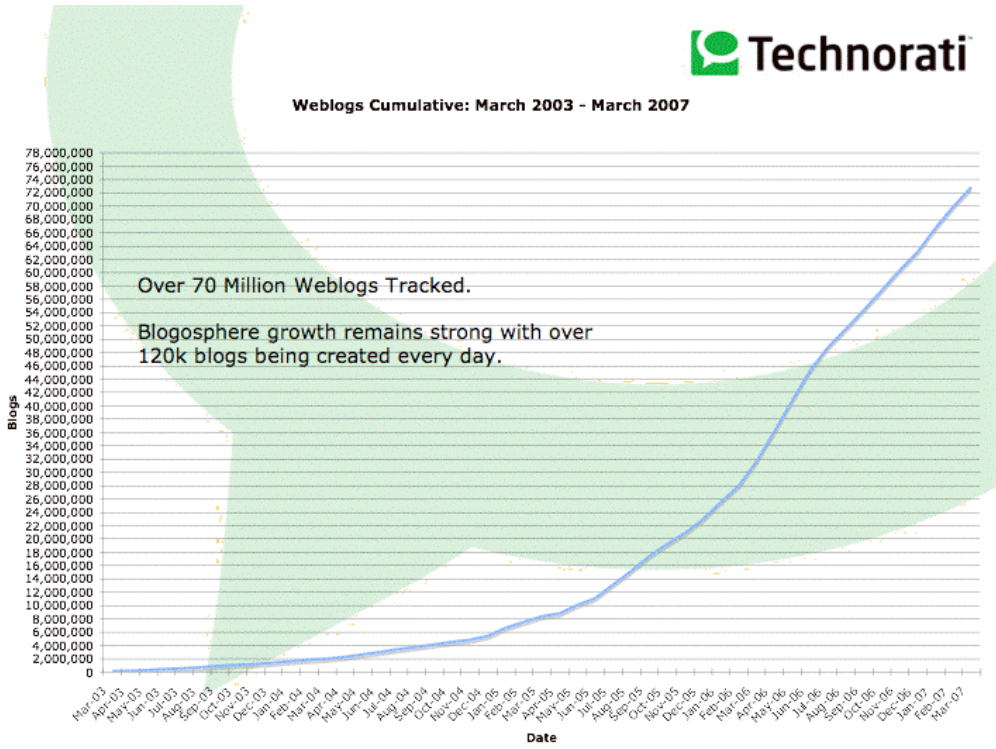

Abbildung 2.6.: Stetiges Wachstum der in der Blogosphäre (Quelle: `http://technorati.com/weblog/`)

– einen vorhandenen Text editieren, verbessern, ergänzen und mit weiteren Seiten verknüpfen. Da ein Wiki für jeden Text eine Historie speichert, in der alle Änderungen seit der ersten Version aufgezeichnet werden, können Inhalte durch diesen offenen Ansatz nicht dauerhaft beschädigt werden.

Durch eine kooperative Verschlagwortung von Artikeln automatisch ein Kategoriensystem aufgebaut, das ein kontrolliertes, strukturiertes Vokabular für die Erschließung und die Verschlagwortung weiterer Inhalte zur Verfügung stellt.

Das wichtigste Gestaltungselement sind Hypertext-Verknüpfungen zur referenziellen Vernetzung der Lemmata. Externe Verweise führen dabei zumeist zu Quellen, die entweder weiterführende Informationen bereitstellen oder die im Artikel gemachten Aussagen belegen. Der Verweis auf einen nicht existierenden Wiki-Inhalt birgt keinen Fehler, sondern eine Möglichkeit, den fehlenden Inhalt sofort zu erfassen.

Eine Übersicht über die Bandbreite an Nutzungsmöglichkeiten liefert das Wikiverzeichnis[67].

[67] Siehe `http://www.wikiservice.at/gruender/wiki.cgi?WikiVerzeichnis`. Letzter Zugriff: 31.07.2008.

Podcasts Als *Podcasting* wird im Allgemeinen das Produzieren, Anbieten und Übertragen von Audio- und Videodateien bezeichnet[68]. Podcasts sind Sendungen bzw. Serien von Sendungen und können ähnlich wie News oder Blogbeiträge per *RSS-Feed* abonniert werden (siehe Abschnitt 2.3.3.1). Dies ermöglicht es, neue Folgen automatisch aus dem Web zu laden. Somit steckt keine neue Technologie hinter diesem Begriff, sondern wiederum eine neues Nutzungskonzept bereits vorhandener Technologien. Da die Produktion von Audio- bzw. Videodateien auf aktuellen Multimedia-Computern für viele Anwender oftmals schon mit darauf vorinstallierten Software einfach möglich ist und eine einfache Veröffentlichung der Dateien als Informationsangebot über Plattformen wie *podcast.de*, *podster.de* oder Apples *iTunes* kostenlos erfolgen kann, existieren neben professionellen Angeboten von bspw. WDR und BBC auch eine Vielzahl individueller Audio-Tagebücher und Videoserien mit thematischen Schwerpunkten[69]. Ohne großen Aufwand kann sich jedermann mit Netzzugang aus dem weltweiten Angebot an Podcasts ein individuelles, auf die persönlichen Interessen abgestimmtes Audio- oder Videoprogramm abonnieren, das sich bei neuen Folgen automatisch aktualisiert.

Gemeinschaftliches Indexieren Eine besondere Form der semantischen Verknüpfung von Inhalten ist das *Social Tagging*[70]. Diese personengebundene Verschlagwortung von Inhalten durch *Tags* (Deskriptoren) stellt im Gegensatz zur formalen Semantik der Idee des klassischen *Semantic Webs*[71] eine bewusst unreglementierte Generierung von Metadaten dar. Nach kurzer Zeit bildet sich über die Gesamtheit aller Nutzer somit meist ein Schlagwortsystem mit einem Kernbestand an Begriffen, der für eine gezielte Suche nach Inhalten brauchbar ist.

Die durch gemeinschaftliches Indexieren nach dem „bottom-up"-Prinzip erstellte Sammlung von Tags wird neusprachlich *Folksonomy* genannt[72]. Als lebendiges, offenes und kollaboratives Kategoriensystem hat eine Folksonomy auch soziale Aspekte: Über die Personengebundenheit ihrer Schlagwörter können Benutzer Nutzer mit ähnlichen Interessenslagen und persönlichen Strukturen ausfindig machen, und darüber wiederum an weitere für sie interessante Inhalte gelangen.

Die Schlagwörter einer Folksonomy werden oft in so genannten *Tag Clouds* zusammengefasst, in denen die Begriffe visualisiert werden. Die Schriftgröße der einzelnen Schlagwörter variiert proportional zu ihrer Häufigkeit (vgl. [Möl05], S. 155).

Die öffentliche Verschlagwortung persönlicher Lesezeichen im Web stellt die derzeit am meisten genutzte Form des Social Taggings dar, das so genannte *Social Bookmarking*. Spezialisierte Dienste wie *del.icio.us* oder *furl.net* unterstützen diese Speicherung und Auszeichnung von Hyperlinks und ermöglichen darüber eine Suche nach Inhalten im Netz, deren Relevanz durch das Kollektiv im Allgemeinen und das Individuum im Besonderen bestimmt wird (vgl. [Alb07], S. 91f.).

[68] Abgeleitet von den Begriffen *Broadcasting* und Apples *iPod* wird Podcasting häufig jedoch nur auf die Verbreitung von Audiodateien bezogen.

[69] Bspw. Nachrichten und Politik, Musik, Wirtschaft, Humor, Sport oder Technik.

[70] Auch *Collaborative Tagging* oder schlicht „gemeinschaftliches Indexieren" genannt.

[71] Auszeichnung von Inhalten durch maschinenlesbare semantische Informationen, die durch Taxonomien und Ontologien formal strukturiert sind.

[72] Ein Neologismus aus den englischen Begriffen *folk* und *taxonomy*, der zurückzuführen ist auf Thomas Vander Wal.

Soziale Netzwerke und Online Communities Die Interaktion auf und mit sozialen Netzstrukturen ist eine fundamentale Ausprägung des Web 2.0. Nach O´Murchu et al. haben soziale Netzwerke als Vorstufe virtueller Communities die Aufgabe, Menschen zu verbinden und Informationen über diese in den Nutzerprofilen zu geben: „A social networking site (SNS) connects and presents people based on information gathered about them, as stored in their user profiles“ [OBD04], S. 2. Im Web erstellen die Nutzer eines sozialen Netzwerkes ein ausführliches Profil, in dem sie ihre Interessen und Schwerpunkte sowie Möglichkeiten zur Kontaktaufnahme preisgeben. Im Sinne des *Social Networkings* spiegelt dieses Profil – im Gegensatz zum allgemein üblichen Avatar in Communities – die Persönlichkeit des Nutzers so exakt wie möglich wider. Anonymität ist beim Aufbau sozialer Strukturen nicht erwünscht. So können Teilnehmer untereinander durch geeignete Funktionen der technischen Plattform ihren Kontakt zu anderen Teilnehmern oder auch Gruppen bestätigen und Kontakte anderer verfolgen. Es lassen sich somit zielgerichtet sowohl geschäftliche Beziehungen als auch Freundschaften pflegen, das Auffinden von Menschen mit gemeinsamen Interessen wird erleichtert.

Virtuelle soziale Netzwerke und Online-Gemeinschaften funktionieren nach Manuel Castells aufgrund zweier Merkmale: Erstens die freie horizontale Kommunikation (n:m-Kommunikation) und zweitens die „selbstgesteuerte Vernetzung“ der eigenen Person, um ein Netzwerk im Netz zu bilden (vgl. [Cas05], S. 66). Es entsteht somit eine *Selbst-Veröffentlichung*, *Selbst-Organisation* und *Selbst-Vernetzung* im Internet (vgl. ebd.).

2.3.3. Technologische Trends

2.3.3.1. Neue Mechanismen zur semantischen Verknüpfung von Inhalten

Die Blogosphäre bildet sich autonom als hoch vernetzte Struktur um Micro-Contents, erstellt von unzähligen, unabhängig voneinander agierenden Autoren. Ein „Innen“ und „Außen“ gibt es nicht mehr: Externe Inhalte von „Außen“ werden aggregiert und innerhalb einer persönlichen Arbeitsumgebung verfügbar gemacht. Vom Benutzer selbst erstellte Inhalte sind sofort für alle verfügbar. Die Netzstrukturen des Webs ermöglicht somit ein kooperatives Publizieren und eine semantische Verflechtung. Aus technischer Sicht werden hierzu bestimmte standardisierte technische Funktionen – Trackbacks, Pingbacks und Permalinks:

Trackbacks Als *Trackback* bezeichnet man eine Funktion, mit der sich Weblogs gegenseitig über Reaktionen und Kommentare auf Einträge untereinander austauschen können. Hierdurch entstehen unter anderem die wichtigen und typischen sozialen und semantischen Verlinkungen und Interaktionen der Blogosphäre: Blogger können sich in ihren eigenen Weblogs direkt auf Beiträge anderer beziehen, was als Reaktion direkt am Ausgangsbeitrag sichtbar wird. Somit ist die direkte Responsivität von Trackbacks ein charakteristisches Element vieler Blogs.

Pingbacks Pingbacks funktionieren analog zu Trackbacks. Mit dem Unterschied, dass alle in einem Beitrag aufgeführten Links automatisch angepingt, also benachrichtigt werden. Hinter den Ping-Links müssen Pingback-fähige Server stehen, um das

Signal empfangen und verarbeiten können, um ihrerseits eine Verlinkung vornehmen zu können. Üblich ist auch der Einsatz von Pingbacks an Weblog-Portalen, um die Beiträge zu verbreiten oder zu vernetzen.

Permalinks Lange Zeit waren Weblogs wie Webseiten meist nur als solche referenzierbar, so dass verlinkte Beiträge immer tiefer auf eine Seite nach unten wanderten oder durch nachfolgende ersetzt wurden. Permalinks bezeichnen Verweise auf eine Webseite oder einen Artikel, die für einen möglichst langen Zeitraum gültig bleiben. Diese Links werden meist von Weblogsystemen nach einem bestimmten Muster automatisch erstellt und den neuen Beiträgen zugewiesen.

Verbreitung und Verdichtung von Inhalten Durch die ständig zunehmende Zahl autonomer Inhalte-Anbieter im Internet und die damit einhergehende Dezentralisierung von Informationen bedarf es alltagstauglicher Technologien, um Inhalte menschen- und maschinenlesbar auszuzeichnen und um sie serverseitig auf Portalen oder clientseitig in sog. Newsreadern individuell zusammenführen zu können. Die Lösung ist ein standardisiertes, XML-basiertes Datenformat RSS (Real Simple Syndication), das Anbieter zusätzlich zum Inhalt vorhalten können: „RSS (which stands for RDF Site Summary, but often refered to as Really Simple Syndication) is a computer-generated data-file format that sites use to communicate their contents to other sites and applications. Using a really simple definition structure (thus the name,) RSS makes it easy for developers to extract and integrate data from other sources into their own“ [Spo07].

RSS oder auch Atom-Feeds sind XML-basierte Metadatenformate, die sich besonders gut zur Syndizierung[73] von Micro Content (also kleinsten Informationseinheiten) eignen. Beide werden serverseitig in Form eines Daten-Feeds veröffentlicht, um clientseitig überwacht zu werden.
Durch ihr maschinenlesbares Format können RSS-Daten automatisch weiterverarbeitet, gefiltert, sortiert und in andere Umgebungen integriert werden. Beispielsweise bieten Suchdienste darauf basierend neu zusammengesetzte RSS-Feeds zu einem bestimmten Stichwort an, die automatisch bei neuen oder geänderten Beiträgen aktualisiert werden, und somit bedeutend aktueller sind als klassische Suchdienste, die mit Hilfe von Webcrawlern Webseiten zunächst durchsuchen und analysieren müssen.

Am Häufigsten findet die Syndikation ihre Anwendung aber in der *Aggregation*, also in der Sammlung von Feeds in einer dafür spezialisierten Anwendung[74]. Somit kann eine große Anzahl von sich ändernden Webseiten zentral und automatisiert verfolgt werden.

2.3.3.2. Mash-Ups mit offenen APIs und Web-Services

Ein Kennzeichen des Web 2.0 ist die Verbreitung von Teilfunktionalität komplexerer (Web-)Anwendungen oder auch einzelnen Diensten als leichtgewichtige Web-Applikation auf Basis von Standards wie Web-Services über SOAP (vgl. Abschnitt 4.1.1.3). *Software as a Service* (kurz: SaaS) heißt dieses neue Konzept, bei dem es sich – der Name sagt

[73] Syndikation bezeichnet die Überlassung lizensierter Inhalte zur Weiterverwendung.
[74] *Aggregatoren* oder *News-* bzw. *Feedreader* genannt.

es schon – also nicht um eine neue Software-Kategorie geht, sondern vielmehr um die Bereitstellung bekannter Funktionalität.

In Unternehmenskontexten liegt bei der Nutzung von SaaS der Hauptvorteil darin, auf sich schnell ändernde Geschäftsprozesse entsprechend flexibel reagieren und benötigte neue Funktionen schnell zur Verfügung zu haben. Aber nicht nur professionelle Nutzer und Firmenkunden können von diesen Diensten profitieren: Durch die Einbindung von Online-Werbung und Affilitate-Programmen auf der eigenen Webseite kann auch der „einfache Internetanwender" die Angebote von Google, Amazon, ebay und Co. als Verdienstmöglichkeit nutzen. Implementierungen dieser Schnittstellen werden von den Dienste-Anbietern meistens oft auch in mehreren Programmiersprachen bereitgestellt, damit der Kunde die Funktion in eigenen Anwendungen nutzen kann[75].

Von einem *Mashup*[76] spricht man, wenn zwei oder auch oft mehrere Datenquellen externer Anbieter über offene Schnittstellen auf einer Webseite zusammengebracht werden. Andere Methoden der Einbindung sind RSS-Feeds oder Javascript. Der Anwendungsbereich bei Mashups liegt also in der Entwicklung neuer Dienste durch die Zusammenführung und nahtlose (Re-)Kombination bereits bestehender Inhalte anderer Anbieter über offene Schnittstellen (vgl. [Alb07], S. 133)[77]. Das Prinzip der Verdichtung von Informationen durch Kontextualisierung findet auch hier Anwendung, und so werden Mashups zu einem der wichtigsten Merkmale des Web 2.0.

2.3.3.3. Rich Client Applications

Im Laufe der letzten zehn Jahre ist die Anzahl der Techniken, die das Web dynamischer machen sollen, stetig gestiegen. Anders als dynamisches HTML (DHTML) und Flash, welches die Darstellung von Webseiten beleben soll, setzt AJAX (kurz für: *Asynchronous Javascript And XML*) bei der Kommunikation zwischen Browser und Webserver an und tauscht zwischen ihnen asynchron XML-Nachrichten aus. AJAX kann immer an den Stellen eingesetzt werden, wenn die Verarbeitungslogik von Benutzereingaben nicht auf zu große Datenmengen oder zu komplexe Funktionen zugreifen muss. Effektiv ist es also vor allem für komfortable Web-Anwendungen mit einem hohen Interaktionsgrad. Der Browser schickt die eingegebenen Daten im Hintergrund an den Server, ohne Zutun des Benutzers. Im Gegensatz zum klassischen Ansatz, der erst auf ein „Submit" des Formulars die Daten an den Server sendet. Dieser verarbeitet die Daten und schickt eine Nachricht, die das Ergebnis der aufwändigen Prüfung enthält, an den Browser zurück.

Die für dieses Vorgehen angewendeten Technologien gibt es seit einigen Jahren[78]. Neu

[75] Auf das Konzept der Funktionsintegration wird im Abschnitt 4.1.1.3 noch einmal detaillierter eingegangen.

[76] Der Begriff *Mashup* kommt ursprünglich aus der Musik und bezieht sich auf die Mischung der Musik eines Songs mit dem Gesang eines anderen Songs, um daraus einen neuen Song zu produzieren (vgl. [Alb07], S. 132).

[77] Sofern die Datenströme zugänglich, verwertbar und lizenzrechtlich verwendbar sind.

[78] Im Einzelnen handelt es sich um (vgl. [GGA06], S. 5): (1) standardgerechte Präsentation mit XHTML und CSS, (2) dynamische Anzeigen und Interaktivität mittels des Document Object Model, kurz DOM, (3) Datenaustausch und -manipulation mit XML und XSLT, (4) asynchrone Datenabfrage unter der Verwendung des XMLHttpRequest-Objektes oder XMLHTTP sowie dem Bindeglied Javascript.

ist allenfalls das breite Verständnis bei Entwicklern, dass in komfortablen Applikationen die Kommunikation zwischen Browser und Server nicht zwangsläufig auf benutzerausgelöste Ereignisse beschränkt ist. Durch diesen Paradigmenwechsel gleicht sich die Bedienbarkeit von Web-Anwendungen immer mehr dem Verhalten „nativer“ Applikationen an[79]. Weiterhin trägt die Verbreitung von AJAX zum Entstehen besserer Bibliotheken sowohl auf Server- als auch auf Client-Seite bei, die ihrerseits die Entwicklung komfortabler Anwendungen einfacher machen.

2.3.4. Die Auswirkungen des Web 2.0 auf das E-Learning

Das Web 2.0 stellt einen grundsätzlichen Wandel in der Wahrnehmung und in der Nutzung des Internets dar, der zum einen für die Planung und Konzeption des didaktischen Designs von E-Learning bedeutsam ist, zum anderen auch Veränderungen an der technischen Konzeption und am Aufbau von virtuellen Lernumgebungen impliziert. Die Aufweichung der Grenzen zwischen Autoren und Konsumenten, zwischen der Zentralisierung und Dezentralisierung von Inhalten und Diensten sowie die Öffnung von Privatsphären bedeuten analoge Veränderungen für die Grenzen des klassischen E-Learnings (vgl. [Ker07]):

Lerner werden zu Mitautoren von Lernangeboten. In vielen virtuellen Lernumgebungen wurde bei der Umsetzung die Wissensbildung nur als Rezeptionsprozess begriffen, als Folge von Faktenlernen und Routinen (vgl. [Man04], S. 19). Die klassische Rolleneinteilung erlaubt es nur dem Autor/Tutor bzw. didaktischen Designer, Lernmaterialien aufzubereiten und einzustellen[80]. Mit neuen Werkzeugen wie kooperativen Wissensräumen, Wikis, Annotationen an und in Informationsobjekten[81] und *Social Bookmarks* können Lernende selbst die Lerninhalte aktiv mit gestalten und ihre Lernwege durch eigene semantische Verknüpfungen strukturieren, diese mit anderen Kommilitonen austauschen und diskutieren. Dadurch kann ein neuartiger komplexer, vernetzter Umgang mit Informationen und darauf aufbauendem Wissen ebenso erlernt werden, wie die kritische, wertende Betrachtung von Informationen und die Akzeptanz unterschiedlicher Sichtweisen auf Themen und verschiedene Weltanschauungen (vgl. [Erp06], S. 20ff.).

Lernen wird ubiquitär, Lernumgebungen mobil. Notebook-Universitäten[82] bieten Infrastrukturen, in denen mobile Endgeräte als Wissenswerkzeuge eine „wirkliche“ Vernetzung der Lernorte auf dem Campus (*on campus*) und außerhalb des

[79] „Basically, what AJAX means is “Javascript now works”. And that in turn means that web-based applications can now be made to work much more like desktop ones“ Paul Graham, November 2005.

[80] Und dies geschah meist auch noch ohne Berücksichtigung individueller Lernerinteressen und statisch mit einer einseitigen Sichtweise (vgl. [Erp06], S. 20ff.).

[81] Annotationen können sich dabei nicht nur auf reinen Text beziehen, wie z. B. ein Kommentar in einem Weblog, sondern auch direkt auf Bild- oder Videoausschnitte wie bspw. bei dem Fotodienst `http://www.flickr.com`.

[82] Eine „[. . .] Organisationsform der Hochschule, in der der Einsatz mobiler Rechner sowie die verstärkte Nutzung moderner Kommunikationstechniken und -möglichkeiten sowohl auf Seite der Lehrenden als auch auf Seiten der Studierenden integrativer Bestandteil der alltäglichen Ausbildung ist. In Abgrenzung zum Begriff der *Virtuellen Universität* zielt die *Notebook-University* primär auf die mobile (oder ubiquitäre) Nutzung moderner Informations- und Kommunikationstechnologien in Präsenzhochschulen [. . .]“ ab (vgl. [Bau03a], S. 35ff.).

Campus (*off campus*) erlauben. Viele Lernaktivitäten sind daher nicht mehr an einen festen Lernort gekoppelt, sondern können bspw. im Seminarraum, in der Mensa/Cafeteria, im Praktikumsbetrieb, vom Wohnheim oder zu Hause aus und sogar unterwegs auf dem Weg zur Uni oder ins Wochenende via iPod, Handy oder Notebook stattfinden (vgl. [Ker04b]). Die durchgängige Verfügbarkeit digitaler Informationen an allen Orten, an denen Lehrende und Lernende mit Wissen arbeiten, ist dadurch möglich (*Ubiquität*).

Lernen wird zu einer öffentlichen Aktivität. In konstruktivistisch geprägten Lernansätzen steht nicht mehr der instruierende Lehrende im Vordergrund, sondern der aktiv Lernende. Dieser verharrt nicht mehr in einer rezeptiven Lernhaltung, sondern ist maßgeblich an der Steuerung des Lerngeschehens durch Aktivitäten beteiligt, in denen gleichzeitig Leistungen erbracht werden, die öffentlich sichtbar sind (vgl. hierzu [Ede00], S. 287). Soziale Software ermöglicht in diesem Kontext öffentlich nachvollziehbare Interaktionen ihrer Benutzer, wie die Veröffentlichung von Informationen, Informationsmanagement und Austausch von Inhalten auf kooperativen Wegen (bspw. über Diskussionsbeiträge, Portfolios, Beiträge in Weblogs oder Wikis; vgl. [Ric06], S. 8).

Im Folgenden soll kurz auf die Potenziale sozialer Software aus didaktischer Sicht eingegangen werden, bevor der Einfluss des Web 2.0 auf die technische Umsetzung virtueller Lernumgebungen skizziert wird.

2.3.4.1. Potenziale sozialer Software für das E-Learning

Das Web 2.0 bietet Potenziale für das Lernen und den Kompetenzerwerb. Schmidt nennt in [Sch06] dazu drei wichtige Aspekte:

1. Auf- und Ausbau eines persönlichen *Informationsmanagements*: Informationen und Wissen können produziert und rezipiert, gesammelt und semantisch neu strukturiert werden.

2. Auf- und Ausbau eines persönlichen *Beziehungsmanagements*: Zu anderen Wissensträgern können Kontakte über ein soziales Netz aufgebaut und nachhaltig gepflegt werden.

3. Auf- und Ausbau eines *Identitätsmanagements*. Durch das Publizieren von eigenen Inhalten und Ansichten und dem Auseinandersetzen mit dem kritischen Feedback der Leserschaft kann man mit der Zeit eine eigene digitale Identität/Reputation entwickeln. Dazu gehört auch die Reflexion seiner selbst.

Genau diese drei Begriffe stehen auch für „die Auseinandersetzung des Einzelnen mit der gegenständlichen sozialen Umwelt ebenso wie mit sich selbst, dem individuellen Wissen und Können und den eigenen Einstellungen und Werten“ ([Rei07], S. 11), was letztendlich die Voraussetzung dafür ist, Kompetenzen zu entwickeln, die zum Handeln und Problemlösen befähigen (vgl. ebd.).
Unter diesen Aspekten werden im Folgenden die Potenziale einzelner Werkzeuge und Dienste für das E-Learning beschrieben.

Weblogs Weblogs sind die derzeit im Web am weitesten verbreitete und angenommene Technologie, mit der aktives Schreiben und Lesen praktiziert wird. In einer Lernumgebung ergeben sich durch Weblogs Interaktions- und Kommunikationsmöglichkeiten zwischen Lernenden untereinander sowie zwischen Lehrenden und Lernenden (vgl. [Ric06], S. 8). Sie ermöglichen ein selbständiges, problemorientiertes, exploratives als auch aktives und kooperatives Lernen und bieten den Lehrenden eine transparente Möglichkeit der Betreuung. Für den didaktischen Einsatz sehen Beuschel und Draheim verschiedene Medienmerkmale eines Weblogs, die bei richtiger Anleitung zum Tragen kommen können (vgl. [BD05], S. 228ff.):

Explorationsmedium Die hochgradige Vernetztheit der Blogosphäre bietet für die Online-Recherche ein Potenzial für selbstorganisiertes und exploratives Lernen. Des Weiteren kann die kompetente Einschätzung der Glaubwürdigkeit einer Information als wichtiger Bestandteil von Medienkompetenz erachtet werden.

Reflexives Medium Informationen in der Blogosphäre sind direkt Kritiken, Verbesserungen und Erweiterungen ausgesetzt, welche die Inhalte verändern, aktualisieren oder erweitern können. Dadurch wird eine diskursive Art des Schreibens gefördert und ausgebildet; Studierende kommentieren die Beiträge anderer und können lernen Kritik anzunehmen (vgl. [Ora02])[83].

Soziales Medium Erfahrungen der Studierenden aus Diskussionen in Weblogs können das Verständnis für die soziale Konstruktion von Informationen unterstützen (vgl. [Ora03]).

Medium subjektiver Entwicklung Beim *Bloggen* kann die Fähigkeit zur Selbstorganisation und das Lernmanagement gestärkt und ein kompetente Umgang mit Texten und fachlichen Ausdrücken erlernt werden. Schreibblockaden können abgebaut werden (vgl. [BD05], S. 229).

In Lernumgebungen eingesetzt, können Weblogs verschiedenste Unterstützungsfunktionen übernehmen. Richardson nennt in [Ric06], S. 40ff., mögliche Einsatzgebiete, wie zum Beispiel als reflexives Journal für Lehrer zur Kommunikation, für Feedbacks, sowie zur Strukturierung und Reflexion von Lerneinheiten. Weiterhin schlägt er Weblogs für Klassenräume, beziehungsweise Kursgruppen oder Projektgruppen vor, welche für die Organisation des Unterrichtsablaufs, zur Gruppenkommunikation und Gruppenarbeit dienen können. Zusätzlich empfiehlt Richardson individuelle Weblogs für jeden Lernenden, für eigenständige Lernprozesse, Ablage und Arbeitsumgebung (vgl. [Ric06], S. 40ff.). Eine darüber hinausgehende Übersicht der Einsatzfelder von Weblogs in der Bildung stellt Leslie in [Les03] vor.

[83] In diesem Zusammenhang sieht Rösch (zitiert in [Kös05]) Potenziale von Weblogs für wissenschaftliches Arbeiten, nämlich 1) als Quelle für neue Ideen, 2) als Informationsfilter, 3) als Verbreitungsmedium und 4) als Testumgebung, um für Thesen und Theorien die Reaktionen der Scientific Community zu bekommen. Die dabei entstehende „echtere, spontanere und unmittelbarere Wissenschaftskommunikation“ (Baumgartner in [Kös05]) kann dabei Entwicklungsprozesse vorantreiben. Hemmend kann sich allerdings die Publikation unsicherer Ergebnisse in einem öffentlichen Raum auswirken: Wissenschaftler publizieren in der Regel erst dann, wenn sichere Ergebnisse vorliegen.

Wikis Für einen Einsatz in virtuellen Lernumgebungen bieten Wikis zwei vorrangige Funktionen. Zum einen können sie als Informationsquelle dienen, zum anderen als Orte kooperativer Wissensarbeit (vgl. [Mit06], S. 119).

Im Internet existiert eine Vielzahl virtueller Enzyklopädien, die bei der Recherche in einem Problemlöseprozess als wichtige Informationsquelle dienen können. Ihr Einsatz ist jedoch im Bildungsbereich umstritten, da Inhalte nicht zwangsläufig von renommierten Experten verfasst wurden und eine nachvollziehbare Quelle nicht grundsätzlich gegeben ist[84]. Unter dem Gesichtspunkt des kritischen und reflexiven Umgangs mit Informationen können Lernende jedoch aktiv an Wikis teilnehmen und ihren Beitrag zur Qualitätssicherung durch Verbesserungen leisten. Ab diesem Punkt nehmen sie selber an öffentlichen Kommunikationsprozessen teil und können wertvolle Erfahrungen durch Interaktion mit der dahinter stehenden Gemeinschaft erlangen. Sie sind ihrerseits Lob und Kritik ausgesetzt und erlangen in Analogie zu den Weblogs dadurch soziale Kompetenz.

Als Orte kooperativer Wissensarbeit kann ein Wiki eine Plattform zur Ideenfindung, für Gruppenarbeiten, Archivierung oder Präsentation von Ergebnissen darstellen. Dabei sind selbst Kooperationen mit anderen Kursgruppen oder Lernumgebungen möglich (vgl. [Mit06], S. 125).

Für den didaktischen Einsatz sehen Thelen und Gruber folgende Merkmale (vgl. [TG03], S. 359f.):

Förderung eines wissenschaftlichen Schreibstils Die aktive und kritische Auseinandersetzung mit bereits erstellten, wissenschaftlich gut formulierten Texten in einer Umgebung von kritisch korrigierenden Autoren kann den eigenen Schreibstil positiv beeinflussen.

Schaffung eines „Mikrokosmos“ für wissenschaftliche Prozesse Wissenschaftliches Arbeiten ist grundsätzlich ein kooperativer, sozialer Prozess. Wiki-Technologien können hierbei einen Ort für wissenschaftliches, kooperatives Arbeiten innerhalb einer Lernumgebung darstellen.

Revisionskontrolle von Arbeitsgruppen Das kooperative Verfassen wissenschaftlicher Arbeiten kann innerhalb eines Wikis strukturiert und dokumentiert ablaufen, da Veränderungen anhand von Revisionskontrollen des Wikis aufgezeichnet werden. Lehrende und Lernende können dadurch Gedankengänge sowie gestörte Arbeitsabläufe transparent nachvollziehen.

Kooperationen über Distanzen Wikis eignen sich für den Einsatz bei Seminar- oder Hausarbeiten, da kooperatives Arbeiten auch ortsunabhängig realisiert werden kann. Dabei verlieren Hausarbeiten ihren isolierenden Charakter und Hypothesen können schon frühzeitig in der Gemeinschaft diskutiert werden.

[84] Mitchell stellt beispielhaft die Frage, ob man Wikipedia vertrauen kann und ob Wikipedia als legitime Quelle innerhalb wissenschaftlicher Ausarbeitungen akzeptiert werden kann (vgl. [Mit06], S. 121). Thelen und Gruber merken hierzu kritisch an, dass durch das gemeinschaftliche Editieren in Wikis wichtige Metainformationen über die Autoren verloren gingen, die für ein kritisches Interpretieren der Aussagen vonnöten sind (vgl. [TG03], S. 358).

Zusammenfassend bieten Wikis Unterstützungsfunktionen für ein kooperatives, selbstgesteuertes, aktives und kritisches Lernen. Dabei können Erkenntnisse strukturiert und wissenschaftlich überarbeitet dargestellt werden, wobei eine kritische Betrachtung und die Editiermöglichkeiten einer dahinter stehenden Gemeinschaft Qualitätsaspekte realisieren. Wikis sind dazu geeignet, erarbeitete Informationen aufzubewahren, die durch eine Gemeinschaft aktuell gehalten und erweitert werden können.

Weitere Technologien Richardson nennt in [Ric06], S. 8f., weitere Web 2.0-Technologien, die Potenziale für den Einsatz in der Lehre haben, wie Tagging/Social Bookmarking und Podcasts.

Das semantische Strukturieren von Inhalten durch Verschlagwortung fördert beim Lernenden ein zielgerichtetes, selbstorganisiertes Lernen. Weiterhin haben Lehrende die Möglichkeit, Lernwege anhand der vergebenen Schlagworte nachzuvollziehen (vgl. [Ric06], S. 93).

Werden für Lerngemeinschaften bestimmte Indizierungskonventionen vereinbart, kann innerhalb vernetzter Lernumgebungen eine kooperativ gepflegte Kollektion hochgradig strukturierter und indizierter Verweise auf interne und externe Informationsquellen entstehen, die über Newsfeeds strukturiert abrufbar sind. Hierüber können kooperative Informationsrecherchen und Informationstausch, kombiniert mit Speicher- und Exportmöglichkeiten realisiert werden (vgl. [Ric06], S. 98)[85].

Darüber hinaus wird das Auffinden und Herstellen sozialer Kontakte zwischen Lernenden mit gleicher Interessenbasis deutlich vereinfacht. Baumgartner sieht diesbzgl. bei großen Bildungsinstituten mit mehreren tausend Eingeschriebenen eine wertvolle Unterstützungsmaßnahme, da Kontakte zwischen Personen mit gleich gelagerten Interessen am ganzen (virtuellen) Campus – unabhängig von den gerade besuchten Kursen – hergestellt werden können (vgl. [Bau06c]).

Audio- und Video-Podcasts werden in der Lehre bereits häufiger eingesetzt, meist in Form von Vorlesungsaufzeichnungen (vgl. [Lit07]). Die Treiber dafür sind einerseits die wachsenden Studierendenzahlen und die daraus resultierenden überfüllten Hörsäle, andererseits die Nachfrage nach ortsunabhängigen und zeitlich flexiblen Blended-Learning-Arrangements, die auf Medien- und Methodenmixen beruhen (vgl. [AWKL06]). Bekannte Beispiele sind hierfür die Podcasting-Plattform für deutsche Hochschulen *podcampus.de* und *iTunes U*, ein Bereich im iTunes Store von Apple, in dem Vorlesungsaufzeichnungen und weitere Inhalte wie Sprachkurse, Labordemonstrationen bis hin zu Führungen über das Hochschulgelände kostenlos von bekannten US-Universitäten zur Verfügung gestellt werden. Gegenüber der reinen Nutzung von Podcasts als weiteren Distributionsweg für Lerninhalte gibt es – wie bei den Weblogs – auch didaktische Ansätze, in denen Studierende zur Produktion eigener Podcasts motiviert werden (vgl. [Lit07]).

[85] Der freie Internet-Dienst *CiteULike.org* hilft bspw. in diesem Zusammenhang Akademikern weltweit, ihre gelesenen wissenschaftlichen Artikel mit anderen zu teilen, zu speichern und zu organisieren. Weitere Beispiele sind die Social Bookmarking-Dienste der University of Pennsylvania (`http://tags.library.upenn.edu/`) und der Harvard Law School (`http://h2obeta.law.harvard.edu/home.do`). Letzter Zugriff: 31.07.2008.

2.3.4.2. Technische Implikationen auf virtuelle Lernumgebungen

Die Berücksichtigung von modernen Technologien und neuem Nutzerverhalten des Web 2.0 aus didaktischer Sicht hat folglich auch große technische Implikationen auf die Konzeption und Entwicklung virtueller Lernumgebungen, die in diesem Abschnitt skizziert werden sollen.

Benutzergenerierte Inhalte und Interaktion Durch die Integration von Werkzeugen wie Weblogs und Wikis in virtuellen Lernumgebungen sowie Möglichkeiten, fremde Inhalte zusammen mit eigenen Materialien in selbstorganisierten Wissensräumen strukturieren zu können, können insbesondere konstruktivistische Ansätze unterstützt werden, die das Lernen wieder stärker als Produktions- und nicht als Rezeptionsprozess begreifen. Dazu müssen Erstellungs- und Verwaltungsfunktionen sowohl für klassische, durch Metadaten formal strukturierte und standardisierte Lernobjekte angeboten werden als auch für benutzergenerierte Inhalte mit niedriger struktureller Komplexität (*Microcontent*, s. [Ale06], [Ker07]), wie Weblog-Posts, Wiki-Einträge oder kommentierte Hyperlinks zu externen Inhalten. Zusätzlich bedarf es einem Angebot von Medienfunktionen[86] zur Interaktion sowohl mit klassischen Lernobjekten als auch mit Microcontent, um neben darstellenden und problemorientierten Lehrformen auch ein selbstorganisiertes Lernen nach konstruktivistischen Ansätzen unterstützen zu können (*aktives bzw. ko-aktives Lernen*, vgl. hierzu [Hes04] und [Kei07]).

Eine werkzeugübergreifende freie Kombinierbarkeit und Wiederverwendbarkeit von Inhalten und somit auch eine durchgängige und medienbruchfreie Verfügbarkeit während des Lernprozesses innerhalb einer virtuellen Lernumgebung impliziert eine konzeptionelle und technologische Generalisierung als Informationsobjekte, unabhängig von Format, Herkunft und Granularität. Zack beschreibt in [Zac99], S. 48, ein solches Konzept und spricht dabei von einer so genannten *knowledge unit* „[...] as formally defined, atomic packet of knowledge that can be labeled, indexed, stored, retrieved and manipulated. The format, size, and content of knowledge units may vary, depending on the type of explicit knowledge being stored and the context of its use." Seine Definition berücksichtigt dabei schon die Benutzung von Informationsobjekten in verschiedenen Kontexten, was ein weiteres wichtiges Moment der semantischen Interoperabilität ist: Durch eine strikte Trennung von Attributen und Inhalten kann das MVC-Entwicklungsmuster (*Model View Controller*, vgl. [BMRS96]) auf Informationsobjekte respektive Strukturen von Informationsobjekten angewendet werden, was eine Wiederverwendung bzw. Weiterbearbeitung von Inhalten in verschiedenen Sichten mit jeweils eigenen responsiven Eigenschaften ermöglicht (vgl. [Kei07]). Eine Sicht wertet dabei jeweils unterschiedliche Attribute eines Objekts aus, ändert aber nicht das eigentliche Objekt.

Ein Beispiel dafür findet sich in [RHS05]: Multimediale Lerninhalte des virtuellen Studienfachs VORMS[87] können durch Umsetzung dieses Konzeptes lernnetzübergreifend

[86] Hampel unterscheidet diesbzgl. in [Ham02], S. 44ff. vier individuelle (Erzeugen, Löschen, Arrangieren und Verknüpfen) und drei kooperative Medienfunktionen (Übertragen, Zugreifen und Synchronisieren).

[87] Das virtuelle Studienfach Operations Research/Management Science (VORMS) wurde 2000 als Verbundprojekt im Förderprogramm Neue Medien in der Bildung des BMBF initiiert. Sieben Lehrstühle

sowohl miteinander als auch mit beliebigen benutzergenerierten Inhalten kombiniert werden. Spezialisierte Lernplattformen und Werkzeuge beschreiben verschiedene Sichten auf diese Informationen. Die Loslösung von den durch LMML (*Learning Material Markup Language*, vgl. [SF01]) und LOM erzeugten – für die Erstellung rein virtueller Kurse durchaus notwendigen – semantischen Abhängigkeiten und Hierarchien ist dabei eine grundlegende Voraussetzung für die freie Kombinierbarkeit und Wiederverwendbarkeit von Lernobjekten in selbstadministrierten virtuellen Wissensräumen unter dem Blickwinkel des selbstorganisierten Lernens. Durch die Generalisierung als Informationsobjekt konnten in diesem Fall verschiedene Aggregationsstufen von Lernnetzen sowie benutzergenerierte Inhalte gleich behandelt und referenziert werden, was zu einer neuen Qualität der semantischen Objektstrukturen und vielfältigen Kommunikations- und Interaktionsmechanismen führte. Hierbei wurden Medienbrüche zwischen Lernwerkzeugen gezielt reduziert, indem Medien- und Kommunikationsfunktionen im Zuge der Generalisierung direkt an Informationsobjekten verankert wurden (vgl. hierzu auch [SHB04]). Auf diese Art und Weise sind synchrone wie asynchrone Interaktions- und Kommunikationsmechanismen unmittelbar mit den semantischen Strukturen der im virtuellen Studienfach VORMS eingesetzten Lernumgebungen verknüpft und stehen nicht – wie in den meisten Systemen als erhebliches Defizit empfunden – weitgehend isoliert neben den eigentlichen Wissensstrukturen.

Offene Schnittstellen für Werkzeuge und Inhalte Die heutige Nutzung des Webs als Plattform basiert nicht mehr nur auf Web-Browsern, sondern auf vielen Diensten und Praktiken, wie Peer-to-Peer-Netzwerke, spezielle Dienste für Foto- und Videospeicherung und -streaming, Chat und Videoconferencing, Web-Services, mobile Szenarios usw.

Um den Ansprüchen der Benutzer gerecht zu werden, die ihre eigene Auswahl und Konfiguration an generischen Werkzeugen für ihre tägliche Wissensarbeit nutzen – bspw. zur Bearbeitung von Dokumenten, zur Kommunikation oder Kontaktpflege – sollte eine moderne Lernplattform nicht mehr als Insellösung im Netz implementiert werden, sondern offene Schnittstellen zu entsprechenden populären Web-Diensten, Werkzeugen und Endgeräten anbieten (vgl. [Wil06], [Ker07])[88]. Somit werden Benutzer in die Lage versetzt, ihre eigene, ganz persönliche Lernumgebung (PLE, *Personal Learning Environment*, vgl. ebd.) aus den Bildungsangeboten, (Kommunikations-)Werkzeugen, Aggregatoren, Web-Diensten und Endgeräten zusammenzustellen und aufeinander abzustimmen. Kerres schreibt dazu in [Ker07]: „For the user, this *personal learning environment* is not a separate space on the internet, it is an essential part of the users workspace. It should be highly integrated with the users' framework of tools for his/her personal use of the internet".

an sechs deutschen Hochschulen haben in der dreijährigen Projektlaufzeit Lehrangebote mit verschiedenen inhaltlichen Schwerpunkten gemäß ihrer Kernkompetenz didaktisch und elektronisch aufbereitet: `http://www.vorms.org`.

[88] „Die Aufforderung, mit einem zum Beispiel in der Lernplattform inkludierten Diskussionsforum, Blog-, Chat- oder Konferenztool zu arbeiten, erscheint so als ob wir von den Studierenden fordern würden, sie müssten ihre Mitschriften auf kariertem Papier mit Bleistiften der Stärke HB mitschreiben und anschließend in Ordnern der Marke X archivieren" [Ker06].

Beispielhaft dafür ist die Kompatibilität von Podcasts mit portablen (videofähigen) Musikabspielgeräten und ihr bequemes Abonnement- und Auslieferungsmodell, das sich unter dem Aspekt einer verbesserten Lernerzentrierung von der herkömmlichen Bereitstellung von Audio- und Videoinhalten in geschlossenen Plattformen deutlich unterscheidet (vgl. [Lit07]).

Weitere hoch integrative Technologien, mit denen offene Lernumgebungen umgesetzt werden können, sind bspw. die auf S. 54ff. vorgestellten RSS-Feeds, offene Programmierschnittstellen und Mashups. Bei der Zusammenstellung und Konfiguration einer solchen persönlichen Lernumgebung kann der Benutzer dann Aggregations-Dienste oder Browser-Erweiterungen zu Hilfe nehmen[89].

Aus Sicht der Implementierung und Bereitstellung virtueller Lernumgebungen impliziert dieser Ansatz einen Fokuswechsel weg vom Funktionsumfang einer Insellösung hin zu einer offenen Infrastruktur: „A PLE is personally constructed and discovered – you don't provide a PLE like you provide the VLE. Its not a box. Working with PLEs is about tailoring the infrastructure to be supportive“ [Wil06], vgl. hierzu auch [KS05a] und [FM06][90].

Die hochintegrativen Technologien brauchen in einer solchen offenen Infrastruktur nicht nur einseitig angeboten werden, sondern können genau so gut dazu genutzt werden, externe Inhalte und Funktionen anderer Plattformen einzubinden und somit den Daten- und Funktionsumfang virtueller Lernumgebungen zu erweitern. So können zum Beispiel Schnittstellen zu externen Inhalten wie virtuellen Bibliotheken, digitalen Verlagsarchiven oder anderen Content-Repositories[91] technisch umgesetzt werden (vgl. bspw. [BHH+06]), ebenso wie Funktionen zur Hot-Spot-Annotation auf Bildern und Grafiken, die von externen Plattformen via Mashup in die Infrastruktur eingebunden werden.

2.4. Zusammenfassung

Um den neuen Anforderungen der Wissensgesellschaft und des lebenslangen Lernens – manifestiert durch Studienreformen, Internationalisierung, Qualitätsentwicklung und der Immatrikulation von Angehörigen einer Generation, die mit neuen Medien und Technologien groß geworden ist (*digital natives*) – wirksam begegnen zu können, bedarf es es eines soliden alltagstauglichen und breitenwirksamen Einsatz von computerunterstütztem Lernen.

E-Learning birgt viele Potenziale für die Vermittlung von Handlungskompetenz, aktuellen Inhalten und problemorientierten Lernen. Insbesondere Web 2.0-Werkzeuge wie Wikis, Blogs und soziale Netzwerke sind durch ihren Fokus auf Interaktivität und Partizipation sowie auf das soziale Feedback geradezu prädestiniert für informelles Lernen, persönliche Wissensorganisation und kooperative Szenarien.

[89] Beispielsweise iGoogle (http://www.google.de/ig), `suprglu.com` oder `greasepot.net`.

[90] Feldstein prägte in der amerikanischen Szene den Begriff *Learning Management Operation System* (LMOS) als Bezeichnung für eine solche offene, technische Infrastruktur.

[91] So bietet beispielsweise das europäische *Ariadne Knowledge Pool System*, das auf einem Netzwerk von verteilten Lernobjekt-Repositories mit Inhalten und Metadaten basiert (s. `http://www.ariadne-eu.org/`), zu jedem ihrer Knoten synchrone und asynchrone SQI-Schnittstellen (*Simple Query Interface*) an, zum Suchen und Auslesen von Inhalten. Der weltweite Pendant dazu ist das *Global Learning Objects Brokered Exchange* (GLOBE, s. `http://globe.edna.edu.au/`).

Die an vielen Universitäten vorhandene, verteilte kooperative IT-Versorgung von Lernumgebungen und Basistechnologien stellt jedoch zumeist keine attraktive Infrastruktur für multimediales Lehren und Lernen dar: Dem zusätzlichen Aufwand für mehrfache Datenerfassung in Kombination mit Kosten für die Inhalteerstellung stehen oft nicht genügend Kosten- oder zumindest Zeiteinsparungen gegenüber, um Dozierende zum Einsatz der neuen Medien in der Lehre zu motivieren[92]. Medienbrüche werden durch monolithische Lernmanagementsysteme und proprietäre Applikationen eher erzwungen, als das die einzelnen Lerninhalte durchgängig und systemübergreifend in Lernprozessen verfügbar wären. Auch die Einbettung mediengestützter Lehr- und Lernprozesse in institutionale Kontexte wird durch proprietäre Schnittstellen erschwert, sodass dadurch höhere Entwicklungs-, Wartungs- und Prozesskosten entstehen. Hier besteht Handlungsbedarf.

[92] Optimistische Schätzungen gingen 2004 davon aus, dass zu dem Zeitpunkt nur 5 % aller Hochschullehrenden neue Medien in der Lehre einsetzen (vgl. [CB04], S. 10).

3. Terminologiebasierte Spezifizierung von Lernumgebungen

Werden universitäre E-Learning-Werkzeuge individuell entwickelt, ist die Abstimmung zwischen Anwendern, Didaktikern, Mediendesignern und Programmierern essenziell, um ein gemeinsames Verständnis für die Aufgabenstellung sicherzustellen. In Unterkapitel 1.3 wurde die Kommunikation zwischen den Projektbeteiligten jedoch als oftmals gestört charakterisiert: Schwierig sind zum einen die unterschiedlichen Blickwinkel und Intentionen von Begriffen und Konzepten, zum anderen der Bruch zwischen der präformalen Alltagssprache der Anwender und der zur Spezifikation häufig genutzten formalen künstlichen Sprachen und Diagrammmethoden.

In diesem Kapitel wird ein normsprachlicher Ansatz zur Spezifikation erarbeitet mit dem Ziel, ein alltagstaugliches Werkzeug zur Anforderungsbeschreibung universitärer Lernumgebungen zu schaffen, das zur Verbesserung der Kommunikation beiträgt und mit dem dadurch zeitintensive prototypische Entwicklungen oder gar kostenintensive Fehlentwicklungen deutlich reduziert werden können.

Zunächst wird das Konzept der terminologiebasierten Softwareentwicklung eingeführt (3.1). Dazu wird die Problemstellung durch eine genauere Betrachtung möglicher Kommunikationsstörungen konkretisiert und Normsprachen als ein Hilfsmittel vorgestellt, an denen sich Kommunikation störungsfrei ausrichten kann. Darüber hinaus wird erläutert, wie eine Normsprache im Softwareentwicklungsprozess eingesetzt werden kann.

Im Abschnitt 3.2 wird die Wissensraummetapher als ein semantisches Bezugssystem hergeleitet, dessen Konstrukte zur Beschreibung von Lern- und Arbeitsszenarien verwendet werden können. Es wird begründet, dass diese Metapher sowohl aus lerntheoretischer als auch aus kognitionspsychologischer Sicht dazu geeignet ist.

Auf Basis der Wissensraummetapher wird in 3.3 der terminologiebasierte Ansatz für die objektorientierte Spezifikation von Schienmann (TAOS, vgl. [Sch95]) zur normsprachlichen Beschreibung von Lern- und Arbeitsszenarien angewendet. Dazu wird eine Terminologie von Begriffen vorgestellt, die neben den Konstrukten der Wissensraummetapher auch organisatorische und systemtechnische Konzepte umfasst. Auf dieser Grundlage werden die zur Spezifizierung notwendigen Satzmuster konstruiert und ihre Anwendbarkeit zur Beschreibung statischer, funktionaler und dynamischer Aspekte von Lernumgebungen demonstriert.

Das Kapitel schließt mit einer Zusammenfassung.

3.1. Terminologiebasierte Softwareentwicklung

Terminologiebasierung ist zu verstehen als eine Untersuchung natürlichsprachlicher Äußerungen in Hinsicht auf ihre Intention oder Kommunikation im Anwendungsbereich. Diese werden schrittweise und kontrolliert in intersubektiv begründete Aussagen auf Basis einer normierten Fachsprache überführt. Der wesentliche Punkt dabei ist, dass informationsverarbeitende Handlungen im Anwendungssystem auf Fachbegriffen bzw. Fachkonzepten basieren, die zum konstruierenden Bestand der Theorie eines Fachbereichs gehören. Diese Fachbegriffe sind System-Designern und -Entwicklern nicht unmittelbar einsichtig, wohl aber dem Anwender aus dem Fachbereich, was einen sprachkritischen Rekonstruktionsprozess notwendig macht, der in der terminologiebasierten Softwareentwicklung als Teil eines Entwicklungssystems verwaltet wird (vgl. [Ort95], S. 148, [Sch97a], S. 11).

In diesem Abschnitt soll der terminologiebasierte Entwicklungsansatz näher erklärt werden. Zuerst werden jedoch die genannten Kommunikationsstörungen näher spezifiziert, um dadurch das Problemverständnis zu stärken.

3.1.1. Effekte und Defekte natürlicher Sprachen

Das Ermitteln von Anforderungen, Zielen und Rahmenbedingungen, auf deren Grundlage das spätere System entwickelt wird, geschieht zumeist unter Zuhilfenahme von Ermittlungstechniken wie Interviews, Dokumentenanalyse und Beobachtungen etc. und dem damit verbundenen Konsultieren von „fachlich kompetenten Quellen“ (vgl. [Rup07], S. 108)[1]. Neben weiteren Anforderungsquellen wie Dokumente und bereits existierende Systeme, stellen diese *Stakeholder*[2] eine Quelle für Aussagen über das zu entwickelnde System dar (vgl. [Poh07], S. 314).

Die Kommunikation zwischen Systemanalysten und Stakeholdern verläuft meistens natürlichsprachlich, da natürliche Sprachen den Befragten i. d. R. besser verständlich sind als die formalen Diagramm- oder Beschreibungssprachen, die seitens der Softwareentwickler zur Spezifikation des Systems genutzt werden. Jedoch weisen natürliche Sprachen zwei grundsätzliche Aspekte auf, die zum Teil große Abweichungen gegenüber einer präzisen, einfachen Formulierung von Aussagen bedingen: Spracheffekte und Sprachdefekte.

3.1.1.1. Sprachliche Effekte

Eine natürliche Sprache ist immer Ausdrucksmittel der persönlichen Wirklichkeit eines jeden Individuums. Ausgangspunkt dafür ist die Annahme, dass jeder Mensch die Welt anders wahrnimmt (*selektive Wahrnehmung*). Bei sprachlichen und schriftlichen Äußerungen muss ein Individuum zunächst geistig eine vollständige Repräsentation für

[1] Mehr Informationen zu Methoden der Anforderungsermittlung und Erhebungstechniken sind zu finden in [RR99], S. 82ff., [Sch02], S. 196ff., [Poh07], S. 323ff., [Tha02], S. 91ff. und [Rup07], S. 115ff.

[2] „Many people and organizations are interested in the construction of a software system. We call these *stakeholders*: The customer, the end users, the developers, the project manager, the maintainers, and even those who market the system are a few examples“ [BCK03], S. 6.

den zu äußernden Aspekt auf Grundlage seiner persönlichen Wirklichkeit herstellen (*Tiefenstruktur*, vgl. [SOP01], S. 7). Ausgehend von dieser Repräsentation bildet es dann ein sprachliches Konstrukt (*Oberflächenstruktur*), welches es äußert oder niederschreibt. Dieses durchlief zu dem Zeitpunkt, an dem es geäußert oder niedergeschrieben wird, bereits ein oder mehrere Transformationsprozesse. So genannte Darstellungstransformationen werden durchgeführt, wenn die persönliche Wirklichkeit mithilfe von Sprache ausgedrückt wird (vgl. ebd.). Hierbei lassen sich drei verschiedene Transformationsarten unterscheiden, durch die Informationsverluste oder Falschaussagen bei der Anforderungserhebung entstehen können: *Tilgung*, *Generalisierung* und *Verzerrung*. Diese Transformationen haben wiederum *sprachliche Effekte* zur Folge, anhand deren sie der Analyst im Rahmen einer grammatikalischen und semantischen Analyse erkennen und ggf. beheben kann[3].

Tilgung mithilfe der *Tilgung* wird die Komplexität der Realität auf ein Ausmaß reduziert, mit dem der Mensch umgehen kann. Es findet eine Selektion statt, bei der die Wahrnehmung nur auf einen relevanten Ausschnitt der Wirklichkeit gelenkt wird (vgl. [Rup07], S. 143). Wichtige Anforderungen können dabei verloren gehen. Die sprachlichen Effekte der Tilgung sind:

Unvollständig spezifizierte Prozesswörter sind Verben, Adjektive oder Adverbien, die durch mehr als ein Substantiv beschrieben werden müssen, um als vollständig angesehen werden zu können (vgl. [Rup07], S. 148).

Unvollständige Vergleiche und Steigerungen liegen vor, wenn keine Bezugspunkte angegeben sind. Außerdem müssen Vergleiche und Steigerungen immer messbar sein (vgl. [Rup07], S. 151).

Modaloperatoren der Notwendigkeit sind Begriffe wie *müssen* oder *sollte*, bei denen Forderungen an das Verhalten des späteren Systems gestellt werden (vgl. ebd., S. 153). Hier gilt es zu beachten, dass nicht nur das Normalverhalten beschrieben wird, sondern auch das Ausnahmeverhalten (vgl. ebd.). Ein Begriff wie *sollte* impliziert immer das mögliche Vorhandensein einer Ausnahme: „Das System sollte die Zulassung prüfen“. Was passiert jedoch, wenn eine Überprüfung aus technischem Grund nicht möglich ist? Für diesen Fall gilt es eine Ausnahme zu definieren.

Implizite Annahmen sind Fakten, die „erfahrenen Anwendern völlig selbstverständlich sind, sodass [...] sie bei der Erhebung von Anforderungen gar nicht mehr kommuniziert werden“ ([Rup07], S. 154). Diese Annahmen gilt es auf jeden Fall zu dokumentieren, da ein wenig erfahrener Anwender diese nicht unbedingt für selbstverständlich halten muss und bei der Implementierung falsch getroffene Annahmen die korrekte Funktionalität des Systems beeinträchtigen können (vgl. [SOP01], S. 10).

[3] Eine Methodensammlung zur Aufdeckung und Behebung dieser Defizite stellt bspw. das SOPHIST-REgelwerk dar (vgl. [SOP01]).

Generalisierung Bei der *Generalisierung* werden „Elemente oder Teile eines persönlichen Modells von der ursprünglichen Erfahrung abgelöst [...], um dann die gesamte Kategorie, von der diese Erfahrung ein Beispiel darstellt, zu verkörpern“ [BG94], S. 35. Durch diese Verallgemeinerung können wichtige Sonder- und Fehlerfälle häufig übersehen werden (vgl. [Rup07], S. 157). Die sprachlichen Effekte der Generalisierung sind:

Universalquantoren wie *immer*, *alle* oder *nie* spezifizieren Häufigkeiten und treffen somit eine Aussage über das Verhalten von einer Menge von Objekten (vgl. [Rup07], S. 157). Die Gefahr besteht in der Verallgemeinerung und der Nichtbeachtung von Ausnahmen. Das festgelegte Verhalten muss nicht zwangsläufig auf alle Objekte der Menge zutreffen.

Unvollständig spezifizierte Bedingungen liegen vor, wenn nicht alle möglichen Verhalten eines solchen Konstruktes (*wenn ... dann ...*) definiert werden. Häufig wird vergessen zu spezifizieren, was passiert, wenn die Bedingung nicht eintritt (vgl. [SOP01], S. 15).

Substantive ohne Bezugsindex sind ähnlich wie *unvollständige Prozesswörter* nicht ausreichend beschrieben.

Verzerrung Die *Verzerrung* schließlich ermöglicht es dem Menschen, die Realität auf Grundlage von gemachten Erfahrungen zu verändern (vgl. [BG94], S. 37). Hierdurch kann ein für einen neutralen Betrachter verzerrtes Bild der Realität entstehen, da die Realität und Erfahrung umgestaltet oder sogar verfälscht werden kann (vgl. [Rup07], S. 143, S. 162). Besonders die Verzerrung ist für den Analysten problematisch, weil er nicht beurteilen kann, ob die ihm dargelegte Formulierung des Stakeholders richtig oder verzerrt sind. Die sprachlichen Effekte der Verzerrung sind:

Nominalisierung stellt eine „komplexe Transformation dar, die ein Verb aus der Tiefenstruktur in ein Substantiv der Oberflächenstruktur verwandelt“ [Rup07], S. 163. Diese Substantive stehen dann häufig repräsentativ für einen ursprünglich umfangreicheren Prozess. Ein Informationsverlust ist vorprogrammiert.

Funktionsgefüge sind „Kombinationen aus inhaltsarmen Verben (machen, können, haben, sein, ...) und sinngebenden Substantiven“ [Rup07], S. 165. Das Problem bei diesen Konstrukten ist die mangelhafte Präzisierungsfähigkeit dieser Verben. Sie beschreiben den Prozess nicht ausreichend.

Doch nicht nur die Anwendung der Sprache als Ausdrucksmittel der individuellen Wirklichkeit, sondern auch die Sprache an sich kann Defizite aufweisen, die Störungen in der Kommunikation hervorrufen können.

3.1.1.2. Sprachdefekte

Zur Erklärung von Sprachdefekten wird zunächst einmal ein Begriffsmodell hergeleitet: Begriffe sind eine abstrakte, gedankliche Darstellung der wesentlichen Merkmale von Dingen. Sie sind Mittel des gedanklichen Ordnens (Begreifens) und werden darum auch

zur Kommunikation zwischen Menschen eingesetzt (vgl. [Bro04], S. 231). Nach [Ort97] hat ein Begriff neben seiner *Bezeichnung* zur Identifizierung (Begriffswort, Benennung) eine *Intension* (Begriffsinhalt, Schema) und eine *Extension* (Begriffsausprägung, Begriffsumfang). Die Ebene der Intension definiert durch eine Menge von Kriterien das Schema eines Begriffes, anhand dessen entschieden werden kann, ob ein bestimmtes Objekt unter einen Begriff fällt oder nicht. Komplementär dazu repräsentiert die Extension eines Begriffes die Gesamtheit aller Objekte, die genau diesem Begriff zugeordnet werden können. Ortner unterscheidet dabei zwei Ebenen der Extension: (1) Konkrete oder abstrakte Gegenstände, die unter den Begriff fallen, (2) singuläre softwaretechnische Abbildungen oder Beschreibungen der Gegenstände (vgl. Abb. 3.1).

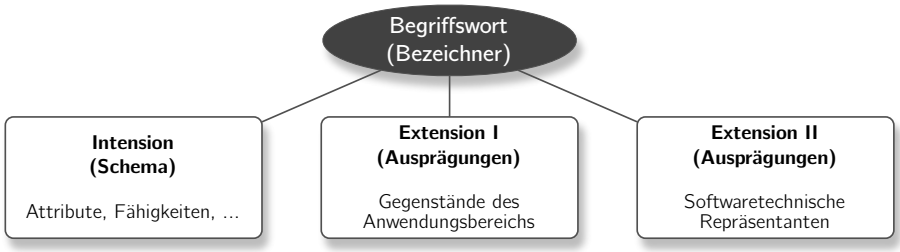

Abbildung 3.1.: Begriffsmodell (eigene Darstellung in Anlehnung an [Ort97], S. 84f., [Sch97a], S. 143).

Da bei natürlichsprachlicher Kommunikation die Intension und Extension der verwendeten Begriffe von der individuellen gedanklichen Darstellung aller Beteiligter abhängig ist, kann es bei Anforderungserhebungen und Systemanalysen zu Kommunikationsstörungen kommen: Die Zuordnung zwischen der Intension des Systemanalysten und den von den Stakeholdern verwendeten Begriffsbezeichnungen bzw. den Begriffsbezeichnungen und ihrer Extension ist nicht in jedem Fall eindeutig. Dadurch können verschiedene Sprachdefekte entstehen (vgl. [Ort93]):

- Als Synonyme werden Begriffe mit identischer Intension und Extension bezeichnet, deren Bezeichner dadurch gegeneinander ausgetauscht werden können. Der Gebrauch synonymer Begriffe ist dann unproblematisch, wenn allen Beteiligten diese Eigenschaft bekannt ist.
- Homonyme sind Begriffe, die verschiedene Intensionen und Extensionen haben, jedoch einen gemeinsamen Bezeichner besitzen. Dies kann zu schweren Missverständnissen führen, sodass bei der Anforderungserhebung in solchen Fällen ein Kontext angegeben werden muss, damit die Bezeichner unterschieden werden können.
- Eine Äquipollenz meint die unterschiedliche Bezeichnung eines Objektes (Extension) aufgrund unterschiedlicher Blickwinkel (Intension).
- Von Vagheit spricht man, wenn zwischen Begriffen keine inhaltlich (intensional) klare Abgrenzung vorhanden ist und dadurch die Zuordnung von Objekten zu Begriffen unklar ist. Durch eine Präzisierung der Intension können Vagheiten beseitigt werden.

Insgesamt hängt die Häufigkeit des Auftretens und die Bandbreite der sprachlichen Effekte und Defekte bei der Anforderungsanalyse tendenziell von der verwendeten Erhebungstechnik ab. Im Rahmen von Interviews oder Selbstaufschreibungen sind eher unstrukturierte, verkürzte oder ad-hoc formulierte Aussagen zu erwarten (vgl. [Leh98b]). Gewünscht sind jedoch einfache, aber präzise Formulierungen von intersubjektiven Aussagen, die von allen am Anforderungsprozess beteiligten Personen gleichermaßen verstanden werden.

3.1.2. Normsprachen

So genannte *kontrollierte Sprachen*[4] wie zum Beispiel ASD Simplified Technical English (ASD-STE100, vgl. [ASD07]) oder das Siemens DokumentationsDeutsch (SDD, vgl. [Sch96]) sind Teilmengen natürlicher Sprachen. Durch Vereinfachungen[5] soll sprachlichen Effekten vorgebeugt und Sprachdefekte behoben und so für mehr Eindeutigkeit in Wortwahl und Satzbau, Konsistenz im Ausdruck und für eine bessere Verständlichkeit gesorgt werden (vgl. [Göp07]).

Normsprachen sind spezielle kontrollierte Sprachen. Sie haben ihren Ursprung in der konstruktiven Wissenschaftstheorie[6] und werden durch eine methodische Rekonstruktion der in einem Anwendungsbereich gesprochenen natürlichen (Fach-)Sprache entwickelt. Die Rekonstruktion umfasst die Klärung der Bedeutung von Wörtern, ihre eindeutige Definition sowie ihre Beziehungen untereinander und Regeln für ihren Gebrauch. Dadurch werden vage und homogene Bezeichnungen entfernt und synonyme Bezeichnungen nur in kontrollierter Form zugelassen werden (vgl. [Sch97a], S. 15, [Ort00], [Leh98a], S. 366; vgl. hierzu auch Abschnitt 4.1.3).

Das Vokabular einer Normsprache setzt sich zu einem Teil aus allgemeinen, themeninvarianten (sachgebietinvarianten) Wörtern zusammen, die für die Repräsentation von normsprachlichen Aussagen benötigt werden (vgl. [Lor00], S. 29ff., [Ort95], S. 153f.). Diese so genannten *Strukturwörter* (syn. *Partikel*) stehen für Beziehungen und Beziehungseigenschaften zwischen Gegenständen eines Realitätsausschnitts. Sie nehmen daher bei der Aussagenbildung eine syntagmatische, strukturierende Bedeutung ein. Dazu gehören Wörter wie Artikel, Pronomen (*Determinans*) oder Präpositionen.

Der zweite Teil des Normsprachenvokabulars definiert Taxonomien von *Themen- oder Fachwörtern*, die zur Bezeichnung von Gegenständen eines Realitätsausschnitts verwendet werden (vgl. ebd.). Diese umfassen beispielsweise Verben, Substantive, Adjektive und Adverbien (vgl. [Sch97a], S. 148f.). Anteilsmäßig wird die Menge an Fachwörtern gegenüber einer relativ fixen Menge von Strukturwörtern im Rahmen von Anpassungen und Erweiterungen einer Normsprache zunehmen, so dass hierfür ein kontrollierter Prozess einzuplanen ist (vgl. [Ort95], S. 156)[7].

[4] Die deutsche Bezeichnung *kontrollierte Sprache* ist auf einen Übersetzungsfehler zurückzuführen: Das englische Verb *to control* bedeutet nicht kontrollieren, sondern vielmehr *steuern* oder *regeln*. Eine *Controlled Language* ist also eine *geregelte* Sprache (vgl. [Göp07]).

[5] Darunter zählen bspw. eingeschränkter Wortschatz, eingeschränkte Grammatik, Regeln für die Konstruktion von Sätzen (so genannte *Satzbaupläne*), evtl. Formatierungsbeschränkungen etc.

[6] Kamlah und Lorenzen schlagen in [KL96] die Einführung einer *Orthosprache* (gr.: orthos, richtig) in allen Fachwissenschaften vor, mit der eindeutige und präzise Aussagen methodisch und zirkelfrei formuliert werden können. Darauf basierend führte Schienmann in [Sch97a], S. 15, dann die Bezeichnung *Normsprache* ein.

[7] Dazu Hellmuth in [Hel97], S. 43: „Terminologiemanagement darf dabei jedoch nicht als einmaliger

Um gültige Aussagen zu bilden und somit auf Basis des vorhandenen Vokabulars weitere Fachbegriffe rekonstruieren zu können, bedarf es neben dem Vokabular noch einer formalen Grammatik mit semantischen Regeln, aus der so genannte *Satzbaupläne* abgeleitet und definiert werden können[8].

An dieser Stelle sollen die genannten Elemente einer Normsprache – Themen- und Fachwörter, Strukturwörter und formale Grammatik – näher erläutert werden, da sie die Grundlage für das Verständnis der in einem späteren Abschnitt des Kapitels konstruierten Normsprache bilden.

Themen- und Fachwörter Themen- und Fachwörter werden in einer Normsprache zunächst zum Aufbau des Vokabulars verwendet, indem neue Begriffe formal durch bereits bestehende Begriffe rekonstruiert werden, wie bspw. durch die Aussage *Student ε („ist“) Person*. Die Aussage definiert eine intensionale Beziehung zwischen der Menge der Studenten und der Menge der Personen. Diese auf Basis von Prädikatenlogik und expliziten Definitionen untereinander abgegrenzten *Prädikatoren* heißen *Termini*. Eine *Terminologie* umfasst sowohl Termini als auch die dazu gehörigen Prädikatorenregeln und Definitionen und stellt damit den materiellen Teil einer Normsprache dar (vgl. [Sch97a], S. 124).

Mithilfe dieser Termini können dann Aussagen zu dem Anwendungsbereich formuliert werden, den das spätere System unterstützen soll. Damit diese Aussagen sich auf einen konkreten Realitätsausschnitt beziehen können, müssen konkrete Objekte durch Eigennamen oder (deiktische) Kennzeichnungen[9] benannt werden können. Dazu wird eine Unterscheidung zwischen *Nominatoren* (bspw. „Meier“) und *Prädikatoren* (bspw. „Student“) gemacht:

❏ Natürliche Sprache:	[Meier ist ein Student.]	
❏ Normsprache:	$[Meier \mid \varepsilon \mid Student]$	(3.1)
❏ Satzbauplan:	$[N \mid \varepsilon \mid Q]$	

Während konkrete Objekte durch Nominatoren (N) benannt werden können, charakterisieren Prädikatoren im Unterschied dazu ihre Eigenschaften. Sie stellen also ihren Begriffsbezeichner und nehmen somit auf intensionaler Ebene eine ordnende Funktion ein, da mittels dieser Bezeichner eine Menge an Objekten gleichgesetzt und von anderen gleichgesetzten Objekten unterschieden werden können (vgl. [Lor00], S. 29).

Schritt angesehen werden, sondern muß sich vielmehr als ein methodisch kontrollierter und auch permanenter Prozeß der Verständigung auf unterschiedlichen Ebenen und zu unterschiedlichen Phasen des Produktentstehungsprozesses verstehen“.

[8] In einer Normsprache getätigte Aussagen sind daher allein schon aufgrund ihrer Form gültig, unabhängig vom verwendeten Fachvokabular (vgl. [Leh98a], S. 366).

[9] „Deiktische Kennzeichnungen sind sprachliche Ausdrücke, welche ihre benennende Funktion erst durch den Bezug auf die jeweilige Sprechsituation erhalten. In Aussagen wird die Kontextabhängigkeit deutlich an der Verwendung von Demontrativpronomen oder Indikatoren wie ≫dies≪, ≫jenes≪, ≫heute morgen≪oder ≫dort drüben≪“ [Sch97a], S. 151. Sie sollten aufgrund ihrer Kontextabhängigkeit jedoch möglichst vermieden werden (vgl. ebd.).

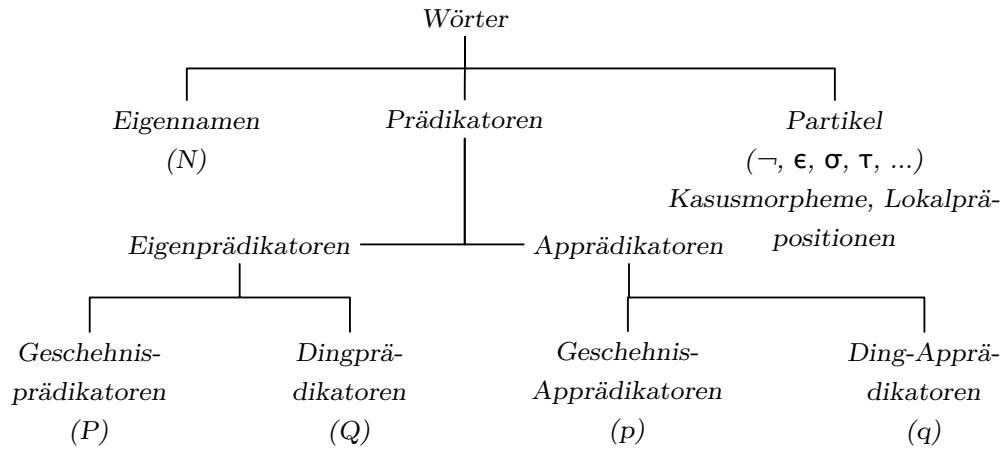

Abbildung 3.2.: Klassifikation von normsprachlichen Wortarten nach Lorenz (eigene Darstellung, in Anlehnung an [Lor00], S. 52).

In der Klassifizierung nach Lorenzen (vgl. ebd., S. 29ff.) werden darüber hinaus Prädikatoren in *Dingprädikatoren* (Q) und *Ding-Apprädikatoren* (q) zur Bezeichnungen und Beschreibung von Gegenständen unterschieden, sowie in *Geschehnisprädikatoren* (P) und *Geschehnis-Apprädikatoren* (p) zur Bezeichnung und Beschreibung von Aktivitäten[10]. Demnach geben die Prädikatoren die primäre Ordnung von Gegenständen und Aktivitäten einer Domäne wieder, Apprädikatoren charakterisieren sie. Diese Klassifikation lehnt sich an eine klassische Trennung von Wortarten in Substantive, Verben, Adjektive und Adverbien an (vgl. ebd., S. 52)[11]. Somit können auch reichhaltigere umgangssprachliche Aussagen in einer Normsprache modelliert werden:

❏ Natürliche Sprache:	[Müller legt computergestützt einen neuen Kurs an.]	
❏ Normsprache:	[Müller | π | computergestützt anlegen | neu Kurs]	(3.2)
❏ Satzbauplan:	[N | π | pP | qQ] mit Tatkopula π (lies: *tut*)	

Durch die Prädikatoren und Nominatoren einer Normsprache ist die Möglichkeit einer konkreten *Gegenstandseinteilung* von Dingen, Eigenschaften und Geschehnissen einer Domäne gegeben. Somit lassen sich Ordnungs- und Teilungs-/Verbindungsaussagen formulieren, aus denen man den Aufbau und die Struktur eines Systems rekonstruieren kann. Aussagen zu Geschehnissen können sich analog dazu auf Operationen (Fähigkeiten,

[10] Da im weiteren Verlauf dieser Arbeit bzgl. der Satzbaupläne auf die Arbeit [Sch97a] von Schienmann verwiesen wird, werden im Folgenden auch seine Variablenbezeichner verwendet.

[11] Schienmann weist in [Sch97a], S. 155, jedoch darauf hin, dass Aktivitäten einer Normsprache unter semantischen Gesichtspunkten nicht immer nur durch umgangssprachliche Verben beschrieben werden müssen, sondern auch Konstrukte wie „Kursanlegen“ oder „Kurs_anlegen“ verwendet werden können.

Methoden) mit Objekten und das Systemverhalten beziehen (vgl. [Ort95], S. 155). Eine solche Einteilung ist als grundlegendes Lösungsprinzip in vielen softwaretechnischen Methoden implizit[12], im Kontext einer an eine natürliche Sprache angelehnte Normsprache jedoch zunächst einmal unabhängig von einem bestimmten softwaretechnischen Paradigma. Damit darüber hinaus aber auch eine Korrektheit von Teilen des Entwurfs und der Entwicklungsresultate beweisbar und dadurch verifizierbar wird, werden sowohl in den Strukturbedingungen des Entwicklungssystem (formal) als auch bei der Rekonstruktion der Themen- und Fachwörter (thematisch) logische Konstrukte angewandt.

Eine Normsprache muss nach Ortner drei Intensionen des Wahrheitswertbegriffes adressieren (vgl. [Ort95], S. 150; vgl. dazu auch [Sch97a], S. 93):

- Eine Aussage ist *formal wahr*, wenn sie eine zugelassene Satzstruktur (Satzbauplan) erfüllt (formal-logischer Wahrheitswertbegriff)

- Eine Aussage ist *thematisch wahr*, wenn sie nur auf definierten (rekonstruierten) Themenwörter basiert (terminologischer Wahrheitswertbegriff)

- Eine Aussage ist *empirisch oder faktisch* wahr, wenn die Aussage dem zu modellierenden Realitätsausschnitt auch tatsächlich entspricht (empirischer Wahrheitswertbegriff).

Eine gültige normsprachliche Aussage, die empirisch oder faktisch wahr ist, ist damit implizit sowohl thematisch wahr als auch formal (vgl. ebd.; vgl. auch [Sch97a], S. 124ff.). Da das empirische oder faktische Wahrheitskriterium in diesem Kontext die Übereinstimmung einer Modellinstanz (bspw. Ausprägung des Datenschemas) mit der Realität betrifft, verbleibt für die Formalisierung der Logik in einer Normsprache – wie oben bereits angedeutet – zum einen das Entwicklungssystem selbst, als auch das zum Entwurf verwendete Vokabular an Fachwörtern. Daher werden logische Konstrukte (1) zur Aussagenbildung (*grammatische Partikel*) und (2) zur Rekonstruktion des Vokabulars (*logische Partikel*) unterschieden.

Grammatische Partikel Die Grammatik einer Normsprache ist Teil des Entwicklungssystems. Daher muss sie sprachliche Möglichkeiten bieten, ein System durch Aussagen zur Struktur (Attribute und Beziehungen), Funktionalität (Fähigkeiten und Interaktionen) und Dynamik (Zuständen und Reihenfolgen) eines Problembereichs zu spezifizieren (vgl. [Sch97a], S. 51f.). Dies ist durch die alleinige Nutzung von Fachwörtern nicht präzise genug möglich, weshalb das Vokabular einer Normsprache darüber hinaus weitere Hilfskonstrukte umfassen muss.

[12] Beispielsweise die Einteilung nach Objekten, Attributen und Operationen beim objektorientierten Paradigma, im relationalen Datenbankentwurf die Gegenstandseinteilung nach *Entities* und *Relations* oder beim Entwurf von Expertensystemen eine Gegenstandseinteilung nach Fakten, Regeln und Schlussarten.

Insbesondere Hilfsverben wie *sein*, *haben* und *werden* können in Ordnungs-, Teilungs-/ Verbindungs- und Geschehnisaussagen als Kopulae[13] das Subjekt mit Prädikatoren verbinden:

Kopula	Affirmativ	Negativ
❑ Seinskopula	ε (lies: *ist*)	ε' (lies: *ist nicht*)
❑ Tatkopula	π (lies: *tut*)	π' (lies: *tut nicht*)
❑ Fähigkeitskopula	γ (lies: *kann*)	γ' (lies: *kann nicht*)
❑ Widerfahrniskopula	τ (lies: *wird*)	τ' (lies: *wird nicht*)
❑ Teilungskopula	σ (lies: *hat*)	σ' (lies: *hat nicht*)
❑ Berechtigungskopula	ϑ (lies: *darf*)	ϑ' (lies: *darf nicht*)

(3.3)

Mit diesen Kopulae lassen sich Aussagen bilden wie [607080 | ε | $Matrikelnummer$], [$Meier$ | σ | 607080], oder [Müller | ϑ' | *benoten* | Meier]. Weiterhin können Präpositionen wie *aus*, *nach*, *mit* und Determinans wie *der*, *die*, *das* als Mittel zur präzisen Aussagenformulierung vonnöten sein.

Logische Partikel Die Beziehungen zwischen Themen- und Fachwörtern untereinander müssen so präzise beschrieben werden können, dass sich allein durch ihre Verwendung Aussagen als thematisch wahr oder falsch beweisen lassen. Zur logischen Abgrenzung oder zur expliziten Definition von Begriffen werden Junktoren und Quantoren der Prädikatenlogik genutzt:

❑ Negator:	$\neg$	(lies: *nicht*)
❑ Konjunktor:	$\wedge$	(lies: *und*)
❑ Adjunktor:	$\vee$	(lies: *oder*)
❑ Subjunktor:	$\rightarrow$	(lies: *wenn, dann*)
❑ Äquijunktor:	$\leftrightarrow$	(lies: *genau dann, wenn*)
	sowie	
❑ Einsquantor:	$\exists$	(lies: *es gilt für ein*)
❑ Allquantor:	$\forall$	(lies: *es gilt für alle*)

(3.4)

Um beispielsweise auszudrücken, dass Studenten nicht mehr als zehn Kurse belegen dürfen, kann man folgende Aussage formulieren:

$$\forall\, x\, \varepsilon\, Student \overset{\leq 10}{\exists}\ y\, \varepsilon\, Kurs\ (x \mid \pi \mid belegen \mid y) \tag{3.5}$$

Ebenso kann durch die Festlegung semantischer Merkmale mithilfe von Konjunktoren die Intension des Prädikators *Student* im Lexikon bestimmt werden:

$$\begin{aligned} x\ \varepsilon\ Student \overset{\text{def}}{=} x\ \varepsilon\ Person\ \wedge\ \exists\, m\ (x \mid hat\_Matrikelnummer \mid m) \\ \wedge\, \exists\, s\ (x \mid ist\_eingeschrieben \mid s)\ \ldots \\ \text{für m} \in \text{Matrikelnummer, s} \in \text{Studiengang, } \ldots \end{aligned} \tag{3.6}$$

Ein Student ist demnach eine Person, die eine Matrikelnummer hat und in einem Studiengang eingeschrieben ist.

[13] Das Hilfsverb wird hierbei als eigenständiges Verb verwendet, vgl. bspw. 3.7 oder 3.9.

3.1.2.1. Satzbaupläne

Satzbaupläne beschreiben die grammatische Struktur von normsprachlichen Sätzen, mit denen umgangssprachliche Aussagen rekonstruiert werden können. Mit diesen Mustern ist eine bestimmte Grundbedeutung von Sätzen verbunden, die sich an der Einteilung eines gewählten Spezifikationsrahmens im Softwareentwurf orientieren muss.

In den oben angeführten Beispielen wurden bereits solche Satzbaupläne vorgestellt. So beschreibt (3.1) das normsprachliches Satzmuster für die elementare *Dingprädikation*, durch die ein durch den Nominator N in der Subjektstellung bezeichnetes konkretes Objekt einer Menge von Objekten auf extensionaler Begriffsebene zugeordnet wird. Solche Aussagen, die eine Ausprägung eines Schemas beschreiben, bezeichnet Ortner als *singuläre Aussagen*. Für die Rekonstruktion von Fachbegriffen durch vorhandenes Vokabular und somit für die Entwicklung eines konzeptuellen Schemas essentiell sind darüber hinaus auch *allgemeine Aussagen*, die semantische Beziehungen zwischen Prädikatoren näher beschreiben. Dazu kann das in (3.1) beschriebene Satzbaumuster verallgemeinert werden, um auch umgangssprachliche Aussagen zu Generalisierungsbeziehungen rekonstruieren zu können:

$$[N_1 \mid \varepsilon \mid N_2] \tag{3.7}$$

Ein Nominator kann hierbei in singulären Aussagen weiterhin zur Bezeichnung eines konkreten Objektes herangezogen werden, darüber hinausgehend aber auch einen Begriff bezeichnen, wonach „'Student' ε 'Person'“ eine ebenso gültige Aussage ist wie „Meier ε 'Student'“[14].

Analog zu (3.7) kann auch ein Satzmuster zur Beschreibung von Kompositionsbeziehungen in singulären oder in allgemeinen Aussagen aufgebaut werden:

$$[N_1 \mid \sigma \mid N_2] \tag{3.8}$$

Als Beispiel mag die Aussage „Kurs σ Teilnehmergruppe“ dienen.

Die Satzbaupläne (3.7) und (3.8) ermöglichen die Rekonstruktion der statischen Struktur einer Domäne, der Satzbauplan in (3.2) bezieht sich im Gegensatz dazu auf die Beschreibung von Verhalten eines Systems innerhalb dieser Domäne, also Interaktionen zwischen Instanzen des konzeptuellen Schemas. Um komplexere Satzstrukturen zu Verhaltensaussagen rekonstruieren zu können, kann das Satzmuster um indirekte Objekte erweitert werden:

$$[N \mid \pi \mid pP \mid q_1Q_1 \mid q_2Q_2^{Fall} \mid \ldots \mid q_nQ_n^{Fall}] \tag{3.9}$$

Indirekten Objekten werden in Normsprachen so genannte *Kasusmorpheme* beigestellt, die den jeweiligen Fall des Objektes anzeigen (vgl. [Lor00], S. 46f.)[15].

[14] Die Hochkommata dienen in diesem Fall nur der Verdeutlichung der erweiterten Nominatorenfunktion und können auch weggelassen werden.

[15] Nach Lorenzen lassen sich mindestens drei Fälle unterscheiden: 1) indirektes Objekt als Hilfsmittel (*Mittelfall*): „Student | π | belegen | Kurs | mit-OnlineAnmeldung“, 2) indirektes Objekt als Ergebnis der Ausführung (*Werkfall*): „Dozent | π | überführen | Anmeldung | zu-Zulassung“, und 3) indirektes Objekt benennt Ziel der Handlung (*Gebefall*): „Dozent | π | überreichen | Teilnahmebestätigung | an-Student“ (vgl. [Lor00], S. 46f)

Ein Beispiel:

[Schulz | π | alphabetisch hinzufügen | neu Anmeldungen | aktuell in-Anmeldeliste]

Als letztes Beispiel für Satzmuster soll an dieser Stelle noch die Modellierung von Aussagen mit einem Ding-Apprädikator vorgestellt werden, in denen kein Ding-Prädikator angegeben ist. Hierzu führt Lorenz den *Leerpädikator* mit dem Symbol *o* ein, der in Aussagen beliebige Prädikatoren ersetzen kann. In der folgenden Aussage bezeichnet „W3045“ einen Kurs:

❏ Natürliche Sprache:	[W3045 ist voll.]	
❏ Normsprache:	[W3045 | ε | voll]	(3.10)
❏ Satzbauplan:	[N | ε | qo]	

Der Leerprädikator kann hier verwendet werden, um den Ding-Prädikator wie in der natürlichsprachlichen Aussage auszusparen, den apprädikativen Charakter eines Objektes aber wiederzugeben. Da der Leerprädikator jedoch – wie der Name schon sagt – keinerlei Intension besitzt, würde die Aussage von dem Kontext abhängig sein, aus dem hervorgeht, dass „W3045“ einen Kurs bezeichnet. Besser daher die Modellierung mit [N | ε | qQ], also „W3045 | ε | voll Kurs“.

3.1.3. Normsprachliche Entwicklung von Software

Auf Basis einer Normsprache mit ihren inhaltlich (fachlich) definierten, normierten Termini können Systemanforderungen in einem Anwendungsgebiet präzise beschrieben werden, ohne spezielle Architekturkonzepte oder Entwicklungsansätze in der Ausprägung von Diagramm- und Beschreibungssprachen berücksichtigen zu müssen. Somit kann zunächst zusammen mit den Anwendern/Fachexperten ein methodenneutraler Fachentwurf der Software erarbeitet werden, der bzgl. seiner Implementierung noch keinen spezifischen softwaretechnischen Lösungsansatz zu erkennen gibt. Dieser erste Fachentwurf kann dann anschließend – ohne Beteiligung der Benutzer – softwaretechnisch geplant (bzw. methodenspezifisch in andere formale Sprachen transformiert) und umgesetzt werden (vgl. [Ort98]). Im Folgenden soll der Ablauf dieses zweigeteilten Fachentwurfs, der in Abbildung 3.3 skizziert ist, erläutert werden.

3.1.3.1. Methodeninvarianter Teil des Fachentwurfs

Der Aufbau eines normsprachlichen Entwicklungssystems für eine Domäne ist iterativ und evolutionär. Mit jedem neuen Softwareprojekt werden das Vokabular und die semantischen Beziehungen weiter ausgebaut. Neben der Aussagensammlung selbst gehört ihre grammatikalische Normierung sowie ihre terminologische Rekonstruktion zum ersten, dem methodenneutralen Teil des Fachentwurfs:

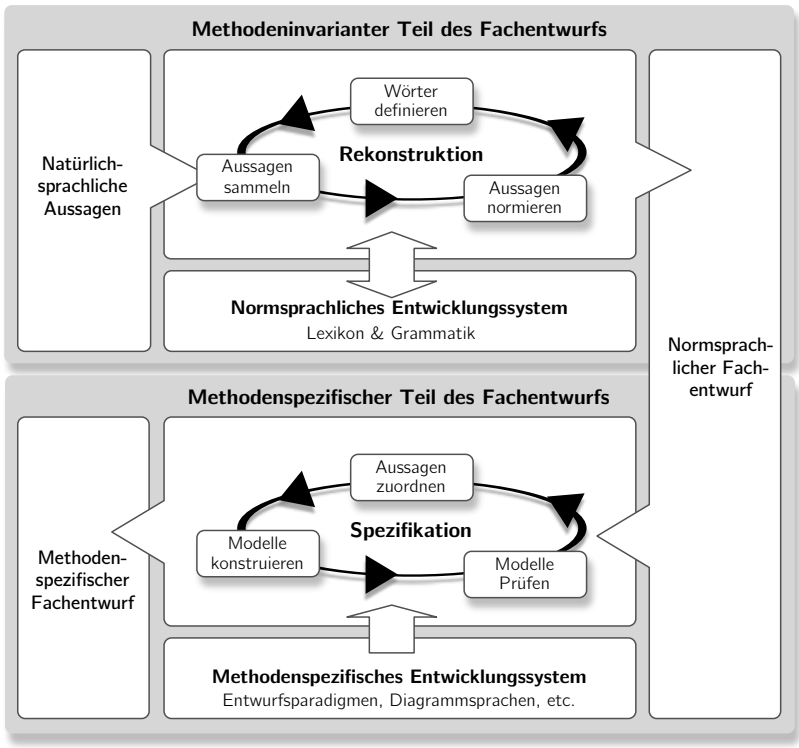

Abbildung 3.3.: Zweigeteilter Fachentwurf auf Basis einer Normsprache (eigene Darstellung).

Aussagennormierung Um syntaktisch normierte, methodenneutrale Aussagen zu erhalten, muss in einem ersten Schritt die Kontextfreiheit jeder Aussage gewährleistet werden. Lehmann schlägt dazu in [Leh98b] die Substitution der Pronomen durch entsprechende Substantive vor. In einem weiteren Schritt sollten komplexe Aussagen in einzelne und mehrdeutige in eindeutige Aussagen überführt werden. Zur Identifikation des ausführenden Objektes sollte man passiv formulierte Aussagen durch aktiv formulierte ersetzen (vgl. ebd.). Nachdem man die Substantive und Verben in eine normierte Form gebracht hat, lassen sich Satzlogik überprüfen und redundante Aussagen entfernen.

Terminologische Rekonstruktion Bei diesem Schritt werden die o. g. Zuordnungsprobleme von Begriffen und Benennungen beseitigt. Dazu können auch neue Begriffe eingeführt werden. Weiterhin werden eindeutige Begriffsbedeutungen und semantische Beziehungen (Generalisierungshierarchien, Kompositionsbeziehungen etc.) zwischen den Begriffen sowie zwischen den Begriffen und Benennungen festgelegt. Lehmann empfiehlt dazu folgendes Vorgehen (vgl. ebd.): Identifizierung aller potenziellen Fachbegriffe und

Normierung ihrer Flexion[16]. Eine Stoppwortliste[17] kann dazu verwendet werden, um semantisch unbedeutende Wörter herauszufiltern. Die verbleibenden Fachbegriffskandidaten werden dann durch explizite Definition auf Basis vorhandener Begriffe rekonstruiert und dem Lexikon hinzugefügt. Eine zentrale Forderung bei der Rekonstruktion ist die Wahrung einer größtmöglichen Kontextunabhängigkeit und die Anwendungsneutralität der definierten Fachwörter: Wörter der Normsprache sollen im Gegensatz zu Wörtern natürlicher Sprache nicht erst im Anwendungskontext eine bestimmte Bedeutung annehmen, sondern als Terminologie immer die selbe festgelegte Semantik besitzen (vgl. [Lor00], S. 39, [Ort95], S. 156).

3.1.3.2. Methodenspezifische Spezifikation auf Basis normsprachlicher Aussagen

Dieser zweite Teil des Fachentwurfes basiert auf den fachlich relevanten, genormten Aussagen und Definitionen des ersten Teils. Diese werden nun schrittweise in methodenspezifische Diagramm- und Beschreibungssprachen transformiert, bevor sie mit einer Programmiersprache umgesetzt werden: Zunächst werden die normsprachlichen Aussagen den relevanten Definitionen anwendungssystemtypischen Architektursichten[18] und den darin verorteten Aspekten[19] zugeordnet (vgl. [Leh98b], [Sch97a], S. 171f.). Der Anwendungstyp wird in vielen Fällen bereits – explizit oder implizit – vorgegeben sein. Sollte dies nicht so sein, so bieten die Problemstellung und die Anforderungen aus dem ersten Teil des Fachentwurfs eine gute Entscheidungsgrundlage zur Abwägung der in Frage kommenden Optionen. Durch diese Strukturierung wird das Modellierungsproblem vereinfacht. Im Zweifelsfall empfiehlt Lehmann eine Mehrfachzuordnung von Aussagen[20].

Um den Aussagen entsprechenden Diagramm- oder formalen Sprachkonstrukten zuordnen zu können, müssen diese klassifiziert werden. Schienmann beschreibt in [Sch97a], S. 171, ein solches Klassifikationsschema mit der Einteilung in Aussagen zu Attributen, Beziehungen, Funktionalität, Berechtigungen, Aktivitäten und Zuständen. So klassifizierte normsprachliche Aussagen können einem entsprechenden Diagramm- oder formalen Sprachkonstrukt zugeordnet und entsprechend syntaktisch normiert und umgesetzt

[16] Flexion beschreibt die Beugung von Wörtern (Konjugation oder Deklination), bspw. durch Veränderung des Wortstammes oder das Anhängen von Wortendungen. Zur Flexion zählen auch Steigerungsformen wie Komparativ und Superlativ. Zwar spielt die Flexion zumeist auf syntaktischer Ebene eine Rolle, Plural und Genus haben jedoch oft auch eine semantische Bedeutung. Das könnte eine Erklärung sein, warum Lehmann ihre Normierung nicht zur grammatikalischen, sondern zur terminologischen Aussagentransformation zählt.

[17] *Stoppwörter* nennt man im *Information-Retrieval* Wörter, die bei einer Volltextindexierung nicht beachtet werden, da sie vor allem syntaktische Funktionen übernehmen und daher keine semantische Bedeutung haben. Geprägt wurde der Begriff durch die Abhandlung von Hans Peter Luhn über die erste Implementierung eines Algorithmus zur Satzextraktion (vgl. [Luh58]).

[18] Die wesentlichen Sichten sind oftmals Statik (*data viel*, *type view* oder *informational aspect*), Funktionalität (*process view*, *mechanism*, *algorithm* oder *functional aspect*) und die Dynamik (*control view*, *time view*, *state view* oder *behavioral view*).

[19] Aspekte sind alle eigenständigen Anforderungen (*concerns*), die funktionale Anforderungen betreffen und sich einfach kapseln lassen, und *System-Level-Concerns*, die technische Randbedingungen darstellen und nicht so einfach gekapselt werden können, da sie viele Teile eines Systems betreffen, wie bspw. das *Logging* oder auch entsprechende fachliche Belange.

[20] Solche miteinander verwobenen Anforderungen werden auch als *crosscutting concerns* bezeichnet, denn sie „schneiden“ quer durch logische Schichten des Systems. Siehe dazu auch S. 143.

werden. Einzelne methodenspezifischen Konstrukte werden abschließend zu vollständigen Diagrammen und Beschreibungen zusammengesetzt und auf Vollständigkeit und Widerspruchsfreiheit überprüft.

3.2. Die Wissensraummetapher als konstruierende Semantik für Lernumgebungen

Die Entwicklung virtueller Lernumgebungen ist ein interdisziplinäres Gebiet, das auf eine Vielzahl von Theorien innerhalb verschiedener Fachbereiche zugreifen kann (vgl. [Blu98], S. 93ff.). Auf den folgenden Seiten soll hergeleitet werden, warum die in dieser Arbeit verwendete Wissensraummetapher als semantisches Bezugssystem einen konstruierenden Charakter für die Domäne des computergestützten Lernen und Arbeitens hat und somit als grundlegende Terminologie zur Entwicklung eingesetzt werden kann. Zunächst werden Theorien der Pädagogik und der Koginitionspsychologie aufgeführt, um die grundlegenden Gestaltungsaspekte von Lernumgebungen zu identifizieren.

Anhand dieser Theorien wird das Wissensraumkonzept schließlich eingeordnet und näher beschrieben.

3.2.1. Lerntheoretische und koginitionspsychologische Grundlagen

Als Leitbild oder Paradigma für die Theorienbildung und Gestaltung von Lernarrangements sind besonders drei fundamentale Orientierungsrichtungen von Relevanz: Der aus dem Reiz-Reaktionslernen hervorgegangene Behaviorismus, der Kognitivismus und der Konstruktivismus (vgl. bspw. [THH+96], S. 42, [BP94], S. 110ff. und [Ste00], S. 818).

3.2.1.1. Behaviorismus

Die Werke des russischen Physiologen Pawlow über das Reiz-Reaktionsverhalten von Hunden (*klassisches Konditionieren*) inspirierten den Amerikaner Watson im Jahr 1913 zu seiner Abhandlung „Psychologie, wie der Behaviorist sie sieht“. Genau wie Pawlow ging Watson davon aus, dass ein konditionierter Reiz durch einen unkonditionierten ersetzt werden kann, wenn beide oft genug zusammen eingesetzt werden. Damit wurden Lernen und Reizsubstitution gleichgesetzt und der Begriff des Behaviorismus als objektive, psychologische Lerntheorie geprägt (vgl. [Kli93], S. 17, [Ede00], S. 67). Etwa zeitgleich mit Pawlow entwickelte der Amerikaner Thorndike – ebenfalls auf Basis von Tierversuchen – das Prinzip der Verstärkungstheorien, wobei der Erfolg einer Reaktion darüber entscheidet, ob die Reaktion in Zukunft häufiger gewählt wird (vgl. [Ede00], S. 65f.). Diese Theorie lieferte einen wesentlichen Beitrag zur behavioristischen Lernauffassung, der auch in der heutigen Zeit zum Teil noch maßgeblich ist[21]. Um 1930 beschrieb Skinner, ein weiterer amerikanischer Wissenschaftler, den Begriff der operanten Konditionierung. Anders als Thorndike wartete Skinner nicht auf zufällige Verhaltensweisen der Probanden,

[21] Beispielsweise fußt der didaktische Ansatz des *Drill&Practice* beim computergestützten Lernen auf dem „Übungsgesetz“ von Thorndike. Dieses Gesetz besagt, dass kurze zeitliche Abfolgen von Reiz-Reaktionsverbindungen das Behalten besonders stärken (vgl. [Kli93], S. 17f.).

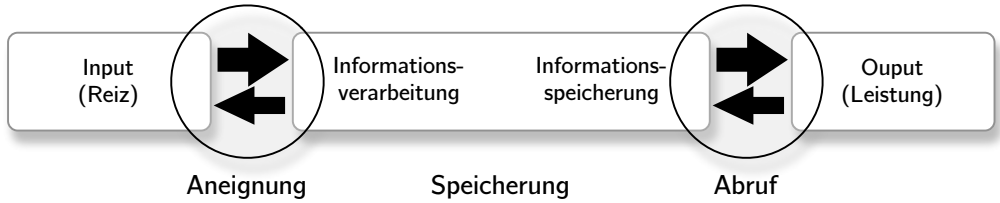

Abbildung 3.4.: Kognitivistisches Grundmodell der menschlichen Informationsverarbeitung (aus: [Ede00], S. 165).

sondern versuchte eine Steuerung, indem er richtige Verhaltensänderungen verstärkt und falsche nicht (vgl. [Ste00], S. 819). Die Arbeiten von Skinner stellen einen wichtigen Bestandteil der behavioristischen Theorie, die vorwiegend dem programmierten Unterricht der 60er Jahre zugrunde gelegt wurden (vgl. ebd.).

Wenngleich der Behaviorismus als Grundlage für die Gestaltung vieler Lernarrangements diente[22], liegt aus heutiger Sicht die Hauptkritik an der fehlenden Bezugnahme auf höherwertige kognitive Lehrziele hinsichtlich des Leistungsniveaus. Es ist zwar notwendig, Faktenwissen und Wissen über Prozeduren oder Prinzipien aus dem Gedächtnis abrufen und verbal wiedergeben zu können. Zur Verbesserung der Problemlösefähigkeiten von Lernenden sind jedoch höhere Lehrziele[23] wie das Verstehen von Informationen, das Anwenden der Regeln und Prinzipien sowie Analyse- und Synthesefähigkeiten notwendig, um korrekte (d. h. im jeweiligen Kontext funktionsfähige) und konsistente Modelle des Problembereichs bilden zu können. Durch behavioristische Ansätze kann lediglich die Wiedergabe von Informationen geprüft werden. Lernprozesse, die kein beobachtbares Verhalten bei dem Lernenden auslösen, können behavioristisch nicht erklärt werden (vgl. [Has95], S. 164). Betrachtet wird einzig die Veränderung des *Inputs* (Stimulation) zum *Output* (Response) als einzige Variable, um einen Lernerfolg zu definieren. Diese Sichtweise eignet sich nicht für Lernprozesse, die komplexere Zusammenhänge als Grundlage haben. Allerdings werden einfach strukturierte Prozesse nach wie vor oftmals behavioristisch konstruiert (vgl. [Ste00], S. 819).

3.2.1.2. Koginitivismus

Im Gegensatz zum Behaviorismus, bei dem äußere Reize ein Individuum steuern, wird im Kognitivismus (lat. cognoscere: *erkennen*) der Lernende als Individuum verstanden, das äußere Reize aktiv und selbständig verarbeitet (vgl. [THH+96], S. 43, [Gül01], S. 81). Der Fokus bei kognitiven Lerntheorien liegt auf den koginitionspsychologischen Annahmen der menschlichen Informationsverarbeitungsfähigkeit, dessen Grundmodell in Abb. 3.4 skizziert ist. In dem Grundmodell werden drei Bereiche angenommen, die

[22] Neben dem Drill&Practice sind als weitere Beispiele der programmierte Unterricht der 60er Jahre zu nennen, ebenso wie Tutorials, Simulatoren und Ansätze des Instructional Designs.

[23] Die Arbeitsgruppe um Bloom formuliert in [BEF+56] ein Schema zur Klassifikation von Lehrzielen auf unterschiedlichen Leistungsniveaus.

jeweils untereinander in einem Kontext stehen und interagieren. In der ersten Phase der Aneignung findet das eigentliche Lernen im engeren Sinne statt. Dabei werden Außenreize vom Individuum aktiv unter Berücksichtigung früherer Erfahrungen wahrgenommen (selektive Wahrnehmung). Diese gefilterten Informationen werden in einem so genannten *Kurzzeitspeicher* zusammen mit vorhandenen Informationen aus dem *Langzeitspeicher* aktiviert[24].

In einer zweiten Phase können in ihrer Bedeutung übereinstimmende oder assoziierte Informationen durch Wiederholung und Ausarbeitung – dies wird durch die dritte Phase *Abruf* repräsentiert – aus dem Kurz- in den Langzeitspeicher übernommen werden[25]. Dadurch entstehen so genannte *mentale Modelle*, also vereinfachte, subjektive Repräsentationen der Realität, die die Grundlage für das Verständnis von Sachverhalten bilden (vgl. [GS83] und [JL83]). Diese individuellen Denkmodelle werden mit zunehmenden Verständnis eines Sachverhaltes elaboriert und angepasst und sind dadurch dynamisch.

Lernen geht in kognitivistischen Lerntheorien daher einher mit der Veränderung kognitiver Strukturen und Prozesse. Demnach ist entscheidend, wie Lernende mit einem Lernangebot umgehen, d. h. welche kognitiven Operationen ausgeführt werden und ob sich diese zu effektivem Lernen, also zum Aufbau von Wissen eignen (vgl. [Ker01], S. 66, [Hes04], S. 17ff.). Die Annahme, dass verschiedene Wissenstypen in unterschiedlichen Subsystemen des Gedächtnisses gespeichert werden und es demnach jeweils andere Prozesse erfordert, um Wissen in dem entsprechenden System dauerhaft zu verankern bzw. abzurufen, ist der Ausgangspunkt der Theorien: „One can communicate verbally one's declarative knowledge, but not one's procedural knowledge. Declarative knowledge seems to be possessed in an all-or-nothing manner whereas procedural knowledge seems to be something that can be partially possessed. One acquires declarative knowledge suddenly by being told whereas one acquires procedural knowledge by performing the skill" [And76], S. 117ff.

Ansatzpunkt didaktischer Überlegungen ist daher vor allem die Klassifikation der Lehr-Lerninhalte, um daran die für die Verarbeitung notwendigen Prozesse abzuleiten. Unter Berücksichtigung des Vorwissens der Lernenden werden dann kognitive Konzepte wie Lernzielkataloge definiert und zur Grundlage instruktionaler Systeme gemacht (vgl. [Sch01b], S. 73).

Die Theorie der Kognition hat zwei wichtige pädagogisch-methodische Konzepte hervorgebracht: Beim *explorativen Lernen* (entdeckenden Lernen) stehen Lernarrangements oder -anwendungen im Zentrum der Betrachtung, die eigenständiges Lernen motivieren sollen, insbesondere einen *definierten* Anregungsgehalt zur Initiierung von selbstgeregelten Lernaktivitäten aufweisen (vgl. [Ker01], S. 219ff.). Statt alle relevanten Informationen vorstrukturiert zu präsentieren zu bekommen, muss der Lernende diese zunächst

[24] Dieses Gedächtnismodell mit einem Kurzzeitgedächtnis als Zwischenspeicher wurde von Atkinson und Shiffrin vorgeschlagen und ist weit verbreitet (vgl. [Ede00], S. 253).

[25] Dieser Anpassungsprozess wurde von Piaget als *kognitive Assimilation* bezeichnet. Sie kommt dann zustande, „wenn ein kognitiv aktiver Organismus eine Erfahrung in die konzeptuelle Struktur einpasst, über die er jeweils verfügt" [Gla94], S. 28. Wird das Ziel nicht erreicht, kann die sich ergebende Störung zu einer Akkomodation führen (vgl. ebd., S. 33).

finden, priorisieren und neu ordnen, bevor er daraus Regeln ableiten und Probleme lösen kann. Somit wird auch ein stärkerer Wert auf *Metakognition* gelegt, also das „Wissen des Lernenden über die eigenen kognitiven Prozesse und deren Bedingungen“ [WKH+93], S. 210. Entdeckendes Lernen ist auch gut mit konstruktivistischen Theorien vereinbar, die im Folgenden noch vorgestellt werden.

Eine spezielle Form des entdeckenden Lernens ist das Lernen mit *Welten* (Simulationen). Hier bewegt sich der Lernende in einer beschränkten Umgebung, in der er bestimmte Gesetze ausprobieren, mit vielfältigen Perspektivwechseln arbeiten und Objekte konstruieren kann (vgl. [Sch01b], S. 50ff.).

Im Zusammenhang mit der Forschung zur Künstlichen Intelligenz ist in den 80er Jahren der Versuch einer Anpassung des Lernangebotes an den aktuellen Wissensstand der Lernenden durch so genannte *Intelligente Tutorielle Systeme* eingeführt worden. Dabei ist es das Ziel, aus Benutzereingaben ein Kompetenzprofil abzuleiten und durch die Gegenüberstellung eines „korrekten“ Modells der Expertise die Kompetenzdefizite zu diagnostizieren. Auf dieser Basis sollte das Lernangebot auf die aktuellen kognitiven Lernprozesse besser angepasst werden als konventionelle CBT-Programme mit fest definierten Lernwegen.

Gegenüber dem Behaviorismus wenden sich kognitivistische Theorien mehr den inneren Vorgängen beim Lernen zu. Die im Rahmen des Kognitivismus entwickelten methodischen Konzepte wie Mikrowelten oder exploratives Lernen lassen sich daher gut mit konstruktivistischen Ansätzen vereinbaren. Andererseits wird dem Kognitivismus unterstellt, einen zu großen Augenmerk auf Methoden der künstlichen Intelligenz zu legen. Ein weiterer Kritikpunkt aus konstruktivistischer Sicht ist sein philosophischer Objektivismus, der aussagt, dass die Welt sich ohne das Subjekt konstruieren lasse. Es gibt im Kognitivismus also keine konstruierte Wahrheit. Wissen wird als etwas aufgefasst, das extern und unabhängig vom Bewusstsein existiert (vgl. [Blu98], S. 113f.). Damit grenzt sich die Theorie hauptsächlich vom Konstruktivismus ab.

3.2.1.3. Partial-Theorien des Lernens

Auf Basis des vorgestellten kognitionspsychologischen Grundmodells haben sich weiterhin einige Partial-Theorien des Lernens entwickelt, die aus ganz unterschiedlichen, mal wahrnehmungs- und motivationspsychologischen, mal informationstheoretischen Richtungen kommen und meist kleinere, isolierte Gestaltungsaspekte von Lernumgebungen betreffen (vgl. [Sch97b], S. 86). Oftmals beziehen sie sich auf die Informationsrepräsentation, wie beispielsweise die Visualisierung, Strukturierung und Granularität.

Ein Beispiel dafür ist die *Dual-coding* Theorie, die auf der Annahme gründet, dass es zwei voneinander getrennte kognitive Kodierungssysteme für die Aufnahme und Verarbeitung verbaler und visueller Informationen gibt. Die Verarbeitung von sprachlichen Informationen verläuft sequentiell, während nicht-verbale Informationen (imaginal, bildlich oder visuell-räumlich) gleichzeitig zur Verarbeitung zur Verfügung stehen. Demnach

ist die Aktivierung der Systeme von der Art des Reizes abhängig; eine doppelte Codierung, die beide Systeme aktiviert, erhöht nach dieser Theorie die Behaltenswahrscheinlichkeit (vgl. [Blu98], S. 98). Die Dual-coding Theorie wird daher häufig zur Begründung eines Multimedia-Einsatzes herbeigezogen (vgl. [Sch97b], S. 88).

Interessant im Zusammenhang mit Hypermedia-Systemen ist die Theorie der *kognitiven Plausibilität*, die davon ausgeht, dass in Analogie zu den vernetzten Strukturen mentaler Modelle direkt auf die Lernförderlichkeit von Hypermedia geschlossen werden kann, also die Strukturgleichheit von Text und Denken koginitiven Lernerfolg impliziert. Blumstengel macht dazu in [Blu98], S. 103, auf die starke Vereinfachung dieser Theorie aufmerksam: Hypermedia-Netzwerke liegen zwar nicht linear vor, werden aber in jedem Fall linear gelesen. Ein positiver Effekt ist in diesem Zusammenhang aber in jedem Fall die Möglichkeit, aufgrund der vernetzten Struktur unterschiedliche Lernwege anbieten zu können (vgl. ebd.).

3.2.1.4. Konstruktivismus

Ähnlich wie der Kognitivismus, betrachten konstruktivistische Lerntheorien den internen Prozess des Verstehens und Verarbeitens von Informationen. Jedoch räumt der Konstruktivismus der individuellen Wahrnehmung, Interpretation und Konstruktion innerhalb des Verarbeitungsprozesses eine deutlich höhere Bedeutung zu als der Kognitivismus. Hiernach werden beim Lernen koginitive Strukturen modifiziert, indem *individuelle* Konstrukte aufgebaut, verknüpft, reorganisiert und modifiziert werden (vgl. [Kli93], S. 134).

Wenngleich der Kognitivismus und der Konstruktivismus grundsätzlich die gleiche Sichtweise vom aktiv lernenden Individuum vertreten, unterscheiden sie sich fundamental in der Sichtweise der Instruktion: Die Auffassung, dass Wissen nur ein individuelles Konstrukt ist, das nicht extern bestimmt werden kann, macht die Instruktion als „Vermittlung von Wissen“ streng genommen unmöglich. Danach dürfte ausschließlich selbstgesteuertes, kollektives Lernen erlaubt sein (radikaler Konstruktivismus, vgl. [THH+96], S. 47).

Mittlerweile hat sich eine gemäßigte Zwischenposition formuliert, die „[...] vom Konstruktivismus die Einsicht in die Bedeutung von handelndem Lernen in komplexen Situationen und Problemräumen [...]“ übernimmt (vgl. [Wei93], S. 13). Gleichzeitig wird aber angenommen, dass Lernende hierfür adäquate, individuelle mentale Modelle oder andere elaborierte Strukturen brauchen, deren Erwerb sich durch Instruktion – also der expliziten Darstellung und Organisation des benötigten Wissens – erleichtern lässt. Gemäßigte Vorstellungen erlauben daher die Instruktion, um den individuellen Konstruktionsprozess anzuregen und zu unterstützen[26]. Der Lehrende ist dabei für die Aktivierung eines natürlichen Lernprozesses verantwortlich und fördert im Weiteren die Metakoginition und Multiperspektivität (vgl. [Kli93], S. 253).

[26] Obwohl dieser so genannte gemäßigte Konstruktivismus sicherlich eine geeignete Basis für das Erlangen von prozeduralem Wissen innerhalb eines universitären, allgemeiner gesagt instruierenden Umfelds darstellt, spiegelt aber gerade die radikale Sichtweise häufig die Situation wider, die ein Lernender beim lebenslangen Lernen vorfindet: Alleinorganisierend und ohne Instruktion einer Lehrkraft.

Unter der Annahme, dass das Erkennen nicht nur in einem Kontext stattfindet, sondern auch mit den Elementen des Kontextes, betonen Theorien des situierten Lernens (*situated cognition*) die soziale Eingebundenheit von Lernprozessen. Demnach sollen die in konstruktivistisch gestalteten Lernsituationen formulierten Aufgaben in Anwendungssituationen eingebettet werden, die reale Arbeits- und Lebenssituationen widerspiegeln. Dies soll eine konkrete Assoziation und Transformation des Gelernten in reale Problemstellungen und somit auch in eine soziale Umwelt ermöglichen (vgl. [Ede00], S. 287). Betont werden dabei insbesondere soziale Aspekte des Lernens, das sich im kommunikativen Austausch zwischen Experten und Lerner vollzieht: „Wissen kann nicht *übermittelt* werden, sondern es wird in Interaktionen zwischen Experten und Lernenden – in authentischen Situationen – jeweils neu *konstruiert* und *ausgehandelt*“ [Ker01], S. 80.

Daraus lassen sich drei Ansätze für die Differenzierung alternativer Lernumgebungen und für ihren Entwurf und Entwicklung ableiten (vgl. [Sch97b], S. 81):

1. Das Lernen als Lehrlingsverhältnis (*cognitive apprenticeship*), wobei die Expertenrolle entweder durch das System oder durch Domänenspezialisten eingenommen werden. Im Kontext eines authentischen Problems wird zunächst (eine modellhafte) Vorstellung der Vorgehensweise gezeigt, die ein Experte anwenden würde. Anschließend hat der Lernende durch Beobachtung und Nachahmung ähnliche konkrete Probleme zu lösen. Das System unterstützt durch Rückmeldungen und Hilfestellungen, die – abhängig vom Wissensstand des Lernenden – schrittweise ausgeblendet werden können.

2. Das Lernen als kommunikatives Handeln in Wissensgemeinschaften (*communities of practice*, CoPs): „A CoP defines itself along three dimensions: its joint enterprise as understood and continually renegotiated by its members, the relationships of mutual engagement that bind members together into a social entity, the shared repertoire of communal ressources (routines, sensibilities, artefacts, vocabulary, styles, etc.) that members have developed over time“ [Wen98a], S. 2. Hier geht es darum, selbstorganisiertes und -gesteuertes Lernen in einem sozialen Umfeld zu ermöglichen, aber Lerngruppen im Sinne der Community of Practice nicht von außen zu organisieren (vgl. [McD99]). Nach Probst muss in diesem Sinne ein Kontext für selbstorganisierte Gruppen geschaffen werden, d. h. kulturelle Voraussetzungen zu garantieren, Zeit zur Verfügung zu stellen, Kontakte zu fördern, Selbstreferenz und Reflektion anzuerkennen, spezifische Geschichten und Sprachspiele zu tolerieren (vgl. [Pro87]).

3. Das Lernen mit kognitionsfördernden Werkzeugen (*cognitive tools*). Dies sind interaktive multimediale Softwareanwendungen, welche eine direkte Manipulation von Objekten zulassen, die bestimmte Sachverhalte repräsentieren [Epp01]. Man kann sie bspw. kopieren, duplizieren oder transformieren, wodurch der kognitive Verarbeitungsprozess verstärkt, erweitert oder unterstützt wird. Anhand des spielerischen oder explorativen Umgangs mit den Werkzeugen wird der Lernende bei der Konstruktion der eigenen gedanklichen Konzepte unterstützt. Zur Klasse der kognitiven Werkzeuge zählen beispielsweise Kommunikationswerkzeuge, *Shared Applications* und *Workspaces*, die eine gemeinsame synchrone und asynchrone

Bearbeitung von Dokumenten ermöglichen, so genannte *Conceptmanager* bspw. zum Erstellen von MindMaps sowie „Pädagogische Agenten“, die – konfigurierbar – beim Lernen anleiten und zum Beispiel durch Informationsrecherche unterstützen können. Eine umfangreiche Beispielsammlung von kognitionsfördernden Werkzeugen findet sich bei Schulmeister in [Sch97b], S. 341ff. Prinzipiell kann jedoch *jedes* Mediensystem, das neben der Wiedergabe *auch* die Bearbeitung und Speicherung von Informationen ermöglicht, als kognitionsförderndes Werkzeug für Lernaktivitäten angesehen werden, *wenn* der Lernprozess an sich dadurch gefördert wird (vgl. [Ker01], S. 247). Einfachste Unterstützungsfunktionen sind bspw. das Unterstreichen oder Hervorheben von Textstellen, Annotationen an Objekten oder das Verknüpfen von Objekten durch Zeiger.

3.2.1.5. Ein strukturgenetischer Wissensbegriff

Sowohl die kognitivistische als auch die konstruktivistische Auffassung des Lernprozesses geht von einer Veränderung kognitiver Strukturen beim Lernenden aus. Erstere vernachlässigt jedoch die internen Verarbeitungprozeduren, wohingegen aus konstruktivistischer Sicht die Möglichkeit der Wissensweitergabe und die Wissensaufnahme, also ein Wissenstransfer, nicht Bestandteil der Theorie ist. Die postulierte Verankerung der Wahrnehmung in individuelle, situative und soziale Kontexte lässt jedoch einen Wissensbegriff notwendig erscheinen, der auf einer erweiterten Grundlage basiert. Er muss über die Personengebundenheit hinaus eine Objektivierung von Wissen umfassen, damit es öffentlich gemacht und Kommunikation zwischen Menschen daran ausgerichtet werden kann. Um dieses erklären, müssen demnach zwei übergeordnete Formen von Wissen unterschieden werden: (1) Personengebundenes Wissen, das auf dynamischen kognitiven Strukturen eines Individuums beruht und nur individuell zugänglich ist, und (2) öffentliches Wissen, das durch vereinbarte Regeln systematisch verbalisiert und explizit artikuliert werden kann.

Der ungarische Wissenschaftler Polanyi führte diesbzgl. in [Pol66] erstmalig den Begriff *tacit knowing* ein, aus dem sich in anschließenden Diskussionen der Term *tacit knowledge* (implizites Wissen) entwickelte und der eine Art von Wissen beschreibt, dass sehr schwierig zu artikulieren, sehr spezifisch und außerdem von persönlichen Fähigkeiten und Erfahrungen eines Individuums abhängig ist: „We know more than we can tell“ [Pol66], S. 4. Beispiele für implizites Wissen sind Eindrücke über die Kultur einer Lerngruppe oder das „Bauchgefühl“ über den Umfang einer gestellten Aufgabe.

Der komplementäre Part, das explizite Wissen (*explicit knowledge*, vgl. [Pol85]) ist leichter kommunizierbar und kann verbal oder mittels Symbolsystemen weitergegeben werden, z. B. in Form von Daten, Informationen oder Dokumenten. Beispiele sind die in einer Vorlesung vermittelten Methoden und Formeln oder die zu erreichenden Punkte einer vollständig gelösten Übungsaufgabe. Auch eine Normsprache fällt in diese Klassifikation[27].

[27] Wobei man bei diesem Beispiel expliziten Wissens noch einmal zwischen dem *kollektiven Wissen* um die Bedeutung der Begriffe einer Normsprache und dem *formalisierten Wissen* um die per Aussagenlogik definierten semantischen Beziehungen und Regeln des Vokabulars unterscheiden kann.

Diese Klassifikation von Wissen hat sich in der Wissenschaft zur Erklärung der Generierung neuen Wissens durchgesetzt (vgl. [Non91], [NT97], [Sve98], S. 98, [Sch00a], S. 30 und [Brä03], S. 29). Nicht zuletzt dadurch, dass ein solches strukturgenetisches Wissensverständnis ein hohes Integrationspotenzial aufweist zwischen Lerntheorien und Theorien des Wissensmanagements (vgl. [SR01]): Explizites Wissen kann zum Gegenstand von Planung, Steuerung und Kontrolle werden und somit den Forderungen eines wirtschaftlich orientierten Managements genügen (bspw. [NT97] oder [PRR03]). Darüber hinaus ist es ebenfalls kompatibel zur Auffassung des Lernens als kognitionsverändernden Prozess, der Grundlage des Kognitivismus und Konstruktivismus ist.

Nonaka und Takeuchi beschreiben in [NT97] auf diesem Wissensverständnis aufbauend eine Theorie, wie persönliche Kenntnisse geteilt und in einen sozialen Kontext[28] eingebracht werden können, um schließlich neues verfügbares Wissen zu produzieren, aus dem wiederum jeder Einzelne individuelles Wissen ableiten kann. Diese Theorie soll an dieser Stelle kurz wiedergegeben werden, da sie u. a. Aufschluss darüber geben kann, wie Lernen in selbstorganisierten Gruppen stattfinden kann und konkrete Handlungsfelder aufzeigt, auf denen eine virtuelle Umgebung diese Prozesse unterstützen kann[29].
Ihr Modell besteht in der Hauptsache aus vier Formen der Wissensumwandlung zwischen implizitem und explizitem Wissen, nämlich Sozialisation, Externalisierung, Kombination und Internalisierung:

Sozialisation Wandlung von implizitem zu implizitem Wissen auf nonverbaler Ebene durch Beobachtung und Nachahmung. Hierdurch können mentale Modelle und Fertigkeiten entstehen. Dies geht vor allem durch den Aufbau physischer Nähe und direkter Interaktionen, wodurch ein *Interaktionsfeld* aufgebaut wird (vgl. [SM02], S. 129f., [NT97], S. 85).

Externalisierung Wandlung von implizitem zu explizitem Wissen durch die Objektivierung impliziten Wissens mithilfe von Modellen, Metaphern, Analogien und Hypothesen. Dadurch wird bisher nicht artikulierbares Wissen soweit aufbereitet, dass es kommuniziert und weitergegeben werden kann. Dieser Prozess wird optimal durch Dialog, Interaktion und kollektiver Reflexion vorangetrieben, indem die Beteiligten bspw. versuchen, ein gemeinsames Konzept zu konstruieren. Insbesondere bei Konzepten, bei deren Entwicklung Deduktion und Induktion miteinander verknüpft werden (vgl. [NT97], S. 77ff.).

Kombination Wandlung von explizitem zu explizitem Wissen auf Ebene der Daten, Informationen und Dokumente. Durch Neustrukturierung und Verknüpfungen dieser Informationen untereinander entsteht aktualisiertes oder neues explizites Wissen, bspw. durch Übertragung von Erkenntnissen auf ein anderes Fachgebiet.

[28] Nonaka und Takeushi erklären ihre Theorie im sozialen Kontext einer Organisation. Sie kann jedoch ebenso gut auf Lerngruppen respektive Wissensgemeinschaften im Allgemeinen bezogen werden. Aktuelle lerntheoretische Ansätze wie die *verteilte Kognition* teilen mit der Theorie von Nonaka und Takeushi den Gedanken, dass das Wissen in einer Lerngruppe nicht in identischer Form bei allen Lernern vorliegt, sondern auf alle Lerner einer (nicht disjunkten) Gruppe verteilt ist (vgl. [KM07]).

[29] Die Möglichkeiten und Grenzen von Interventionen auf solch fragile soziale Konstrukte wie selbstorganisierte Wissensgemeinschaften zeigt Romhard mithilfe dieses Modells in [Rom02], S. 53ff.

Internalisierung Wandlung von explizitem zu implizitem Wissen durch Lernen und dem Machen eigener Erfahrungen. Diese Transformation ähnelt der Sozialisation, nur dass die externen Informationen in einer konkreten, artikulierten und expliziten Form vorliegen. Durch einen Austausch innerhalb sozialer Kontexte wie Lerngruppen oder ganze Unternehmungen kann hierdurch kollektives Wissen geschaffen werden, basierend auf gemeinsamen mentalen Modellen und gemeinsames Know-how.

Das Wissen wird also zunächst jeweils individuell erzeugt, kann aber zur Nutzbarmachung mithilfe geeigneter Mechanismen explizit gemacht und ausgetauscht werden. Die vier Formen der Wissensumwandlung ergeben in ihrer Kombination den Effekt einer Spirale (vgl. Abb. 3.5), welche die Dynamik der Evolution einer Wissensgemeinschaft hinsichtlich ihrer Gemeinschaftskultur und ihres kollektiven Wissens beschreibt.

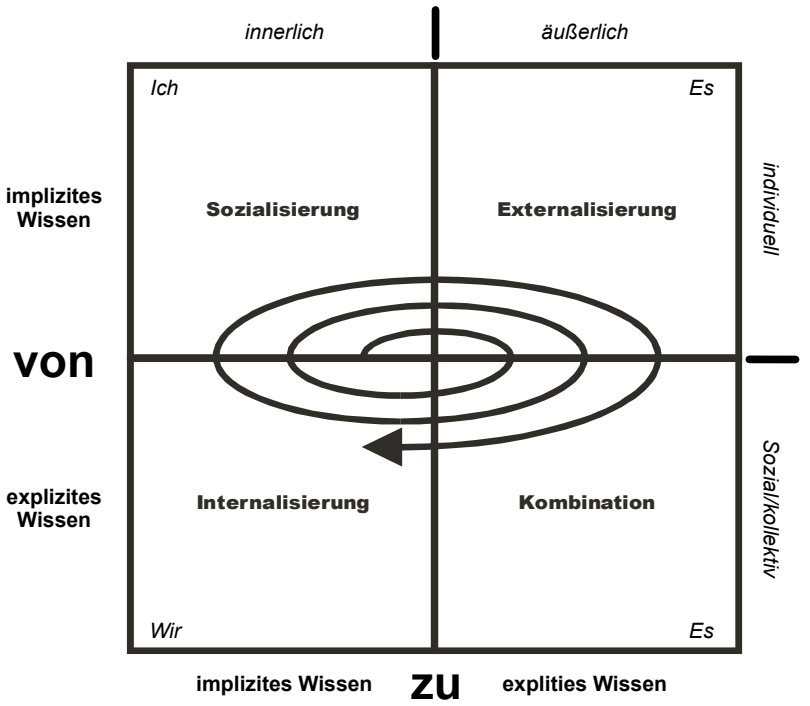

Abbildung 3.5.: Vier Formen der Wissensumwandlung (eigene Darstellung in Anlehnung an [NT97], S. 84 und [Rom02], S. 64).

3.2.2. Wissensräume als konstruierender Theoriebestandteil

„Knowledge needs a physical context if its to be created [...]“ [NTK01], S. 22. Nonaka et al. begründen damit das Konzept des *ba*[30]. Wörtlich übersetzt bedeutet „ba“ Ort; dieser Ort ist jedoch nicht nur im physischen, sondern ebenso im virtuellen und mentalen Sinne gemeint. Ein Ort im physischen Sinn kann ganz klassisch als real existierender Raum angesehen werden. Dieser wird auf virtueller Ebene ergänzt durch bspw. virtuelle Kontakte, Nachrichten und Mitteilungen oder Dokumenten und auf mentaler Ebene zum Beispiel durch geteilte Erfahrungen und gemeinsame Ideen. Das *ba* ist dabei in stetigem Wandel: Es ist immer offen und kann durch die Beteiligten nach Belieben betreten und verlassen werden, was ihm einen Ad-hoc-Charakter verleiht (vgl. [Ren03], S. 100f.). Im Zentrum des Konzeptes steht dabei nicht nur der Austausch von Wissen, sondern vor allem die Generierung von Wissen. Implizites Wissen kann durch Externalisierung in gemeinsame Handlungsräume eingebracht und dort kombiniert werden. Von diesem Gruppenwissen kann durch Internalisierung neues individuelles Wissen erzeugt werden (vgl. Abb. 3.5). Zusammengefasst stellt das *ba* also einen Bezugsrahmen bereit, auf den sich die Beteiligten beziehen können und innerhalb dessen sie durch Interaktion Wissen entwickeln und generieren können.

Virtuelle Wissensräume implementieren diesen Bezugsrahmen. Sie schaffen eine Umgebung, die durch die Beteiligten gestaltet und weiterentwickelt werden kann, ebenso wie das in ihr enthaltene Wissen. Die Nutzung der Raum-Metapher in computergestützten kooperativen Arbeits- und Lernumgebungen ist häufig intuitiv verständlich, da sich virtuelle Räume durch dieselben Eigenschaften auszeichnen wie reale (vgl. hierzu [GR98], [PSBWW99] und [Ham02]):

- Ein virtueller Raum ist persistent und hat feste Grenzen
- Der virtuelle Raum dient sowohl als Strukturierungsinstrument für die in ihm enthaltenen Objekte, wie z. B. Dokumente oder Personen, als auch als Ort der Arbeit mit diesen Objekten (*Handlungsraum*)
- Virtuelle Räume sind aufgrund ihrer Strukturierbarkeit an persönliche oder gruppenspezifische Vorlieben und verschiedenste Arbeitssituationen anpassbar
- Innerhalb eines virtuellen Raums ist sich eine Person allen anderen Mitbenutzern in diesem Raum bewusst und kann etwas über die Aktivität dieser Personen erfahren bzw. mit ihnen in Verbindung treten (*Awareness*)
- Der virtuelle Raum bietet sowohl die Möglichkeit, den Zugang zum Raum selbst als auch den Zugang zu den im Raum enthaltenen Objekte einzuschränken (vgl. hierzu insbesondere [Ham04]).

Im Folgenden sollen einzelne Aspekte virtueller Räume – Strukturierung, kognitive Unterstützung und soziale Aspekte – näher erklärt werden.

[30] Seinen Ursprung hat das Konzept des „ba“ in der japanischen Philosophie und ist nach einigen Weiterentwicklungen Grundlage einiger Arbeiten zum Wissensmanagement geworden (neben [NTK01] vgl. bspw. [KIN00], S. 176ff. und [Ren03], S. 100ff.).

3.2.2.1. Semantische Strukturierung von Räumen und Informationen

Räumliche Anordnung gehört zu den mächtigsten Interventionsfeldern der Wissensorganisation: Der Raum kanalisiert Kommunikation und ermöglicht oder beschränkt die Strukturierung, Verteilung und Vermittlung von Informationen (vgl. [Roe02], S. 95). Somit ist er in der Lage, verschiedene Arbeits- und Lernkontexte herzustellen, indem unterschiedliche Sichten auf Informationen ausgestaltet werden (vgl. [Zac99]). Darüber hinaus dienen geeignete semantische Informationsstrukturen auf lange Sicht auch dem Erinnern (vgl. [Ham02], S. 26).

Die Persistenzschichten raumbasierter CSCW/L-Anwendungen verwalten daher i. d. R. nicht nur die Inhalte (Content) an sich, sondern müssen auch objektrelationale Strukturen abbilden und Funktionen zu ihrer Verwaltung. Persistente Strukturen lassen sich sowohl über Räume als auch über Informationen respektive Informationsobjekte (*knowledge units*) bilden: „The repository structure also includes the schemes for linking and cross-referencing knowledge units. These links may represent concepual associations, ordered sequences, causality or other relationships depending on the type of knowledge being stored." [Zac99], S. 46.

Wie reale Räume auch, können ihre virtuellen Pedanten Türen zu benachbarten Räumen oder Unterräumen haben. Anders als in der Realität, können in einer virtuellen Umgebung sogar Türen eingesetzt werden, die als Hyperlink auf entferntere Räume verweisen. Benutzer können sich über diese Verknüpfungen – entsprechende Zutrittsrechte vorausgesetzt – in diesen Raumstrukturen bewegen. Somit können virtuelle Räume grundlegend frei entlang einer semantischen Struktur oder eines didaktischen Konzepts miteinander verbunden werden. Diese Verknüpfbarkeit virtueller Räume bildet ein zentrales Gestaltungselement für Lern- und Arbeitsumgebungen, da durch sie raumübergreifende, kontextbezogene semantische Aufbau- und organisatorische Ablaufstrukturen nachgebildet bzw. konstruiert werden können, die für die Arbeit und das Lernen von Einzelnen und Gruppen gleichermaßen von Nutzen sind.

Abbildung 3.6 veranschaulicht dieses am Beispiel individueller Sichten für Student und Dozent: Über hierarchische Raumkonstrukte werden zwei unterschiedliche Makrostrukturen konstruiert, die über Hyperlinks auf verschiedene Einstiegspunkte in die Mikrostruktur (hier: Einfacher Raumkomplex einer Lehrveranstaltung) verweisen. Die Makrostrukturen werden in diesem Fall durch das System selbst oder über einen Administrator zentral verwaltet; die Kursraumstruktur kann durch den Dozenten selbst frei gestaltet werden. Virtuelle Räume können somit also eine semantische, logische und zeitliche Gruppierung einer Anzahl von Objekten, Personen und strukturellen Beziehungen darstellen (vgl. [RHS05]).

Den Aufbau virtueller Raumkomplexe kann der Benutzer selbst vornehmen, indem er Unterräume oder Türen einfügt; er kann allerdings auch durch einen Administrator übernommen oder nach einem vom System vorgegebenen „Bauplan" automatisch vorgenommen werden. Darüber lassen sich – wie im Beispiel gezeigt – Anordnungen virtueller Räume beschreiben, also auch bestimmte Sichten auf verschiedene Wissensorganisationen vorstrukturieren. In der Benutzungsoberfläche kann die Wahrnehmung der Benutzer dann dahingehend gesteuert werden, dass diese in einer derartig aufbereiteten Sicht die

Aufbau- und Ablaufstrukturen von didaktischen Szenarien bis hin zu vollständigen Lehrveranstaltungsformen problemlos erkennen können[31].

Die didaktische Spannweite dieses Ansatzes ist weitreichend und umfasst theoretisch die gesamte Skala der Selbstkontrolle: Je nach rechte- oder sichtenspezifischer Einschränkung können somit verschiedenste Szenarien umgesetzt werden, bspw. ein frei gestaltbares Jour Fixe-Konzept (vgl. [HKSE03]), eine ablauforganisatorisch strikt vordefinierte erwägungsdidaktische Pyramidendiskussion (vgl. [HH05]) oder einfach nur die Bereitstellung von Materialien einer Lehrveranstaltung zum Herunterladen.

Hyperstrukturen können nicht nur Räume, sondern auch Informationsobjekte miteinander auf verschiedene Art und Weise in Beziehung zueinander setzen. So genannte Links (*shadow objects*) bieten Verknüpfungen zu einzelnen Objekten und Dokumenten, unabhängig ihrer umgebenden Raumstruktur. Dadurch wird eine mehrfache logische Präsenz von Informationen trotz einer (nicht zwangsläufigen) physikalischen Einmaligkeit ermöglicht, wie sie für Redundanz von Informationen bzw. Wissen in unterschiedlichen semantischen und didaktischen Kontexten notwendig ist (vgl. hierzu [NT97], S. 188ff.). Eine zweite Möglichkeit der Referenzierung von Objekten kann auch durch dynamische Attribut-Zeiger erfolgen, sowohl ein- als auch mehrdimensional[32].

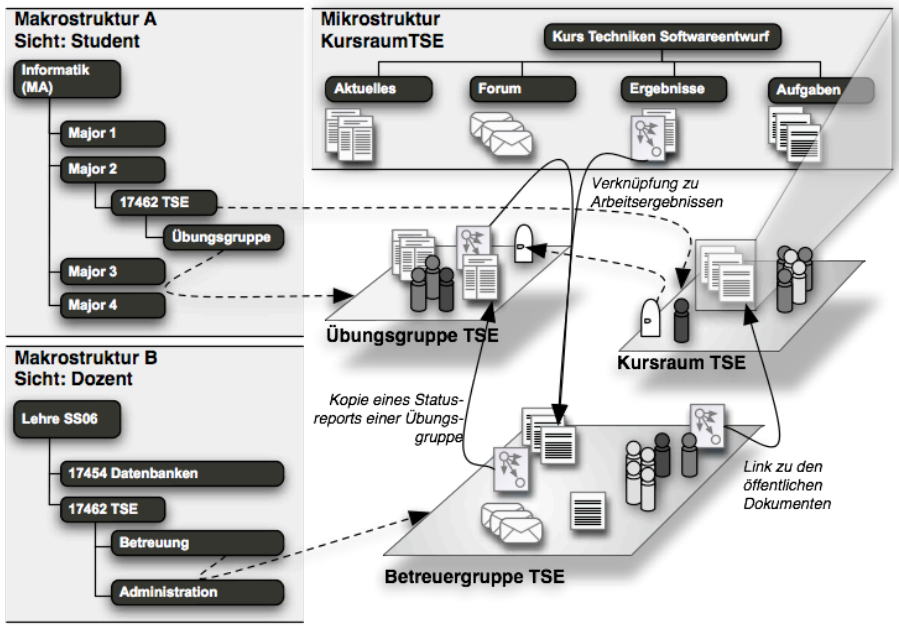

Abbildung 3.6.: Wissensraumstrukturen

[31] Abhängig von der Implementierung der Benutzungsschnittstelle kann die Raum-Metapher dabei auch in der grafischen Oberfläche der Komponenten zum Ausdruck kommen, sie muss es aber nicht.

[32] Im mehrdimensionalen Fall – also mehrere Referenzen pro Objekt-Attribut – lassen sich entsprechende Konstrukte wie Queues, Stacks oder auch einfache Arrays als Attributwert anwenden.

3.2.2.2. Generalisierung von Informationsträgern

Durch eine Generalisierung als Informationsobjekt[33] können Informationen verschiedenster Quellen und Medienformate in den virtuellen Handlungs- und Wahrnehmungsraum der Teilnehmer eingebunden und miteinander verknüpft werden. Ein Informationsobjekt kann also ein Textdokument, ein Videofilm, ein Forenbeitrag oder auch der Abschnitt eines Lernweges sein, dessen Lernobjekte in externen Lernumgebungen vorgehalten werden[34].

Es ergeben sich somit Potenziale hinsichtlich der Schaffung von Wissen, da über diese Generalisierung Informationen unabhängig ihrer Herkunft und ihres Formats semantisch kombiniert und (re-)strukturiert werden können. Auf individueller Ebene wird die Externalisierung mentaler Modelle unterstützt, auf kooperativer Ebene die Wissenstransformation „Kombination“ (vgl. Abb. 3.7).

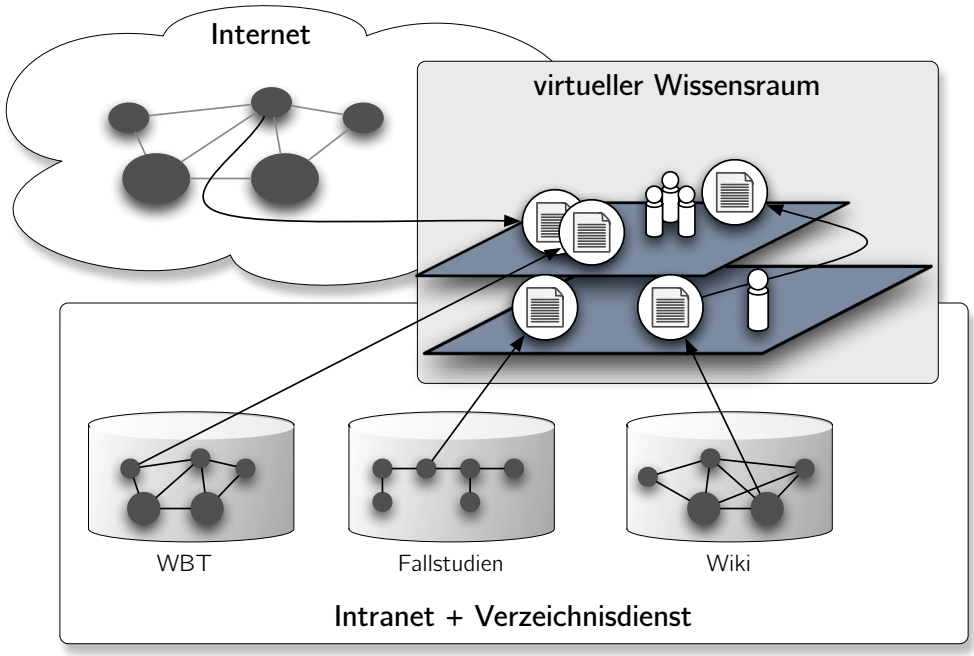

Abbildung 3.7.: Durch Generalisierung können Informationen unabhängig von Herkunft und Format zur Externalisierung mentaler Konzepte und zur Kombination expliziten Wissens in Gruppen genutzt werden.

[33] Vgl. hierzu die Definition von Zack, die in dieser Arbeit bereits auf S. 61 angeführt wurde.

[34] Lernobjekte werden so aus vorgegebenen Lernnetzen und Aggregationsebenen herausgelöst und in virtuellen Räumen zu einem Objekt der Konstruktion. Eine solche Integration externer Lernplattformen in den virtuellen Wissensraum wird in [RHS05] beschrieben.

3.2.2.3. Medienfunktionen und Rechtemanagement

Virtuelle Räume dienen also – wie dargelegt – als Instrument, um Wissen zu strukturieren und individuelle und kollektive Lernprozesse durch die Schaffung semantischer Kontexte zu fördern. Daneben haben sie aber auch noch eine zweite wichtige Funktion (vgl. [PSBWW99], S. 108, [Ham02], S. S. 41ff.): Sie bieten als „Ort der Arbeit" einen Handlungsraum (*activity space*) für die direkte Manipulation von Objekten (vgl. [Shn83])[35]. Dadurch können Eigenaktivität und konstruktives, selbstbestimmtes Vorgehen unterstützt oder Kooperation und Wissenstausch ermöglicht werden, was interne Verarbeitungsprozesse nicht nur beim Einzelnen, sondern auch in der Gruppe begünstigt (*verteilte Kognition*, vgl. [Wes01], S. 196, [Ker01], S. 166, [Hes04], S. 16, [KM07]). Beispielsweise können Lerner gemeinsam eine explizite Repräsentation des verteilten Wissens einer Gruppe in Form von Diagrammen oder Begriffsgraphen aufbauen. Dazu müssen sie technisch in die Lage versetzt werden, eigene Objekte zu erzeugen, Verknüpfungen zwischen Objekten zu erstellen, diese zu arrangieren und zu transportieren. Die Handlungen der Lernenden an und mit Objekten müssen zudem synchronisierbar sein.

Hampel beschreibt dazu in [Ham02], S. 41ff., eine Reihe primärer Medienfunktionen auf Objekten, die – entweder individuell oder kooperativ ausgeführt – zugunsten interner Verarbeitungsprozesse (kognitive Operationen) wirken können:

Erzeugen Möglichkeiten des Entwurfs von Zeichen, wie bspw. Schreiben, Anfertigen von Skizzen und Konstruieren von Modellen. Wichtig hierbei ist die dauerhafte Persistenz der erstellten Objekte im Handlungsraum für die Zeit des Lernprozesses.

Löschen Erzeugte Objekte müssen wieder gezielt aus dem Wahrnehmungsfeld des Benutzers entfernt werden können, bspw. bei veralteten oder redundanten Informationen.

Arrangieren bedeutet das „In-Beziehung-Setzen" verschiedener Objekte miteinander, um Zusammenhänge im Wahrnehmungsfeld aufzunehmen. Objekte können dazu räumlich gruppiert werden.

Verknüpfen Objekte können auch – unabhängig ihres räumlichen Kontextes – auch über Attribute (z. B. gemeinsame *Tags*) miteinander verknüpft werden.

Übertragen Aktiver Austausch von Objekten zwischen Benutzern durch direkte Weitergabe.

Zugreifen Zugriff auf Materialien ohne Einwirkung eines Bereitstellers, bspw. auf ein in einem Raum abgelegtes Objekt.

Synchronisation Abgleichen bzw. Aktualisieren von Darstellungen auf Räume und Objekte, um Rückmeldungen über Handlungen an Objekten und gegenseitige Wahrnehmung von Kooperationspartnern zu erlangen.

[35] Konstruktivisten knüpfen hohe Erwartungen an Methoden der direkten Manipulation von Objekten, da sie dem Paradigma folgen, Lernen nicht vom Tun zu trennen (vgl. [Sch97b], S. 341).

Ein für die Restrukturierung von Informationsträgern besonders wichtiges Werkzeug ist eine Zwischenablage, im Kontext virtueller Wissensräume auch manchmal als *Rucksack* paraphrasiert. Objekte können – entsprechende Zugriffsrechte vorausgesetzt – als Original, Kopie oder Verknüpfung von einer Umgebung in einen anderen bzw. einen weiteren Kontext übertragen werden, indem sie in die Zwischenablage aufgenommen und am Zielort abgelegt werden. Auf diese Art können Objekte bspw. von individuellen Wissensräumen in kooperative eingebracht werden oder vice versa. Eine Möglichkeit zur Unterstrukturierung dieses persistenten Zwischenspeichers versetzt den Benutzer in die Lage, Informationen bereits im Moment der Aufnahme zu kontextualisieren. Diese Informationen können dann in andere Räume – als einzelnes Objekt oder als Struktur – übertragen und somit in individuelle oder kooperative Lernszenarien (wieder) verwendet werden (vgl. hierzu auch Abb. 6.1).

Der Zugriff auf virtuelle Räume und die in ihnen gespeicherten Objekte muss gezielt eingeschränkt werden können. Ein übliches Vorgehen ist das Arbeiten mit Benutzergruppen und Rollen, an die bestimmte Rechte gebunden werden. Auf diese Weise können für eine Gruppe von Nutzern bzw. Lernenden abgestufte Zugriffsrechte auf virtuelle Lernräume und darin enthaltene Materialien eindeutig definiert und gegenüber anderen Nutzern abgegrenzt werden, so dass ein hochflexibles Zugriffsrechtesystem umgesetzt werden kann. Dabei lassen sich die Rechtestrukturen zentralen Kategorien zuordnen (vgl. [Ham04]):

- ➢ Vererbung von Rechten durch eine gegebene Gruppenstruktur
- ➢ Transfer (Delegation) des Rechts, explizit neue Rechte setzen zu dürfen
- ➢ Vererbung und Ableitung von Rechten durch a) die Umgebung oder b) einem fixen Referenzobjekt.

Damit sind die Grundelemente zur Einschränkung virtueller Wissensräumen gegeben, mit denen rollenspezifische Sichten auf Lernszenarien implementiert werden können.

3.2.2.4. Gegenseitige Wahrnehmung

Virtuelle Räume sollten mit Mechanismen ausgestattet sein, die es den Lernenden ermöglichen, Anwesenheit und Aktivitäten anderer Lernender wahrzunehmen und mit ihnen in Kontakt zu treten. Neben der Anwesenheit anderer Nutzer im Raum ist eine zeitnahe Visualisierung von Handlungen an Räumen und Materialien wichtig. Die Auswirkungen des Erstellens, Löschens von Räumen sowie des Verschiebens, Kopierens, räumlichen Arrangierens und Löschens von Objekten sind in kooperativen Szenarien schnell für alle Anwesenden anzupassen. Dies kann synchron erfolgen, aber auch asynchron. So macht es bspw. Sinn, neue Materialien hervorzuheben (graphisch z. B. durch eigene Farbe oder ein Symbol), Veränderungen an Materialien können geeignet in Attributen kenntlich gemacht werden. Auf neue Annotationen an Materialien kann durch adäquate Darstellung der Materialien selbst hingewiesen werden. Diskussionsbeiträge müssen einem Mitglied zugeordnet werden können, das diese verfasst hat. Ebenso sind neue Beiträge in geeigneter Form zu kennzeichnen. Wenn es für die Nutzer in einer virtuellen Arbeitsumgebung

verschiedene Benutzerrollen gibt, so sind diese an der Darstellung der Benutzer zu verdeutlichen (vgl. [Ham02], S. 121f.).

Ein interessante Idee für asynchrone Wahrnehmung liefert das COMETS-Projekt an der Freien Universität Berlin (vgl. [Rad05]): Die Idee von COMETS liegt darin, Lernende miteinander zu vernetzen, die sich im gleichen Arbeits- oder Lernkontext befinden oder – und das ist das besondere – kürzlich befunden haben. Durch ein „Schweifmodell" werden Nutzer sichtbar, die einen bestimmten Kontext kürzlich zuvor besucht haben und somit noch als Hilfesteller dienen können. Dabei regelt jeder Nutzer seine „Schweiflänge" selbst und bestimmt damit, für welchen Zeitraum er für diesen Kontext Ansprechpartner sein möchte.

3.2.3. Technisierung virtueller Wissensräume

Die Metapher des kooperativen virtuellen Wissensraums ist auf raumbasierte Spieleumgebungen im Netz zurückzuführen, den so genannten *Multi User Dungeons* (MUDs) der 80er Jahre (vgl. [LAGS99]). Ein MUD besteht aus miteinander verbundenen Räumen, die Objekte enthalten, die mit einer Vielzahl von Eigenschaften ausgestattet sind. Benutzer konnten sich innerhalb dieser Räume bewegen und die Objekte betrachten und manipulieren. Durch die Interaktion seiner Benutzer mit anderen Benutzern, Objekten und Räumen ist ein MUD ein hochgradig interaktives, sich ständig veränderndes System. Ein hierarchisches Rechtemanagement ermöglichte es dem Administrator eines MUDs, bestimmte Benutzer mit der Fähigkeit auszustatten, neue Räume und Objekte zu erschaffen. Auf diese Weise wurden die Benutzer in die Ausgestaltung der virtuellen Welt mit einbezogen, was eine stärkere Beziehung der Benutzer zu ihren Avataren, den Räumen und Objekten im MUD schaffte.

War die Benutzungsschnittstelle der MUDs noch textbasiert – bspw. befördert das Kommando `@join <Benutzer>` den eigenen Avatar direkt in den Raum, in dem sich auch `<Benutzer>` aufhält – fanden sie mit den MOOs (*MUD Object Oriented*) ihre objektorientierte Umsetzung und mit ihnen auch zu graphischen Oberflächen. Somit sind MUDs quasi die Vorläufer der heutigen *Massive Multiplayer Online Games* (MMOGs) und haben die Grundlage zur Erschaffung von reich ausgearbeiteten Rollenspiel- und anderen virtuellen Welten wie *Second Life*[36] gelegt.

Analog dazu sind im Groupware-Bereich Anwendungen der Systemklasse *Shared Information Space* entstanden, die ebenfalls „der verteilten Nutzung von Informationen dienen, und bei denen gemeinsame Informationsbestände als eine wesentliche Komponente der Kommunikations-, Kooperations- und Koordinationsunterstützung zu betrachten sind" [F+02], S. 247f.

Die Datenspeicherung ist in beiden Fällen persistent, erstreckt sich auf strukturierte sowie unstrukturierte Informationen und stellt außerdem geeignete Zugriffsmechanismen bereit. In der Regel basieren ihre Datenmodelle auf wenigen, zentralen Objekten (vgl. Abb. 3.8):

[36] Second Life ist eine Online-3D-Infrastruktur für von Benutzern selbst gestaltete virtuelle Welten, in der sie als Avatare mit anderen Benutzern interagieren und sogar Handel betreiben können. Derzeit sind über 11 Millionen Nutzerkonten registriert. Die deutsche Webseite ist unter `http://de.secondlife.com` zu erreichen. Letzter Zugriff: 31.07.2008.

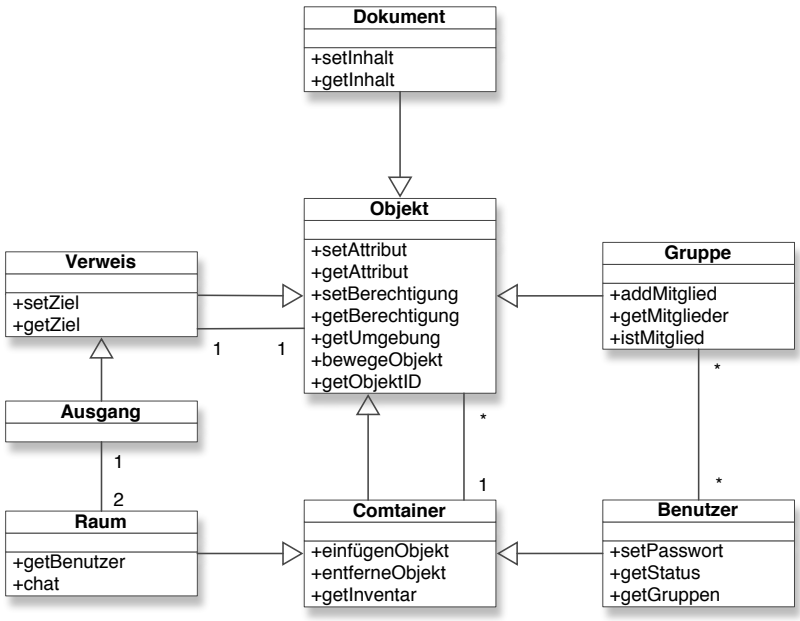

Abbildung 3.8.: Objektmodell der Wissensraummetapher nach [Ham02].

- Das zentrale Element `Objekt` stellt Hauptattribute und generische Medienfunktionen wie Erzeugen, Löschen, Verknüpfen, etc. bereit. Es besitzt auch Funktionen, um Attribute zu setzen und auszulesen, Berechtigungen zu setzen und zu prüfen und Informationen über die Umgebung abzufragen. Damit bildet es den Ausgangspunkt für alle weiteren Klassen, die alle diese Grundeigenschaften und -funktionen erben.
- Ein `Dokument` wird bspw. um die Möglichkeit der Inhaltsverwaltung von Dokumenten erweitert, wie das Setzen und Auslesen des Inhalts, und bspw. einer Versionierung.
- `Container` und `Räume` fassen Objekte zusammen. Während ein Container Dokumente enthalten kann, können sich in einem Raum als Spezialisierung eines Containers auch Nutzer aufhalten. Es besteht bei beiden Typen die Möglichkeit, Objekte hinzuzufügen und den Inhalt des Containers sowie seine Umgebung zu erfragen.
- Eine Besonderheit im Modell von Hampel ist, dass das `Benutzer`-Objekt vom Container abgeleitet wird. So trägt es einerseits als Objekt Berechtigungen und Attribute, kann gleichzeitig als Container beliebige Objekte bzw. Dokumente oder auch einfach nur Verweise auf Dokumente mit sich führen (*inventory*).
- Eine `Gruppe` ist von Objekt abgeleitet und für die Verwaltung der Teilnehmer einer Nutzergruppe zuständig. Es werden Berechtigungen gebündelt und Zugriffsstrukturen verwaltet.

➢ Eine Verknüpfung auf eine interne oder externe Ressource kann durch einen `Verweis` gekapselt werden. Verweise werden als bidirektionale Objekte verwaltet, sie können also auch zurückverfolgt werden und Attribute tragen. Die Klasse `Ausgang` ist eine Spezialisierung eines Verweises und sorgt dafür, dass Räume persistent miteinander verbunden werden können und die Nutzer durch den so entstandenen Korridor entlang ihrer gesetzten Berechtigungen wandern können[37].

Essentiell für die Implementierung von Lernumgebungen auf Basis der Wissensraummetapher sind Objektframeworks, die einen Zugriff auf diese persistenten Strukturen über eine objektorientierte Programmierschnittstelle erlauben. Ein solches Framework ist dabei nicht als bloße Zusammenstellung von Klassen und Funktionen zu verstehen, sondern als ein System wiederverwendbarer Designergebnisse. Schon der Entwurf der Lern- und Arbeitsszenarien kann sich des im Framework verankerten Objektmodells bedienen und ist somit als Ableitung eines konkreten Designs aus abstrakteren Strukturen zu sehen.

Damit solche Systeme als grundlegend in einer heterogenen Umgebung dienen können, wird der Gedanke von angepassten „Sichten“ auf virtuelle Wissensräume und Objekte selbst bei der Integration standardisierter Internet-Protokolle konsequent fortgesetzt: Gängige Kommunikations- und Infrastrukturprotokolle wie WebDAV, FTP, RSS, Instant Messaging oder Mail-Protokolle werden dabei innerhalb eines Raumes zusammengeführt. Diese direkte Verankerung von Medien- und Kommunikationsfunktionen in semantischen Strukturen verringert zum einen Medienbrüche, kann zum anderen aber zu einer völlig neuen Qualität medialer Integration von Semantik und Kommunikation führen (vgl. [SHB04]).

Eine weitergehende Betrachtung von Objektframeworks zur Entwicklung von Lern- und Arbeitsumgebungen erfolgt im Kontext der zu konzipierenden Softwarearchitektur in Abschnitt 4.1.5.3.

3.3. Normsprachliche Spezifizierung kooperativer Lernumgebungen

In diesem Abschnitt wird beschrieben, wie Lern- und Arbeitsszenarien auf Grundlage der Wissensraummetapher materialsprachlich spezifiziert und mithilfe einer wissensraumbasierten Klassenbibliothek (oder Framework) bzw. eines entsprechenden Modells von Proxy-Objekten softwaretechnisch umgesetzt werden kann.

Dazu wird zuerst eine Taxonomie von Eigenprädikatoren vorgestellt, die eine erste Basis für die Bildung normsprachlicher Aussagen zur Beschreibung von Lern- und Arbeitsumgebungen im universitären Kontext darstellt. Sie enthält die Bezeichner von Konzepten der vier Spezifikationsebenen Begriffe, Pädagogik, (SW-)Technik und Schnittstellen (vgl. Abb. 3.9).

[37] Eine Klasse von diesem Typ ermöglicht die semantische Vernetzung von Räumen, ist aber nicht immer vorhanden. Daher können in den meisten Systemen Räume analog zu einer Verzeichnisstruktur nur hierarchisch angeordnet werden.

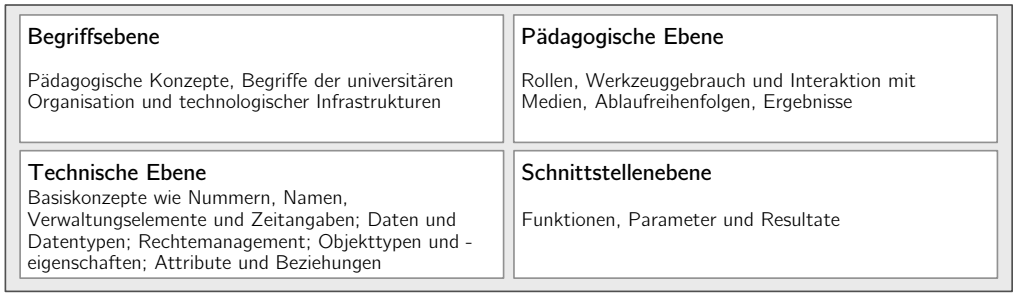

Abbildung 3.9.: Spezifikationsebenen universitärer Lern- und Arbeitsumgebungen (eigene Darstellung).

Ein wichtiger Bestandteil dieser Terminologie sind die Konzepte der Wissensraummetapher nach Hampel (vgl. [Ham02], S. 193ff.), mit deren Hilfe große Teile der Persistenz und der Interaktion mit Medien rekonstruiert werden können. Satzmuster und Beispiele für diese normsprachliche Rekonstruktion werden im darauf folgenden Abschnitt vorgestellt. Dabei wird – in Anlehnung an [Sch97a] – die Spezifikation unter den drei wesentlichen Perspektiven Statik, Funktionalität und Dynamik erklärt[38].

Zuletzt werden noch wichtige implementierungsspezifische Aspekte erläutert, bevor eine Zusammenfassung das Kapitel abschließt.

3.3.1. Das normsprachliche Lexikon

In der Abbildung 3.2 wurden bereits die Wortarten gegeneinander abgegrenzt, die bei der normsprachlichen Schemaerstellung verwendet werden können: Ding- und Geschehnisprädikatoren, Ding- und Geschehnisapprädikatoren sowie Partikel wie Präpositionen, Demonstrativpronomen usw. Für die Rekonstruktion und Beschreibung von domänen- und anwendungsspezifischen Konzepten sind in erster Linie die Eigenprädikatoren mit in das Lexikon aufzunehmen. Also Substantive und Tätigkeitsbeschreibungen wie Verben oder Ähnliches. Darüber hinaus werden Apprädikatoren (also Adjektive und Adverbien) für die Rekonstruktion von Zuständen und Zustandsübergängen von Konzepten benötigt. I. d. R. ist dies jedoch eher anwendungs- als domänenspezifisch. Im Kontext des Lexikons werden daher nur wenige Apprädikatoren behandelt[39]. Auf Präpositionen, bestimmte und unbestimmte Artikel usw. wird im Lexikon ganz verzichtet, da diese im Regelfall keiner weiteren Erklärung bedürfen.

[38] Die Unterscheidung der drei wichtigsten Aspekte der Modellierung von Softwaresystemen – Daten, Funktion, und zeitliches Verhalten – wird in nahezu allen verbreiteten objektorientierten Analyse- und Entwurfsmethoden vorgenommen und durch spezielle Techniken unterstützt.

[39] Speziell sind dies verschiedene Stati wie *eingeloggt*, *authentifiziert*, *angemeldet*, *registriert* oder *zugelassen*.

3.3.1.1. Dingprädikatoren des universitären Lernens und Arbeitens

Das Begriffssystem der Domäne „universitäres E-Learning“ enthält Konzepte der vier Spezifizierungsebenen Begriffe, Pädagogik, (SW-) Technik und Schnittstellen, mit denen Anforderungen aus Sicht der Projektbeteiligten formuliert werden können (vgl. Abb. 3.9).

Begriffsebene Universitäre Lern- und Arbeitsszenarien sind begrifflich eingebettet in hochschulische Organisationskonzepte und in eine technische Infrastruktur. Dem entsprechend gehören grundlegende Organisations- und Technologiekonzepte wie verschiedene *Organisationseinheiten* und Typen von *Mitarbeitern* dazu. Ebenso Technologien wie *Lernobjekte* und verschiedene *Lernwerkzeuge*, wie bspw. ein $Whiteboard$ oder ein $Mail - Client$.

Pädagogische Ebene Zur grundlegenden Beschreibung didaktischer Szenarien wurden die Konzepte des IMS Learning Designs (IMS LD, s. S. 43) mit in die Taxonomie übernommen. Die IMS-Spezifikation hat den Anspruch, vollständige didaktische Arrangements innerhalb einer definierten Umgebung mitsamt ihren Rollen, Aktivitäten, den verwendeten Lehr-/Lernmaterialien und den erzielten Ergebnissen XML-basiert abzubilden. Didaktische Methoden (Szenarien) werden im Learning Design analog zu einem Theaterspiel (*play*) in ein oder mehrere sequentiell aufeinander folgende Akte (*acts*) aufgeteilt, in denen die Personen in bestimmten Rollen (*roles*/*role-parts*) Handlungen (*activities*) vornehmen. Nach dem in Abbildung 3.10 dargestellten Informationsmodell des IMS LD lässt sich das Kernkonzept der Spezifikation daher formulieren als „Personen führen in unterschiedlichen Rollen Aktivitäten in bestimmten (Lern-) Umgebungen durch“. In Anlehnung daran wurden die einzelnen Konzepte $Person$, $Rolle$, $Aktivität$ und $Umgebung$ in das Lexikon aufgenommen. Weiterhin wurde die Klassifizierung als $Lernaktivität$ und $Unterstützungsaktivität$ sowie mit dem Konzept der $Aktivitätsstruktur$ die Möglichkeit zur Strukturierung von Aktivitäten durch Teil-Aktivitäten übernommen. Die für die Anwendung dieser Konzepte benötigten Satzmuster zur Formulierung von Spezialisierungen, Kompositionen und Tätigkeitsaussagen werden in den weiteren Abschnitten dieses Kapitels vorgestellt.

Basiskonzepte Zur Beschreibung informationstechnischer Aspekte wie die Verwaltung von Objekten und Daten im Allgemeinen werden in der Terminologie Basiskonzepte wie *Nummern*, *Namen*, *Verwaltungselemente* und *Zeitangaben* definiert. Darüber hinaus erfolgt die persistente Abbildung von Objekten der zu implementierenden Lern- und Arbeitsumgebungen über die bereits vorgestellten Basisobjekte der Wissensraummetapher (vgl. Abschnitt 3.2.2): *Dokument*, *Objekt*, *Container* und *Räume*, *Benutzer* und *Gruppen* sowie Verweise wie *Verknüpfungen* und *Ausgänge*. Konzepte des universitären E-Learnings sollten – wenn möglich – auf Basis dieser Begriffe normsprachlich durch Spezialisierung und Komposition rekonstruiert werden.

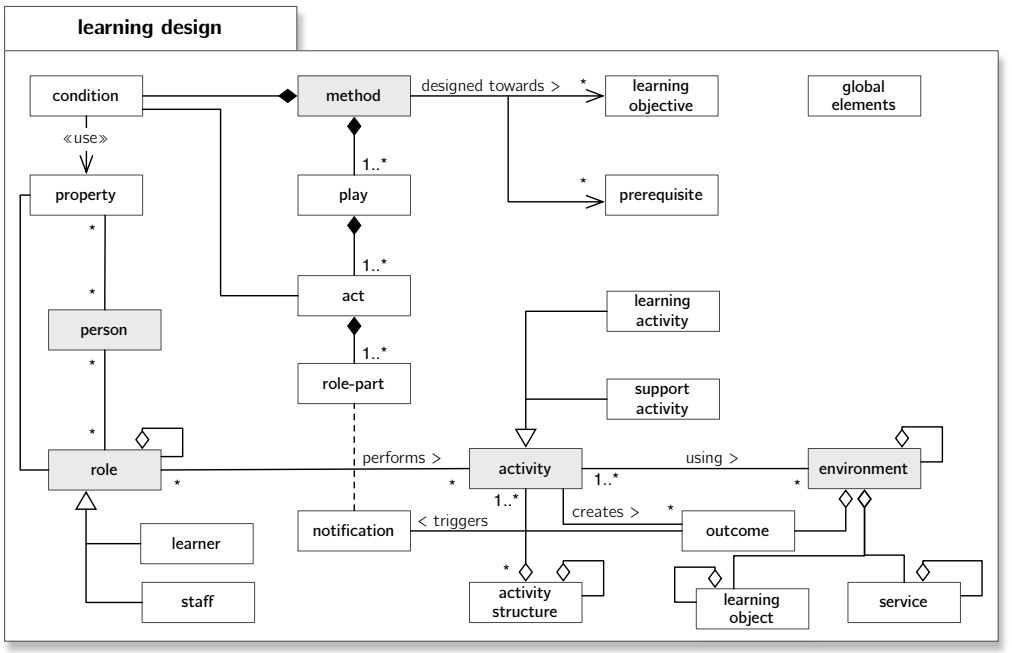

Abbildung 3.10.: Das Informationsmodell des IMS LD (aus: [IMS03b]).

Dabei können Konzepte des Learning Designs mit Konzepten der Wissensraummetapher kombiniert bzw. abgebildet werden. Bspw. können Rollen durch Gruppen umgesetzt, Wissensräume als Lernumgebung gestaltet und Aktivitätsstrukturen durch Medienfunktionen unterstützt werden.

Schnittstellen In dieser Arbeit wurde bereits auf S. 26ff. die Wichtigkeit der Integration moderner Lernumgebungen in die umgebende IT-Infrastruktur herausgestellt. Eine Anforderung an die zu erstellende Normsprache ist daher die Beschreibungsfähigkeit von funktionalen Schnittstellen zwischen Systemen, genauer, mit welchen *Parametern* eine *Operation* von einem *Klassifizierer* (*Klasse* oder *Komponente*) aufgerufen werden kann und welche *Resultate* zurückgegeben werden. Dabei soll die Normsprache jedoch nur eine Ergänzung zu existierenden technischen Beschreibungssprachen von Schnittstellen[40] darstellen und diese nicht ersetzen. Die Kommunikation des Systemanalysten mit den Endanwendern verläuft meistens nicht auf einem solch technischen Niveau, so dass diese grundlegende Beschreibung von (entfernten) Funktionsaufrufen in der Regel ausreicht.

[40] Bspw. die Web Service Description Language (WSDL) oder die Component Definition Language (CDL).

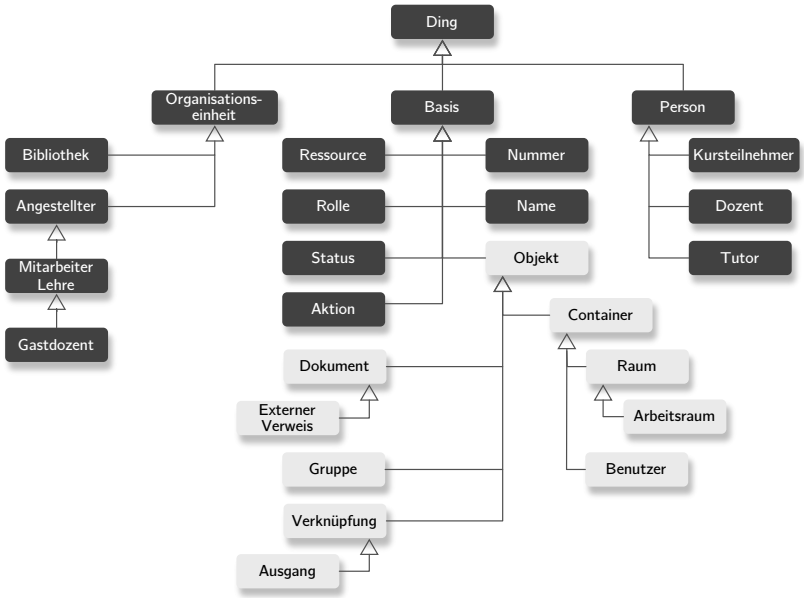

Abbildung 3.11.: Die Basis-Dingprädikatoren (Q) der Terminologie (Ausschnitt: Objekte der Wissensraummetapher, vgl. Anhang S. 256)

3.3.1.2. Geschehnisprädikatoren

Aktivitäten können normsprachlich durch Geschehnisprädikatoren ausgedrückt werden. In Kombination mit einer entsprechenden Kopulae können damit sowohl Verhaltensaussagen als auch Aussagen zu Befähigungen und Berechtigungen von handelnden Subjekten gemacht werden (vgl. Abb. 3.3). Durch Satzmuster zur Komposition können verschiedene Aktivitäten Instanzen des vorgestellten Konzepts der Aktivitätsstruktur zugeordnet werden, das somit eine beliebig tiefe Hierarchie beschreiben kann. Später in diesem Kapitel wird noch gezeigt werden, wie zeitliche und/oder kausale Beziehungen zwischen (Teil-) Aktivitäten zur Beschreibung von sequentiellen, parallelen und alternativen Ablaufreihenfolgen abgebildet werden können.

Die bereits vorgestellten primären Medienfunktionen (S. 92) bilden einen großen Teil der Aktivitäten, nämlich *Interaktionen* von Benutzern mit den Informationsobjekten. Ebenfalls in den Bereich der Interaktion fallend ergänzen *aufrufen* und *zurückgeben* die Dingprädikatoren zur Schnittstellenbeschreibung zwischen zwei Komponenten.

Weiterhin sind Dingprädikatoren zur Beschreibung

- der Durchführung von in sich abgeschlossenen Szenarien (*beginnen, abbrechen, abschließen*)

- von Benutzereingaben (*eingeben, aufrufen, registrieren, etc.*)

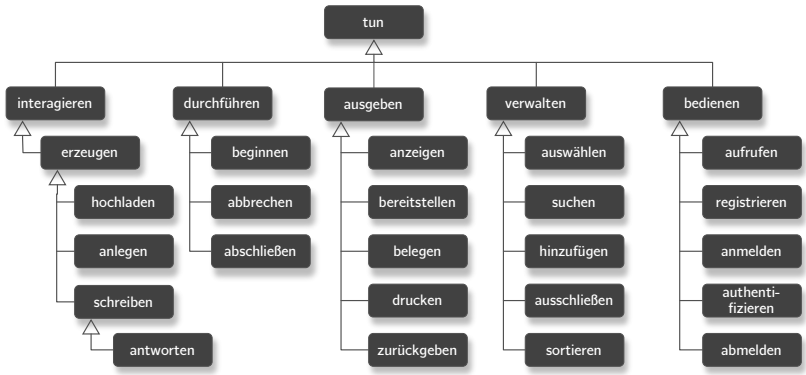

Abbildung 3.12.: Die Basis-Geschehnisprädikatoren (P) der Terminologie (Ausschnitt, vgl. Anhang S. 255)

- ➢ von Systemausgaben als Antwortverhalten auf Benutzereingaben (*anzeigen, drucken, etc.*) und
- ➢ von möglichen Verwaltungstätigkeiten (*suchen, hinzufügen, sortieren, etc.*).

Die Abbildungen 3.11 und 3.12 geben noch einmal einen Überblick über die Konzepte.

3.3.2. Statische Sicht

Die statische Sicht[41] umfasst die Spezifikation der statischen, d. h. über die Zeit unveränderlichen Teile des grundlegenden Informationsmodells. Dazu werden Klassen von Objekten beschrieben, ihre Attribute und Gültigkeitsbereiche, sowie ihre strukturellen Beziehungen zueinander. In diesem Abschnitt werden diese statischen Aspekte aufgegriffen und gezeigt, welche Grammatik, insbesondere welche Satzbaupläne dazu benötigt werden, um relevante Konzepte (Prädikatoren) eines Anwendungsbereiches sowie die semantischen Beziehungen zwischen ihnen definieren zu können.

Statische Beziehungen zwischen den Prädikatoren sind bspw. Kompositionen, Relationen, Vererbungsbeziehungen, Existenzabhängigkeiten und Zuordnungen von konkreten Objekten zu einer Menge von Objekten. Einige Beispiele zur Modellierung wurden bereits in Kapitel 4.1 gegeben, auf die im Folgenden weiter aufgebaut wird.

3.3.2.1. Kompositionen

Kompositionen (*composite aggregations*) beschreiben Ganzes-Teil-Beziehungen, bei der die Instanz eines systemhaften *Ganzen* durch die Instanzen seiner Teile definiert wird. Hierüber lassen sich bspw. komplexere Objektstrukturen konstruieren, die sich aus verschiedenen einfachen Typen zusammensetzen. Aber auch einfache Inklusionsbeziehungen wie Zugehörigkeiten sind darüber beschreibbar. Nach [Sch97a], S. 226, lassen sich fünf

[41] In der Literatur auch oft als *Datensicht* oder *Struktursicht* bezeichnet.

Kompositionstypen unterscheiden: Ganzheit, Kollektion, Behältnis, Verschmelzung und Zuordnung. Um eine Spezifikation auf Basis der Normsprache entsprechend differenziert vornehmen zu können, lautet das Satzmuster für Aussagen zu Ganzes-Teil-Beziehungen in Erweiterung zu 3.8:

$$[N_1 \mid \sigma_{Kompositionstyp} \mid N_2] \tag{3.11}$$

Somit lassen sich bspw. Unterstrukturen, Mitgliedschaften und Umgebungen beschreiben:

❑ Kursraum	$\sigma_{Ganzheit}$	Materialpool
❑ Gruppe	$\sigma_{Kollektion}$	Gruppenmitglied
❑ Raum	$\sigma_{Behältnis}$	Benutzer, Container, Dokument, Verweis

(3.12)

Die Definition einer Gruppe sieht vor, dass sie sowohl aus Benutzern (Gruppenmitgliedern) als auch aus Gruppen (Untergruppen) bestehen kann. Dabei sollen alle Mitglieder von Untergruppen zwangsläufig auch Mitglied in den entsprechenden Obergruppen sein. Auf dieser Forderung baut ein Teil des dynamischen Rechtemanagements (S. 107) auf, weshalb sie an dieser Stelle normsprachlich definiert werden soll:

$$\forall\, x, y\, \varepsilon\, Gruppe, z\, \varepsilon\, Benutzer\ (\\ (x\, \sigma_{Kollektion}\, y) \wedge (y\, \sigma_{Kollektion}\, z) \rightarrow x\, \sigma_{Kollektion}\, z\ ) \tag{3.13}$$

Partivative Beziehungen können so stark sein, dass ein Teil ohne das dazugehörende Ganze alleine nicht existieren kann. Zum Beispiel kann es kein Gruppenmitglied ohne die dazugehörende Gruppe geben. Aber auch der Inhalt eines Raums kann u. U. existenziell von seiner Umgebung abhängig sein. Daher kann eine normsprachliche *Existenzabhängigkeit* nicht per se zwischen zwei Prädikatoren definiert werden, sondern muss auch die Beziehung (*Konnexion*) umfassen, welche die Abhängigkeit begründet. Eine existenzielle Abhängigkeit von den Objekten in einem Raum zum Raum selbst kann so beschrieben werden:

$$\forall\, x\, \varepsilon\, Raum, y\, \varepsilon\, \{Container, Dokument, Verweis\}\ (\\ x\, \varepsilon\, existieren\ \wedge x \sigma_{Behältnis} y \rightarrow y\, \varepsilon\, existieren\ ) \tag{3.14}$$

Demnach soll der Inhalt eines Raums gelöscht werden, wenn auch der Raum gelöscht wird. Benutzer, die sich in einem Raum aufhalten können, sind in dieser Aussage (und somit auch in der Löschung) nicht eingeschlossen.

Ist es notwendig, eine Beziehung zwischen zwei Prädikatoren näher zu beschreiben, sie also als eigenständige Entität behandeln zu müssen, können dazu so genannte Relatoren verwendet werden (vgl. [Sch97a], S. 205). Hierdurch werden Beziehungen zwischen konkreten Ausprägungen spezifischer Prädikatoren als eigenständige konkrete Objekte unter einem neuen Prädikator (Relator) zusammengefasst. Seine Instanzen sind dabei existenziell abhängig von den durch die ihn verbundenen Objekten.

Im vorgestellten Objektmodell der Wissensraummetapher können Relatoren beispielsweise die Verknüpfungsmöglichkeiten von Räumen mittels Türen respektive Ausgängen

darstellen. Der Relator „Ausgang“ kann danach wie folgt beschrieben werden:

$$\begin{array}{l} Quelle, Ziel\,\varepsilon\,Raum \\ Ausgang(\,Quelle,\,Ziel\,) \end{array} \tag{3.15}$$

Ein *Ausgang* beschreibt also eine gerichtete Verbindung zwischen zwei Räumen (Quelle und Ziel). Mithilfe dieses neuen Prädikators können dann Aussagen über Raumstrukturen getätigt werden, die an weitere Verbindungen zwischen Objekten gekoppelt sind: „Jede Person, die Mitglied in einer Gruppe ist, hat in ihrem persönlichen Arbeitsraum einen Ausgang zu dem entsprechenden Gruppenarbeitsraum“ kann man normsprachlich mit den Satzbauplänen 3.8 und 3.11 wie folgt ausdrücken:

$$\begin{array}{l} \forall\, a\,\varepsilon\, Person, b, d\,\varepsilon\, Arbeitsraum, c\,\varepsilon\, Gruppe\ (\\ \quad [\,a \mid \sigma \mid b\,] \wedge [\,c \mid \sigma_{Kollektion} \mid a\,] \wedge [\,c \mid \sigma \mid d\,] \\ \quad \rightarrow [\,b \mid \sigma_{Behältnis} \mid Ausgang(b,d)\,]\ ) \end{array} \tag{3.16}$$

Das Konzept der Komposition ist nicht nur auf Ding-Prädikatoren beschränkt. Es kann – auf Geschehnis-Prädikatoren angewendet – auch die Zerlegung von Aktivitäten in einzelne Tätigkeiten beschreiben. In Kombination mit definierten Ablaufreihenfolgen lassen sich damit sehr gut komplexere Lernszenarien definieren (vgl. Abb. 3.14).

3.3.2.2. Spezialisierung

Neben der Komposition ist das Konzept der Spezialisierung sehr wichtig, um das statische Modell eines Anwendungsbereiches zu konstruieren. Die Spezialisierung verbindet zwei Prädikatoren vom selben Typ, wobei einer eine Unterklasse vom anderen ist. Mit anderen Worten: Der zweite Begriff ist generischer oder abstrakter als der erste. Somit lässt sich eine begriffliche Unter-/Überordnung von Prädikatoren festlegen, bei der die unter einem Oberbegriff inkludierten Spezialisierungen dessen Eigenschaften und Beziehungen erben. Das dazugehörige Satzmuster wurde mit 3.7 bereits vorgestellt, soll an dieser Stelle aber noch einmal verfeinert werden, um verschiedene Arten der Inklusion ausdrücken zu können:

$$[N_1 \mid \varepsilon_{Inklusionstyp} \mid N_2] \tag{3.17}$$

Somit lassen sich bspw. statische Art/Gattungs-Beziehungen und dynamische Rollenbeziehungen definieren (vgl. [Sch97a], S. 224f.):

$$\begin{array}{lll} \text{❑ Fakultät} & \varepsilon_{Art} & \text{Organisationseinheit} \\ \text{❑ Gruppenadministrator} & \varepsilon_{Rolle} & \text{Person} \end{array} \tag{3.18}$$

Generelle Aussagen zu diesen Inklusionsbeziehungen sind bspw.
„$\forall x\,(x\,\varepsilon\,Fakultät \rightarrow x\,\varepsilon_{Art}\,Organisationseinheit)$“ oder
„$\forall\, x\varepsilon Person,\ y\,\varepsilon\,Gruppe\,(x\,\pi\,anlegen\,y \ \rightarrow\ Gruppenadministrator(x,y)\,\varepsilon_{Rolle}\,x)$“ mit dem Relator $Gruppenadministrator(Person, Gruppe)$.

Abbildung 3.13 stellt die Umsetzung einer statischen Struktur in Programmcode noch einmal grafisch dar.

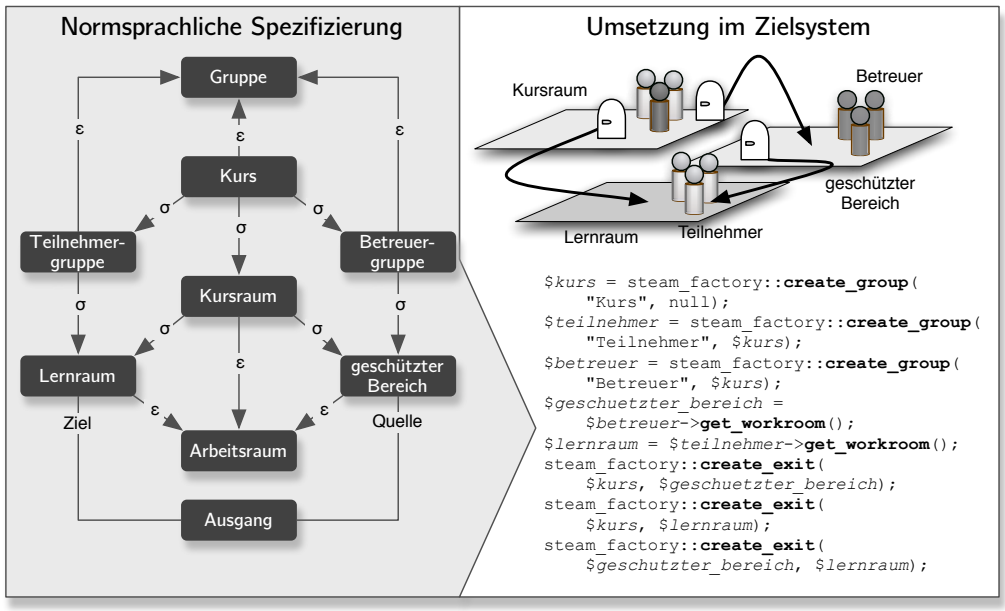

Abbildung 3.13.: Spezifizierung und Implementierung einer statischen Raum- und Gruppenstruktur für Kurse.

3.3.2.3. Attribute

Begriffe typisieren konkrete oder abstrakte Objekte mit gleichen Merkmalen (Attributen). Dabei können sich die Objekte eines Typs durch verschiedene Merkmalsausprägungen voneinander unterscheiden. Mithilfe der Normsprache müssen deshalb (1) allgemeine Aussagen formuliert werden können, welche die Zuordnung von Attributen zu Objekttypen ausdrücken, und (2) singuläre Aussagen zu den Merkmalsausprägungen konkreter Objektinstanzen. Das folgende Satzmuster beschreibt die schematische Zuordnung eines Attributes Q_2 zu Instanzen des Objekttyps Q_1 mit Σ für „haben“. Mehrere Attribute können in einem Satz mit Kommata angehängt werden:

❏ Natürliche Sprache:	[Studenten haben eine Matrikelnummer.]	
❏ Normsprache:	[Student $\mid \Sigma \mid$ Matrikelnummer]	(3.19)
❏ Satzbauplan:	$[Q_1 \mid \Sigma \mid Q_2, \ldots Q_n]$	

Dabei ist das Attribut Q_2 (hier: Matrikelnummer) ein eigenständiger Prädikator, der in Aussagen zur konkreten Ausprägungen herangezogen werden kann:

❏ Natürliche Sprache:	[Der Student Müller hat die Matrikelnummer 123456.]	
❏ Normsprache:	$[(\text{Müller} \mid \varepsilon \mid \text{Student})\sigma(123456 \mid \varepsilon \mid \text{Matrikelnummer})]$	
❏ Satzbauplan (kurz):	$[N_1 \mid \sigma \mid (N_2 \mid \varepsilon \mid Q_1)]$	(3.20)

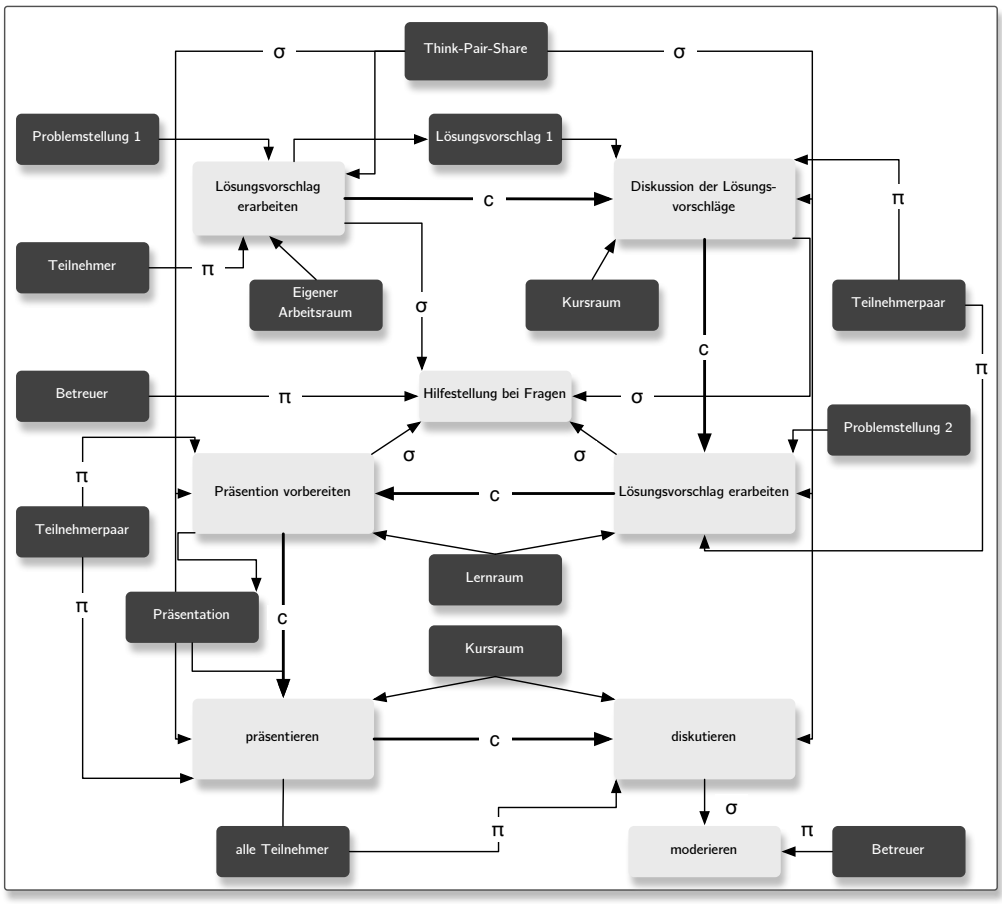

Abbildung 3.14.: Spezifizierung der funktionalen Sicht am Beispiel des didaktischen Szenarios *Think-Pair-Share*.

3.3.3. Funktionale Sicht

Die im vorhergehenden Abschnitt beschriebenen Satzbaupläne ermöglichen die Rekonstruktion der statischen Struktur eines Anwendungsbereichs, d. h. die Definition relevanter Konzepte sowie deren semantische Beziehungen zueinander. Aus funktionaler Sicht werden im Folgenden Möglichkeiten zur Beschreibung des Verhaltens eines Systems innerhalb dieses Anwendungsbereichs aufgezeigt, also beobachtbare Interaktionen zwischen Instanzen dieser Konzepte. Dabei stehen insbesondere die normsprachliche Beschreibung der Handlungsfähigkeit von Objekten, sowie Handlungskontexte mit Vor- und Nachbedingungen im Fokus.

3.3.3.1. Befähigungen

Die im Objektmodell definierten Methoden beschreiben die unterste Verfeinerungsschranke elementarer Handlungen (Teilhandlungen). Auf dieser Basis können mithilfe der im vorigen Abschnitt vorgestellten Komposition komplexere Aktivitätsstrukturen beschrieben werden[42], die der Rekonstruktion von Handlungsabläufen dienen. Ein entsprechendes Satzmuster dazu wurde bereits einleitend mit 3.9 vorgestellt. Beispiele sind:

$$
\begin{aligned}
&\text{❏ } [Benutzer \mid \pi \mid annotieren \mid Antwort \mid an-Forenbeitrag] \\
&\text{❏ } [Student \mid \pi \mid arrangieren \mid Dokument \mid im-Arbeitsraum \mid mit-Whiteboard] \\
&\text{❏ } [Meier \mid \pi \mid schreiben \mid Wikibeitrag \mid mit-Editor \mid mit-Dokument] \\
&\text{❏ } [Administrator \mid \pi \mid einladen \mid Benutzer \mid in-Gruppe \mid mit-Mail]
\end{aligned}
\tag{3.21}
$$

Anhand dieser Aussagen lassen sich die typspezifischen Fähigkeiten von Objekten ableiten, wobei die beschriebene Fähigkeit i. d. R. dem unmittelbar durch die Handlung betroffenem Objekt zugeschrieben wird (vgl. [Sch97a], S. 244f.)[43]. Direktes Objekt der Handlung *annotieren* ist bspw. ein Objekt vom Typ *Antwort*. Als Ergebnis der Handlung wird dieses Objekt an einen Forenbeitrag „angeheftet". Werden im Zielsystem Merkmale von Antworten als Attribute des Objekttyps *Antwort* verwaltet, so sollte – falls andere Designentscheidungen nicht dagegen sprechen – der veränderte Zustand des „angeheftetseins" sowie die dazugehörige Zustands-verändernde Methode dem Typ *Antwort* zugeordnet werden. Gleiches gilt für das Arrangieren von Dokumenten, das Schreiben von Wikibeiträgen und das Einladen von Benutzern.

An den Beispielen lässt sich weiterhin gut erkennen, dass Aussagen nach diesem Muster nicht nur das handelnde Subjekt und das direkte Handlungsobjekt definieren können, sondern darüber hinaus auch den Handlungsraum, Hilfsmittel, und weitere indirekte Objekte. Alles gemeinsam charakterisiert die Handlungssituation.

3.3.3.2. Berechtigungen

Die in Abschnitt 3.2.3 vorgestellten Plattformen nutzen keine statischen Rollen, die auf ein bestimmtes Anwendungsszenario zugeschnitten sind, sondern ein offenes und flexibles, objektbezogenes Rechtemanagement, das auf eine Situation hin angepasst werden kann. Dies kommt der Tatsache entgegen, dass sich Benutzer und Objekte in verschiedenen kooperativen Szenarien zumeist nicht durch ihre allgemeinen Handlungsfähigkeiten unterscheiden, wohl aber durch verschiedene Berechtigungen, diese Handlungen durchzuführen. Zum Beispiel ist ein Student grundsätzlich dazu fähig, den gemeinsamen Dokumentenpool zu einem Lernszenario einzusehen. Er bekommt das Leserecht auf die dort abgelegten Dokumente in diesem Fall aber erst, nachdem er selbst seinen Beitrag in den Pool hochgeladen hat (vgl. [RHS06]).

[42] Bspw. „DokumentLesen $\sigma_{Ganzheit}$ DokumentÖffnen, InhaltAusgeben, InhaltLesen".

[43] Die Zuordnung von Fähigkeiten zu Objekttypen bspw. in der Form von Objektmethoden ist jedoch vorrangig Teil des methodenspezifischen Entwicklungssystems und daher abhängig von den verwendeten Entwurfsmethoden und -paradigmen. Sie soll aus diesem Grund hier nicht näher behandelt werden.

Um solche statischen und dynamischen Berechtigungen normsprachlich rekonstruieren zu können, wird im Folgenden ein Satzmuster vorgestellt, dass auf der in der Aufstellung 3.3 einleitend bereits erwähnten Berechtigungskopula basiert.

❏ Natürliche Sprache:	[Der Student darf den Dokumentenpool nicht lesen.]	
❏ Normsprache:	[Student | ϑ' | lesen | Dokumentenpool]	(3.22)
❏ Satzbauplan:	[N_1 | ϑ | P | N_2]	

Die einzelnen Berechtigungen – also Lesen, Schreiben/Löschen, Ausführen, Annotieren, etc. – können dann als Relatoren definiert werden, damit sie als jeweils eigenständiges Recht in normsprachlichen Aussagen verwendet werden können: [N_1 \| σ \| $Leserecht$ \| $auf - N_2$], oder kurz:

$$\begin{array}{l} Leserecht(Benutzer/Gruppe, Objekt), \text{ mit} \\ \forall\, x\, \varepsilon\, (Benutzer \vee Gruppe), y\, \varepsilon\, Objekt(Leserecht(\, x,\, y\,) \leftrightarrow x\, \vartheta\, lesen\, y) \end{array} \tag{3.23}$$

Hiermit können allgemeine Aussagen zu den Rechten bestimmter Rollen der Anwendung getätigt werden. Singuläre Aussagen definieren darüber hinaus eine objektbezogene *Access Control List* (ACL, vgl. hierzu [Lam74], S. 20, und [GB98]). Also eine zweidimensional angeordnete Matrix mit $f : B \times t$ und $B\, \varepsilon\, (Benutzer \vee Gruppe), t\, \varepsilon\, tun$ und einer Funktion

$$f(Benutzer, tun) = \begin{cases} erlaubt, & \text{wenn Benutzer} \in \text{ACL für } tun \\ nicht\, erlaubt, & \text{wenn Benutzer} \notin \text{ACL für } tun. \end{cases}$$

Die Gesamtheit der Rechte eines Benutzers oder einer Gruppe an einem Objekt wird in Analogie zu den einzelnen eigenständigen Rechten normsprachlich mit dem zweistelligen Relator *Zugriffsrecht*(*Benutzer*/*Gruppe*, *Objekt*) ausgedrückt, welcher für die Modellierung dynamischer Aspekte – insbesondere Rechtevererbung und -weitergabe – benötigt wird. Ohne die Argumente wird der Prädiktor „Zugriffsrecht“ nicht auf die Rechte einzelner Benutzer oder Gruppen beschränkt, sondern repräsentiert alle definierten Zugriffsrechte eines Objektes. Er kann dann als Objektattribut verwendet werden: $Objekt\, \Sigma\, Zugriffsrecht$.

Auf den dynamischen Teil eines flexiblen Rechtemanagements für kooperative virtuelle Umgebungen wird im folgenden Abschnitt *Dynamische Sicht* näher eingegangen.

3.3.4. Dynamische Sicht

3.3.4.1. Rechtevererbung und -weitergabe

Mit dem Satzmuster 3.22 können Zugriffsbeziehungen zwischen Objekttypen und ihren Instanzen definiert werden, die statisch bspw. durch eine Rollenzugehörigkeit erlangt werden kann. In der praktischen Anwendung von Lern- und Arbeitsumgebungen haben Rechtestrukturen darüber hinaus auch dynamischen Charakter. So müssen objektbezogene Zugriffsrechte angepasst werden, wenn ein Dokument von einem zum anderen Benutzer weitergegeben oder von einem privaten in einen Gruppenarbeitsraum bewegt wird. Wie einleitend auf S. 93 schon erwähnt, unterscheidet Hampel diesbzgl. in [Ham04],

S. 17, (1) die Erlangung von Zugriffsrechten über eine soziale Gruppenstruktur, (2) die Übertragung von Rechten eines anderen Benutzers (Delegation) und (3) der Ableitung von Rechten eines Objekts durch seine Umgebung[44].

Die Ableitung von Zugriffsrechten über die Mitgliedschaft in einer Gruppe kann mit der Regel 3.14 und dem Satzmuster 3.22 wie folgt definiert werden:

$$\begin{aligned}&\forall\, x\,\varepsilon Gruppe, y\,\varepsilon Benutzer, z\,\varepsilon Objekt(\\&(x\,\sigma\ Zugriffsrecht\,auf-z)\wedge(x\,\sigma_{Kollektion}\,y)\rightarrow\\&(y\,\sigma\ Zugriffsrecht(x,\ z)\,auf-z))\end{aligned}\tag{3.24}$$

In vielen Umgebungen ist zunächst einmal der „Eigentümer“ bzw. der „Erzeuger“ eines Objektes zuständig für dessen Verwaltung. Dazu ist er i. d. R. im Besitz aller Rechte auf das Objekt. Alle in Abschnitt 4.2.3 vorgestellten Plattformen erlauben jedoch die Delegation der Verantwortung für ein Objekt durch das Einräumen entsprechender Rechte für einzelne Benutzer oder ganze Gruppen. Dieses Recht, anderen Rechte an ein Objekt einzuräumen, wird hier unter dem Relator „Sanktionsrecht“ (aus dem Englischen: *sanction right*, vgl. [Ham04], S. 22) verstanden. Analog zu allen übrigen Rechten (vgl. Definition 3.23) beruht die Rekonstruktion des Sanktionsrechts normsprachlich auf einer Bisubjunktion:

$$\begin{aligned}&Sanktionsrecht(Benutzer/Gruppe, Objekt),\text{ mit}\\&\forall\, x\,\varepsilon\,(Benutzer\vee Gruppe), y\,\varepsilon\,Objekt, y\,\sigma\,(z\,\varepsilon\ Zugriffsrecht)\\&(Sanktionsrecht(\,x,\ y\,)\leftrightarrow x\,\vartheta\,weitergeben\,z)\end{aligned}\tag{3.25}$$

Abschließend ist die Übernahme von Zugriffsrechten von Objekten über ihre Umgebung zu regeln. Dass ein Dokument grundsätzlich erst einmal die Zugriffsrechte seines ihn umgebenden Containers erbt, wird etwa durch die folgende Aussage behauptet:

$$\begin{aligned}&\forall\, a\,\varepsilon\,Dokument, b\,\varepsilon\,Container\\&\quad((b\,\sigma_{Behältnis}\,a)\wedge(b\,\sigma(c\,\varepsilon\ Zugriffsrecht))\rightarrow a\,\sigma\,(c\,\varepsilon\ Zugriffsrecht))\end{aligned}\tag{3.26}$$

3.3.4.2. Aktivitäten

Um komplexe Verläufe von einzelnen Funktionen oder ganzen Prozessketten unter Berücksichtigung von Reihenfolgen, Nebenläufigkeiten und alternativen Entscheidungswegen rekonstruieren zu können, müssen normsprachliche Aussagen dazu in ein kausales Ordnungsschema eingebunden werden.

Schienmann führt dazu in [Sch97a], S. 277ff., nach [Lor00] zunächst einen konstruktiven Subjunktor $\rightarrow_c$ ein, der eine strikte Sequenzialität zwischen dem Vordersatz und dem Hintersatz in einem Bedingungssatz fordert: $A\rightarrow_c B\neq\neg A\vee B$. Somit kann eine Sequenz von Geschehnissen auf Schemaebene dargestellt werden als $A\rightarrow_c B$, wobei die Großbuchstaben zur Bezeichnung von Geschehnissen dienen. Steht A bspw. für „$Benutzer\,\pi\,gründen\,Gruppe$“ und B für „$Benutzer\,\pi\,einladen\,Benutzer\,in-Gruppe$“,

[44] Hampel schließt zusätzlich den Zeitpunkt der Erbschaft als Klassifikationsmerkmal mit ein, also ob ein Objekt die Rechte bspw. beim Erzeugen, Bewegen, oder der Bildung einer neuen Gruppe erlangt. Dies soll jedoch für ein Satzmuster an dieser Stelle keinen Unterschied machen.

dann wird mit dem Bedingungssatz ein kausaler Zusammenhang definiert, dass eine Instanz vom Geschehnistyp A (a) erst passiert sein muss, bevor eine Instanz von B (b) ablaufen kann. Erst wenn ein Benutzer eine Gruppe gegründet hat, kann er dazu andere Benutzer als Mitglieder einladen.

Wenn Geschehnisse *nicht* in einer kausalen Beziehung zueinander stehen, können sie parallel ablaufen (vgl. ebd.). Die Konjunktion $A \rightarrow_c (B \wedge C) \rightarrow_c D$ drückt eine Parallelverzweigung von A nach B und C aus, die wieder vor D zusammengeführt wird.

In Analogie dazu können alternative Entscheidungswege logisch mithilfe einer Disjunktion $A \rightarrow_c (B \vee C) \rightarrow_c D$ modelliert werden. Anstelle einer Parallelverzweigung können hier entweder b oder c als Instanzen der Geschehnisaussagen B oder C ausgeführt werden, um eine Instanz von D aktualisieren zu können.

3.3.4.3. Zustände und Zustandsübergänge

Zuletzt werden Möglichkeiten geschaffen, das Verhalten beliebiger Prädikatoren normsprachlich zu rekonstruieren. Dies ist notwendig, um Aussagen zum Systemverhalten – also wie verhält sich das System in einem bestimmten Zustand beim Eintreffen bestimmter Ereignisse – festzuhalten.

In der UML spezifiziert man dieses Verhalten mit Zuständen, die ein Objekttyp einnehmen kann und Übergängen zwischen den Zuständen, die durch Ereignisse initiiert werden können. Dabei können Attribute eines Prädikators als Zustandsvariablen definiert werden, deren endliche Menge von zulässigen Werten die möglichen Zustandsausprägungen beschreiben.

Normsprachlich werden diese Ausprägungen durch Ding-Apprädikatoren (q), also Adjektiven ausgedrückt. Die möglichen Zustände eines Prädikators bildet daher die Menge alternativer Apprädikatoren, was durch die folgende disjunktive Prädikatorenregel mit den Satzmustern 3.1 und 3.20 dargestellt wird:

$$(x\,\varepsilon\,Q_1) \wedge (Q_1\,\Sigma\,Q_2) \rightarrow\ x\sigma(q_1\,\varepsilon\,Q_2) \vee x\sigma(q_2\,\varepsilon\,Q_2) \vee \ldots \vee x\sigma(q_n\,\varepsilon\,Q_2) \tag{3.27}$$

Eine Veränderung des Zustands von Prädikator Q_1 durch Eintreten eines Ereignisses (Geschehnisaussage $G(x)$ nach Satzmuster 3.9) bedeutet demnach eine Änderung des Attributwerts Q_2 von einem Apprädikator q_a, der den Ausgangszustand beschreibt, hin zu einem Apprädikator q_z, welcher den Zielzustand definiert. Normsprachliche Aussagen zu Zustandswechseln folgen daher dem Muster:

$$\forall x \varepsilon Q_1 (x\,\sigma(q_a\,\varepsilon\,Q_2) \wedge G(x)\ \rightarrow\ x\,\sigma(q_z\,\varepsilon\,Q_2)) \tag{3.28}$$

Konzept	Satzmuster	Nr.
Verhaltensaussagen	$[N \mid \pi \mid pP \mid q_1Q_1 \mid q_2Q_2^{Fall} \mid \ldots \mid q_nQ_n^{Fall}]$	3.9
Komposition	$[N_1 \mid \sigma_{Kompositionstyp} \mid N_2]$	3.11
Spezialisierung	$[N_1 \mid \varepsilon_{Inklusionstyp} \mid N_2]$	3.17
Attributierung	$[Q_1 \mid \Sigma \mid Q_2, \ldots Q_n]$	3.19
Merkmalsausprägung	$[N_1 \mid \sigma \mid (N_2 \mid \varepsilon \mid Q_1)]$	3.20
Berechtigungen	$[N_1 \mid \vartheta \mid P \mid N_2]$	3.22

Tabelle 3.1.: Zusammenfassung der in dieser Arbeit verwendeten Satzmuster

3.4. Zusammenfassung

Mit diesem Kapitel wurde eine terminologiebasierte Entwicklung von Softwarekomponenten für virtuelle Lehr- und Lernumgebungen beschrieben. Dafür wurde zunächst die terminologiebasierte Softwareentwicklung erläutert und Normsprachen als ein Konzept der Anforderungserhebung vorgestellt, das den Vorteil besitzt, unabhängig vom Entwicklungssystem formal gültige Aussagen formulieren zu können. Um statische Strukturen und Medienfunktionen von und in Lern- und Arbeitsszenarien grundlegend beschreiben zu können, wurde die Wissensraummetapher als konstruierender Teil von Lerntheorien und Theorien des Wissensmanagements eingeführt. Darauf aufbauend wurde eine Terminologie zur Beschreibung von Sachverhalten in der Domäne des kooperativen Lernens und Arbeitens entwickelt (Abb. 3.11, Abb. 3.12), sowie logische Regeln und Satzmuster zur Bildung formal gültiger Aussagen abgeleitet (Tab. 3.1). Damit ist der methodeninvariante Teil eines materialsprachlichen Softwareentwurfs umfassend behandelt.

Das hier erarbeitete terminologiebasierte Vorgehen zur Entwicklung softwaregestützter Lern- und Arbeitsszenarien bietet gegenüber herkömmlichen Softwareentwicklungsansätzen einige Erleichterungen: Anstatt ein neues semantisches Datenmodell des zu unterstützenden (Teil-)Problembereichs zu entwerfen, die grundsätzliche Datenstruktur dafür festzulegen und das so entstandene konzeptionelle Schema mit bereits bestehenden Schemata einer zu erweiternden Lern- oder Arbeitsplattform abzugleichen, wird das zu unterstützende Szenario normsprachlich rekonstruiert. Da die in diesem Kapitel vorgestellte Terminologie dazu die Bezeichner grundsätzlicher Elemente, Funktionen und Konzepte virtueller Wissensräume umfasst, können die normsprachlichen Aussagen mithilfe von CSCW-/L-Plattformen umgesetzt werden, die auf dieser Metapher basieren und entsprechende objektorientierte Programmierschnittstellen anbieten. Die Vorteile dieses Vorgehens sind daher vielschichtig:

- Den am Anfang des Kapitels vorgestellten sprachlichen Effekten und Defekten wird bei der Anforderungserhebung durch das definierte Vokabular und durch kontrollierte Satzmuster entgegen gewirkt. Dadurch kann sich die Kommunikation in heterogenen Projektteams verbessern und Anforderungen an virtuellen Lern- und Arbeitsszenarien können eindeutiger formuliert werden.

- Anstatt ein neuartiges Begriffs- respektive Objektmodell für einen Problembereich neu konstruieren zu müssen, können – mit deutlich weniger Aufwand – neue

Konzepte auf Basis des durch die Terminologie beschriebenen Domänenmodells rekonstruiert werden. Das reduziert nicht nur den Aufwand für den Fachentwurf. Eine Wiederverwendung von Konzepten auf Entwurfsebene sorgt darüber hinaus auch für eine einheitliche und stabile Gesamtarchitektur (vgl. [Fra94], S. 28).

- Die Persistierung von Objektstrukturen und -zuständen sowie der lesende und schreibende Zugriff auf Objektdaten wird bei diesem Vorgehen durch ein CSCW/L-Objektrahmenwerk übernommen. Durch die Berücksichtigung der Wissensraummetapher in der Terminologie ist das semantische Datenmodell der zu unterstützenden Szenarien also nicht mehr länger ein fester Bestandteil der Anwendungsentwicklung. Anwendungen können i. d. R. entwickelt, gewartet und erweitert werden, ohne Datenstrukturen oder Persistierungsmethoden verändern zu müssen[45].

- Grundsätzlich ist eine terminologiebasierte Entwicklung auf Basis der hier vorgestellten Normsprache zwar auf objektorientierte Konzepte in der Zielsprache (Programmiersprache) angewiesen, ansonsten jedoch neutral bzgl. der zur Entwicklung eingesetzten Technologie. Bspw. können sowohl klientenseitige Desktop-Anwendungen als auch serverseitige Web-Anwendungen in OO-Programmiersprachen oder -Skriptsprachen implementiert werden.

- Durch die Technisierung der Wissensraummetapher über objektorientierte Klassenbibliotheken stehen in der Zielsprache i. d. R. die gleichen Konzepte wie in der Normsprache zur Verfügung (vgl. [HR05]). Dies erlaubt einerseits das leichtere Programmieren auf einem höheren Abstraktionsgrad, andererseits die fachgerechte Realisierung von Lernszenarien, da es keinen Bruch in den verwendeten Konzepten gibt.

[45] Dazu Coad in [Coa91]: „Why not buy a set of OOA (Object-Oriented Analysis)/OOD (Object-Oriented Design) diagrams and a repository with an initial set of business rules, attributes, and services? Customizing could then be done at the specification level or the design level and the organization could then generate unique code for its needs“.

4. Eine Rahmenarchitektur für universitäres E-Learning

Unter Berücksichtigung der in Kapitel 2 beschriebenen Herausforderungen eines integrierten Informationsmanagements an Hochschulen und dem bereits in Kapitel 3 vorgestellten normsprachlichen Ansatz zur Spezifizierung von Lernszenarien ist im Folgenden als erklärtes zweites Ziel der Arbeit die Spezifikation einer Rahmenarchitektur für komponentenbasierte Lern- und Arbeitsumgebungen zu erarbeiten. Dazu sollen zum einen allgemeine Entwurfsprinzipien von Komponentenarchitekturen herausgearbeitet, zum anderen der statische Teil der Rahmenarchitektur durch Art, Struktur und Zusammenwirken der Softwarebausteine beschrieben werden. Von der fachlichen Funktionalität und der Datenarchitektur wird abstrahiert und eine Abbildung der Softwarekomponenten auf eine IT-Basisinfrastruktur nur im Rahmen einer möglichen Komponentenverteilung diskutiert. Durch diese Abstraktion soll die konzipierte Rahmenarchitektur den Anwendungsbereich verteilter universitärer Lern- und Arbeitsumgebungen möglichst generisch wiedergeben, um einen Leitliniencharakter zu gewährleisten.

Wie in Kapitel 1 erörtert, soll bei der Konzeption die Loslösung von der an einzelnen Systemen orientierten Herangehensweise in der Softwareentwicklung im Vordergrund stehen. Aus diesem Grund wird eine architekturzentrierte Konzeption angestrebt, bei der es darum geht, eine einheitliche Plattform zu schaffen, die für verschiedenste Anwendungen in der Domäne des universitären E-Learnings eine geeignete Basis darstellt.

Diese Sichtweise impliziert eine separate Betrachtung von einer Produktstandardarchitektur[1] mit gut wiederverwendbaren Softwarekomponenten und den darauf basierenden Produktarchitekturen, welche die Standardarchitektur um anwendungsspezifische und daher weniger gut wiederverwendbare Komponenten erweitern.

Die proaktive Wiederverwendung von Komponenten soll sich auch im dynamischen Teil der Rahmenarchitektur wieder finden, die nach Dern neben den Entwurfsprinzipien das Entwicklungsmodell einer Anwendungsarchitektur definiert (vgl. [Der03], S. 19ff.). Dazu wird eine Softwareproduktlinien-orientierte Vorgehensweise beschrieben, die in das Prozessmodell der in Kapitel 6 zu konzipierenden Methodik mit einfließt.

In diesem Kapitel werden zunächst grundlegende technische Konzepte und Prinzipien zur Umsetzung und zum Aufbau von komponentenbasierten IT-Architekturen beschrieben (4.1). Darauf basierend folgt in Abschnitt 4.2 eine Konstruktion der Rahmenarchitektur durch die Identifikation notwendiger Dienste im Kontext eines Schichtmodells, sowie die Abgrenzung von Standard- und Produktarchitektur. Weitere Aspekte, nämlich die Verteilung der Komponenten auf Laufzeitumgebungen, sowie die aus verschiedenen Einsatz-

[1] Im weiteren Verlauf dieser Arbeit wird die Produktstandardarchitektur auch als „Plattform“ bezeichnet.

szenarien abgeleiteten Erweiterungen ergänzen die Architektur. Das Kapitel schließt mit einer Beschreibung der Anwendungsmöglichkeit eines Softwareproduktlinien-orientierten Entwicklungsmodells auf die konzipierte Rahmenarchitektur (4.3).

4.1. Komponentenarchitekturen

Bei der Entwicklung integrierter virtueller Lernumgebungen im universitären Bereich kann grundsätzlich nach verschiedenen Strategien vorgegangen werden (in Anlehnung an [Ort00] und [Bru04]):

- Es wird eine Standardsoftware verwendet, die durch *Customizing* an didaktische und organisatorische Anforderungen angepasst werden kann[2]
- Es wird eine Standardsoftware ausgewählt und das didaktische und organisatorische Vorgehen wird an die Software angepasst
- Die Entwicklung der Lernumgebung erfolgt auf Basis eines Frameworks. Dazu werden vorgefertigte Module des Frameworks konfiguriert und zu einem Gesamtsystem zusammengefasst[3]
- Spezialisierte Fach-Komponenten werden innerhalb einer Softwarearchitektur miteinander kombiniert, um individuelle neue Anwendungslösungen oder neue Komponenten umzusetzen (vgl. [RH05]).

Wurde die letztgenannte Strategie in den vorhergehenden Kapiteln bislang nur mit dem Argument eines integrierten Informationsmanagements motiviert, lässt sich an dieser Stelle im direkten Vergleich alternativer Strategien noch anmerken, dass nur unter dieser Strategie hochschulinterne Eigenentwicklungen integrierter E-Learning-Werkzeuge, die für einen hochschulweiten Einsatz gedacht sind, ökonomisch vertretbar sind: Mit einer komponentenorientierten Architektur können Entwicklungskosten durch die Wiederverwendung bereits etablierter Dienste eingespart werden. Vor allem jedoch Kosten für Wartung und Weiterentwicklung, wenn proaktiv Integration und Modularisierung betrieben wird.

Im Folgenden sollen Komponentenarchitekturen näher ausgeführt werden, um damit ein grundlegendes Verständnis für die zu konzipierende Rahmenarchitektur zu schaffen. Allgemeine Entwurfsprinzipien und Hinweise zur hochverfügbaren Anordnung von Komponenten zählen zum Entwicklungsmodell und somit zum dynamischen Teil der Rahmenarchitektur, und werden daher ebenfalls behandelt.

[2] Standard-Plattformen gibt es im E-Learning sowohl im Open Source-Umfeld (bspw. moodle, Ilias, openUSS) als auch auf dem kommerziellen Markt (bspw. Clix, Blackboard/WebCT). Vgl. dazu [GL05] für eine Untersuchung der Adaptierbarkeit von Open Source-Lernplattformen.

[3] Kerres beschreibt diese Art der Entwicklung konfigurierbarer Lern- und Arbeitsumgebungen beispielhaft in [Ker06] anhand des Content Management Systems Drupal (`http://www.drupal.org/`).

4.1.1. Begriffsdefinitionen

4.1.1.1. Der Architekturbegriff

Der Begriff der Architektur wird in der IT immer dann verwendet, wenn die Struktur und das Zusammenwirken von Informationssystemen oder einzelnen Hardware-/Softwarekomponenten beschrieben werden soll. Eine „IT-Architektur kann [...] als Struktur bzw. Zusammenhang zwischen diversen Komponenten verstanden werden“ [Sch04c], S. 8. Bass et al. konkretisieren diese Aussage: „The architectural view of a system is abstract, distilling away details of implementation, algorithm, and data representation and concentrating on the behavior and interaction of “black box” elements“ [BCK03], S. 3.

Andere Definitionen implizieren darüber hinaus auch die der Architektur zugrunde liegenden Prinzipien, nach denen Entwurfsentscheidungen getroffen werden: Nach Masak ist eine Architektur „[...] eine formale Beschreibung eines Systems, ein detaillierter Plan des Systems und seiner Komponenten, die Struktur der Komponenten, ihre Wechselwirkungen, ihre Prinzipien und Richtlinien, die ihren Entwurf, ihre Entwicklung und Implementierung steuern“ [Mas05], S. 9. Das IEEE hat mit dem ANSI/IEEE-Standard 1471-2000 eine Handlungsempfehlung für das Beschreiben von Architekturen auf dem Gebiet der softwareintensiven Systeme[4] erarbeitet. Dafür wird eine Architektur als „[...] the fundamental organization of a system embodied in its components, their relationships to each others and to the environment and the principles guiding its design and evolution“ definiert (s. [S$^+$00], S. 9). Die Verwendung des Ausdrucks *fundamental organization* beschreibt in dieser Begriffsdefinition grundlegende, einheitliche Prinzipien und Konzepte, während *system* repräsentativ für Applikationen, Plattformen oder Unternehmen selbst verwendet wird. Als Architekturumgebung (*environment*) wird hingegen der entwicklungsgemäße, operative oder programmatische Kontext des Systems verstanden (vgl. [Hil00], S. 10).

Die Definitionen lassen erkennen, dass es sich bei Architekturen im Bereich der softwareintensiven Systeme um komplexe Gebilde handelt, welche die Interaktionen ihrer Teilsysteme und die Teilsysteme selbst (im Weiteren als Komponenten bezeichnet) abstrakt beschreiben und organisieren. Weiterhin bietet eine Architektur Prinzipien zur Weiterentwicklung und Gestaltung der Architektur selbst (vgl. S. 123ff.)[5].

Eine *Rahmenarchitektur* beschreibt ein idealtypisches Muster für eine bestimmte Klasse von Architekturen. Auf Basis dieses allgemeinen Musters können speziellere Modelle mit geringeren Kosten konstruiert werden. Darüber hinaus kann eine Rahmenarchitektur auch als Vergleichsobjekt für andere Modelle dienen, die den gleichen Sachverhalt beschreiben. Eine Abgrenzung zu abstrakteren Rahmenarchitekturen und zum Softwaredesign wurde bereits auf S. 43ff. vorgenommen.

[4] Sämtliche Systeme, bei denen Software essentiellen Einfluss auf Design, Konstruktion, Einsatz und Evolution des Systems in seiner Gesamtheit ausübt (vgl. [LMW05], S. 47).

[5] Dern spricht daher in seiner Definition einer Softwarearchitektur diesbzgl. von einem statischen (Struktur) und einem dynamischen Teil (Entwicklungssystem, vgl. [Der03], S. 18). In dieser Arbeit wird unter dem Architekturbegriff der von ihm als statisch deklarierte Teil verstanden.

4.1.1.2. Der Komponentenbegriff

Vor dem Hintergrund einer komponentenbasierten Architektur kapseln Komponenten logisch zusammenhängende Funktionen eines Systems, um die Wiederverwendbarkeit und den Gesamtüberblick über die Systemstruktur zu erleichtern. Vor allem in der Entwicklung von verteilten Systemen wird auf das Konzept der komponentenbasierten Anwendungsentwicklung zurückgegriffen, um sie an verschiedenen Standorten ablegen und mit einer einheitlichen Zugriffsschnittstelle ausstatten zu können (vgl. [RHQ+04], S. 144). Zudem kann ein bestehendes System durch den Austausch einer Komponente in seiner Funktionalität erweitert werden, da das Interface zur Kommunikation mit anderen Teilen des Systems nicht oder nur wenig modifiziert wird (vgl. [RHQ+04], S. 4).

Obwohl das Konzept komponentenorientierter Softwarearchitekturen nunmehr über 30 Jahre alt ist und entsprechende Technologien wie COM+ und Enterprise Java Beans (EJB) bereits seit mehreren Jahren produktiv in der Softwareentwicklung eingesetzt werden, ist eine einheitliche Definition des Komponentenbegriffs in der Literatur nicht existent (vgl. [Tur99]). Eine vielzitierte Definition einer Komponente wurde von Szyperski und Pfister gegeben (vgl. [SP97]): „A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independendly and is subject to composition by third parties".

Andere Definitionen des Komponentenbegriffs betonen neben der kompositorischen Wiederverwendung noch das Kriterium der Abgeschlossenheit, wenn ihr ihre Bestandteile[6] eindeutig zuordenbar sind, so dass die Komponente als Ganzes klar von anderen Komponenten abgegrenzt werden kann. Idealerweise ist eine klare Trennung der Komponenten deswegen angestrebt, um diese einzeln entwickeln und später zusammenfügen zu können (vgl. [JN05], S. 6f.)[7].

In dieser Arbeit wird unter einer Komponente demnach ein modularer Architekturteil verstanden, der seinen internen kohärenten Funktionsumfang transparent kapselt und so vor dem Nutzer verbirgt. Die darin eingeschlossenen Artefakte bieten im Ganzen eine klar definierte Funktionalität an[8]. Durch ihre Abgeschlossenheit ist eine Komponente ein eigenständiges System. Dabei kann der Funktionsumfang jedoch auch auf der Nutzung von anderen Komponenten basieren. Abbildung 4.1 stellt die Eigenschaften einer Komponente in der *Unified Modelling Language* (UML) dar.

[6] Nach Ackermann et al. zählen zu diesen (Software-)Artefakten der ausführbare Code, darin verankerte Grafiken, Textkonstanten usw., die einen initialen Zustand der Komponente beschreibenden Daten, z. B. Voreinstellungen oder Parameter, sowie Spezifikation, (Anwender-) Dokumentation und (automatisierte) Tests (vgl. [A+02], S. 1). Dazu Rupp in [RHQ+04], S. 144.: „Auf der technischen Seite besteht eine Komponente aus einer Reihe von interagierenden Klassen, die gemeinsam Aufgaben erfüllen und diese nach außen durch eine klar definierte Schnittstelle anbieten".

[7] Eine klare Abgrenzung ist in der Praxis jedoch schwierig, wenn ein Problembereich sich nur schwer oder überhaupt nicht separieren lässt (*crosscutting concerns*), bspw. Instanz-, Logging-, Distributions- und Transaktions-Management (vgl. [JN05]).

[8] Von einer *Fachkomponente* spricht man, wenn eine Komponente eine bestimmte Menge von Funktionen einer bestimmten fachlichen Anwendungsdomäne anbietet (vgl. [A+02], S. 1). Dazu Herzum in [HS00]: „A business component is the software implementation of an autonomous business concept or business process." Andresen in [And03] , S. 18: „Das Wissen einer Software-Komponente repräsentiert ein Konzept eines Geschäftsfeldes".

Abbildung 4.1.: Eine Fachkomponente in UML-Notation

Karlsson klassifiziert in [Kar95], S.12ff. wiederbenutzbare Komponenten nach drei Kriterien, welche im weiteren Verlauf dieses Kapitels zur Beschreibung der Komponentenarten verwendet werden:

Umfang der Wiederverwendbarkeit Zu unterscheiden sind dabei die Kategorien „general“, „domain“ und „product-line“. Generische Komponenten sind bspw. Schaltflächen für Benutzungsschnittstellen. Der Kategorie „domain“ sind Komponenten zuzuordnen, die spezifisch für eine Anwendungsdomäne sind und die Kategorie „product-line“ umfasst Komponenten, die sich auf eine Produktlinie beziehen.

Ziel der Wiederverwendung Dieses Kriterium unterscheidet, ob eine Komponente für den internen Gebrauch oder (auch) für die externe Vermarktung gedacht ist.

Komponentengranularität Das Kriterium gibt an, ob eine Komponente eine geringe oder hohe Granularität aufweist.

Komponenten sind immer Teil einer Softwarearchitektur, welche eine Laufzeitumgebung für Instanzen dieser Komponenten bietet und Interaktionen sowohl zwischen ihnen und Instanzen anderer Komponenten regelt als auch mit der Infrastruktur selbst. Daher implementieren Komponenten von der Softwarearchitektur vorgegebenen Schnittstellen bzw. genügen einem bestimmten Komponentenmodell.

4.1.1.3. Eine serviceorientierte Architektur (SOA)

Serviceorientierte Architekturen stellen eine Fortführung der komponentenorientierten Architekturen dar (vgl. [Tur99], [Ort00], [Sch01a], [And03][9]).

In diesem Kontext wird das Konzept des Dienstes (*Web-Service*) eingeführt, das durch folgende Eigenschaften charakterisiert ist: (1) Ein Web-Service kann mit seiner Dienstbeschreibung bei einem Dienstverzeichnis (*Registry*) registriert werden und ist darüber lokalisierbar. (2) Web-Services unterstützen lose gekoppelte Verbindungen. Das bedeutet, dass Dienste durch andere Dienste mit gleicher Funktionalität ersetzt werden können. Gleichzeitig können gleichartige Dienste mehrfach gestartet werden. Das erhöht die Skalierbarkeit ebenso wie die Ausfallsicherheit. Dienste kapseln (analog zu den Komponenten) also bestimmte Funktionen in einer internen Struktur und bieten nach außen wohl definierte Schnittstellen an, welche über das Netzwerk angesprochen werden können. Durch die klar definierten Schnittstellen der Services wird eine maximale Entkopplung erreicht, die externe Einflüsse so stark verringert, dass Dienste selbst zur Laufzeit dynamisch hinzugebunden oder ausgetauscht werden können.

Anwendungen funktionieren in SOAs als Zusammensetzung von verschiedenen Diensten, die unabhängig von der Anwendung und den IT-Plattformen, auf denen sie laufen, eingesetzt werden können. Da die einzelnen Services als separate Bausteine verfügbar sind, können diese auf verschiedene Art und Weise gruppiert und integriert werden, um so völlig neue Funktionalität zu erstellen. Sie sollen es ermöglichen, jegliche softwaretechnische Anforderung in kürzester Zeit in bestehende Systemlandschaften zu integrieren.

Nach Sommerville können Web-Services durch Komponenten auf unterschiedlichen Abstraktionsebenen implementiert werden: Eine Komponente kann im Extremfall als vollständige Systemabstraktion auftreten und somit ein in sich geschlossenes System mit einer Vielzahl an Diensten repräsentieren. Eine Komponente kann aber auch nur eine einzelne Funktion implementieren, somit würde ein Dienst gleichzeitig eine Komponente repräsentieren (vgl. [Som07], S. 321). Abbildung 4.2 stellt diese Zusammenhänge in einem Metamodell dar.

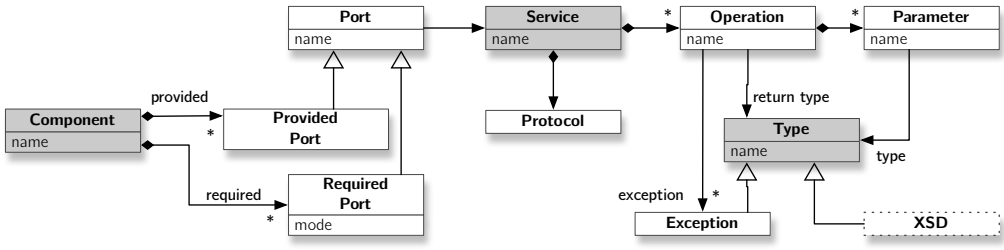

Abbildung 4.2.: Metamodell für Dienste und Komponenten (in Anlehnung an [Völ06]).

[9] Darüber hinaus beschreiben Roth und Hampel in [RH05] die Umsetzung von E-Learning-Software als komponentenorientierte Anwendung.

Dienste sind ein wesentlicher Bestandteil bei der modernen Architekturentwicklung, da sie zunächst unabhängig einer bestimmten Programmiersprache und Plattform von der tatsächlichen Implementierung von Funktionalität abstrahieren und somit im Gegensatz zu rein komponentenorientierten Architekturen flexibler einsetzbar sind (vgl. [Bec03], S. 4f.; vgl. auch [Tur03],S. 3ff.). Als Kommunikationsprotokoll zum Nachrichtenaustausch zwischen Diensten können Basisprotokolle wie bspw. HTTP oder SMTP genutzt werden. Die Nutzung der Dienste erfordert darüber hinausgehend drei Standards: (1) zur Beschreibung von Diensten, (2) zum Suchen und Finden von Diensten in verteilten Umgebungen und (3) zur Übermittlung von Parametern bzw. Ergebnissen:

WSDL Die *Web Service Description Language* (WSDL) ist eine Metasprache, mit der Dienste XML-basierend beschrieben werden können, insbesondere welche Funktionalitäten der Web-Service über welche Objekte mit welchen Parametern bereitstellt[10]. Neben diesen Strukturen und Datentypen definiert eine solche Dienstbeschreibung vor allem das Mapping zwischen den Parametern und den zugehörigen SOAP-Datentypen sowie das Binding an unterliegende Protokolle.

UDDI Mit Hilfe der *Universal Description, Discovery and Integration* (UDDI) als eine Art Verzeichnis können Dienste auf einheitliche Weise beschrieben, aufgefunden und integriert werden. Informationen im UDDI-Verzeichnis können auf verschiedene Arten abgerufen werden. Das Verzeichnis ist dazu dreigeteilt: Die *White Pages* enthalten Adressen und Kontaktinformationen eines Dienstanbieters. Die *Yellow Pages* werden benutzt, um Dienstanbieter zu klassifizieren und zu kategorisieren. Auf diese Weise entsteht eine Art Branchenbuch. Schließlich enthalten die *Green-Pages* noch technische Beschreibungen zu den angebotenen Web-Services. Der Zugriff auf das Verzeichnis erfolgt über eine SOAP-Schnittstelle. Je mehr Web-Services existieren und für verschiedene Zwecke genutzt werden, desto interessanter wird der Gedanke, UDDI einzusetzen (vgl. [Kla04], S. 4).

SOAP Das vormals genannte *Simple Object Access Protocol* definiert ein Rahmenwerk[11], mit dem Dienste untereinander Nachrichten austauschen können. Dazu werden die auszutauschenden Daten als SOAP-Nachricht in einen so genannten SOAP-Umschlag (*Envelope*) verpackt, in dessen *Header*-Element Meta-Informationen zum Routing, zur Verschlüsselung oder zur Transaktionsidentifizierung untergebracht werden können. Der Transport der Nachricht basiert über o. g. Basisprotokolle wie HTTP und ist dadurch auch für den Einsatz in stärker gesicherten Netzwerken geeignet, in denen *Firewalls* nur Ports für Basisdienste durchlässig gestalten. Nicht so umfangreiche und dadurch leichtgewichtigere Alternativen zum SOAP-Standard sind XML-RPC und REST (*Representational State Transfer*)[12].

[10] Ein Vorteil bei WSDL ist die Lesbarkeit beim Entwickler während der Entwicklungsphase durch die klare Strukturierung des XML-Formats.

[11] Das SOAP-Rahmenwerk umfasst Regeln für das Nachrichtendesign, Regeln zur Abbildung und Interpretation von Daten, sowie Regeln für entfernte Prozessaufrufe mittels SOAP.

[12] Weitere Informationen zu XML-RPC finden sich im Internet unter `http://www.xmlrpc.com/`. Letzter Zugriff: 31.07.2008.Eine ausführliche Beschreibung zu REST findet sich in der Dissertation von Fielding (vgl. [Fie00]).

Abbildung 4.3 veranschaulicht das Zusammenspiel der drei genannten Web-Service-Standards bei entfernten Funktionsaufrufen. Ob der Einsatz von Web-Services sinnvoll ist, muss im Einzelfall geprüft werden (vgl. [Wie03], S. 22f.). Derzeitig fehlende Standards bei den so genannten *Web-Services Add-Ons*, also Erweiterungen bspw. zur Integrität, Transaktionssicherheit oder Authentifizierung, können dagegen sprechen. Vor allem bedingt die Nutzung von Web-Services in stark gekoppelten komponentenorientierten Architekturen einen hohen Overhead an Übertragungsvolumen und Rechenleistung, insbesondere bei wenig Austauschdaten[13]. Aufgrund des flexiblen Aufbaus können jedoch komplexere Datenstrukturen in einer SOAP-Nachricht ausgetauscht werden, wohingegen in stark gekoppelten Systemen mehrere Anfragen nötig wären. In diesen Fällen würde ein besseres Nutzlastverhältnis und ein verhältnismäßig geringerer Kommunikationsaufwand erzielt werden.

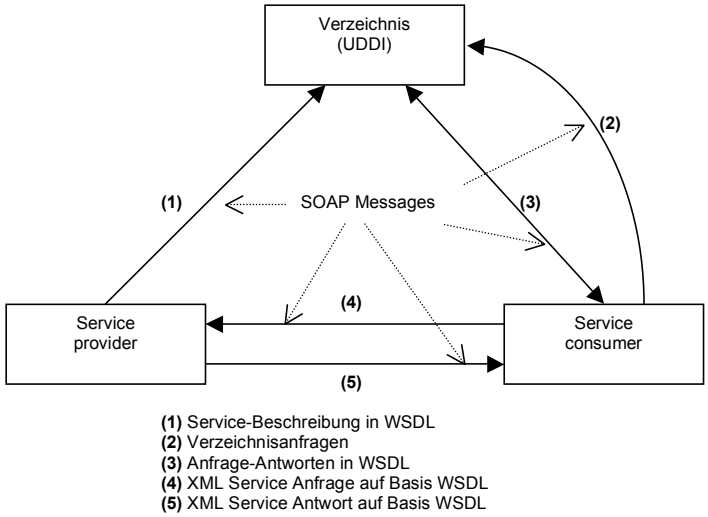

Abbildung 4.3.: Funktionsweise von Web-Services (eigene Darstellung nach [Nag03], S. 7).

4.1.2. Verteilte Informationsverarbeitung auf Basis von Middleware

Bei der kombinierten Nutzung verschiedener Komponenten einer Architektur können zwei Kommunikationsparadigmen unterschieden werden, die den Verhaltensablauf der Interaktion beschreiben und entscheidend für die Flexibilität der Komponenten-Schnittstellen sind: Bei der *synchronen Kommunikation* muss der Sender einer Nachricht auf die Antwort des Empfängers warten, um mit der Verarbeitung fortzufahren. Im Gegensatz dazu wird die Kommunikation zwischen Applikationen als *asynchron* bezeichnet, wenn

[13] Beispiel: Zur Versendung von Wahr/Falsch-Werten muss zunächst ein XML-Dokument gebaut und validiert werden. Durch die für den Versand benötigten Metadaten werden oftmals mehrere hundert Bytes benötigt. In stark gekoppelten Systemen reicht für diese Information ein einziges Bit aus.

der Sender nach Versand einer Nachricht direkt mit der Verarbeitung fortfahren kann, während der Empfänger die Nachricht entweder sofort verarbeitet, oder in einem Puffer vorhält und erst bei Bedarf die Verarbeitung vornimmt. Synchrone und asynchrone Kommunikation schließen sich nicht gegenseitig aus, in der Regel sieht eine wohldefinierte Informationsarchitektur die Verwendung beider Paradigmen vor.

Innerhalb der funktionalen Integration von Komponenten kann man zwischen verschiedenen Ansätze unterscheiden, die auf einem bestimmten Kommunikationsparadigma beruhen und den Punkt des Zugriffs auf einen zu integrierenden Baustein definieren: (1) Integration über entfernte Funktionsaufrufe, (2) Integration über verteilte Objekte und (3) synchrone und asynchrone nachrichtenbasierte Kommunikation (vgl. [OWZ+05], S. 43ff., [RHG05], S. 70f.).

4.1.2.1. Remote Procedure Calls

Die Integration über entfernte Funktionsaufrufe (sog. *Remote Procedure Calls*, RPCs) stellt die älteste Form der funktionalen Integration dar und basiert dabei auf dem synchronen Kommunikationsparadigma. Dazu ruft eine Komponente in einem entfernten System/einer anderen Komponente eine Funktion auf und übergibt dabei die benötigten Daten als Aufrufargument. Die Anzahl der Argumente sowie die verwendeten Datentypen werden durch die Signatur der Funktion definiert[14]. Die Funktion wird meist vom Empfänger als Teil einer Schnittstelle (*Application Programming Interface*, API) zur Verfügung gestellt. RPCs erlauben somit einer Komponente, Funktionen entfernter Systeme für die Bewältigung der eigenen Aufgaben in Anspruch zu nehmen. Auf dem gleichen Weg können allerdings auch gänzlich neue Softwarebausteine geschaffen werden. Hierzu werden mehrere entfernte Funktionsaufrufe durch eine anwendungsübergreifende Funktion zusammengefasst, die eigenständig aufgerufen werden kann.

4.1.2.2. Verteilte Objekte

RPCs stellen bei verteilten Objekten die Basistechnologie dar, um über standardisierte Schnittstellen und Protokolle miteinander zu kommunizieren. Bei solchen objektorientierten Zugriffen auf Fremdanwendungen kann eine Kommunikation zwischen Systemen nur dann stattfinden, wenn sie eine gemeinsame verteilte Objektschnittstellenarchitektur unterstützen. In diesem Bereich sind die Standardisierungen durch das *Component Object Model* (COM) und die *Common Object Request Broker Architecture* (CORBA) am häufigsten anzutreffen. Auch hier ist eine enge Bindung der integrierten Systeme gegeben, da sowohl das Objektmodell als auch die Objektschnittstellenarchitektur gemeinsam verwendet werden müssen.

[14] Schnittstellenbasierte Kommunikation bindet dadurch die aufrufende Komponente stärker an das entfernte System. Wird die Signatur der Funktion in der angebotenen Schnittstelle geändert, muss die aufrufende Komponente ebenfalls angepasst werden, was zu erheblichen Aufwänden führen kann (vgl. [OWZ+05], S. 65). Außerdem ist die aufrufende Komponente auf die Verfügbarkeit des entfernten Systems angewiesen.

4.1.2.3. Message oriented Middleware

Nachrichten-orientierte Middleware (*Message oriented Middleware*, MOM) erfordert im Gegensatz zu den RPCs keine enge Bindung zwischen integrierten Systemen, da Nachrichtendienste die Übermittlung von kleinen Datensätzen vornehmen. Somit braucht die rufende Komponente nicht blockiert werden, während sie auf das Ergebnis des Funktionsaufrufs wartet. Mit der Interprozesskommunikation (IPC) und dem *Message Queuing* (MQ) können zwei Kommunikationsparadigmen unterschieden werden. Bei IPC müssen zum Nachrichtentausch beide Prozesse gleichzeitig aktiv sein, während beim Message-Queue-Modell vor allem Warteschlangen (*Message Queues*) zum Einsatz kommen, die Nachrichten speichern und bei Bedarf oder Verfügbarkeit weiterleiten[15]. Message Broker erweitern das Konzept der Nachrichten-orientierten Übermittlung von Informationen, indem sie einen zentralen Server als Steuerungseinheit bereitstellen, der zusätzlich zur Kommunikation auch eine Transformation von Nachrichten vornehmen kann. Darüber hinaus existieren noch Transaktionsmonitore und Applikationsserver, die zur Transaktions-orientierten Middleware zählen. Beide Softwarekomponenten beziehen sich auf Transaktionen, also abgeschlossene Aufgabeneinheiten mit definiertem Beginn und Ende. Die abzuarbeitenden Aufgaben innerhalb einer Transaktion können unterschiedlichster Natur sein und beispielsweise Applikations- und Datenbankzugriffe, das Absetzen von Nachrichten oder den Aufruf verteilter Objekte umfassen.

Bezogen auf eine universitäre Informationsarchitektur können zum Beispiel durch eine solche Funktionsintegration die Informationen eines neu angelegten Kurses im Administrationssystem über eine Warteschlange automatisch an Lernplattformen weitergeleitet werden, wenn diese Nachrichten vom Typ „Kurs angelegt“ abonniert haben. Die Übermittlung ist dabei eine Folge von Funktionsaufrufen. Middleware-Produkte können hierbei die Steuerung übernehmen. Die Vor- und Nachteile einer Funktionsintegration sind in Tabelle 4.1 zusammengefasst.

Vorteile	**Nachteile**
❏ Weites Spektrum lösbarer Integrationsprobleme (einschließlich Präsentations- und Datenintegration)	❏ Relativ hohe Komplexität, Lernkurve
❏ Hohe Wiederverwendbarkeit der Softwarekomponenten	❏ Modifikation der Anwendungen notwendig
	❏ Zugriff auf Logik kann schwierig sein, wenn Informationen aus dem Quellcode oder der API Spezifikation fehlen

Tabelle 4.1.: Vor- und Nachteile einer Funktionsintegration nach [Kai01].

[15] Beispiele für Message Queuing Produkte sind IBM MQ Series, Microsoft MSMQ oder Tibco Rendevous.

4.1.3. Entwurfsprinzipien von Komponentenarchitekturen

Prinzipien beschreiben Leitkriterien, denen ein Entwurf folgen sollte. Sie bilden „den gemeinsamen, verbindlichen Hintergrund, den die Zerlegung und Aufteilung in Einzelkomponenten benötigt“ [BV03], S. 42. Im Gegensatz zu Entwurfsmustern beschreiben Prinzipien nicht die betroffenen Komponenten oder deren Gestaltung in einem Systemkonzept, sondern legen grundlegende Eigenschaften einer Architektur fest. Ein Entwurfsmuster kann auf verschiedenen Prinzipien beruhen. Auch Heuristiken sind ein Beispiel für Prinzipien, die sich in der Praxis bewährt haben und als generelle Richtlinien in die Architekturplanung einfließen (vgl. [Ham05b], S. 72). Ein bekanntes Beispiel ist der Architekturgrundsatz *Separation of Concerns* (Trennung der Zuständigkeiten) von Dijkstra. Der Grundsatz genießt in der Softwareentwicklung besondere Bedeutung, kann aber auch auf die IT-Architektur angewandt werden. Er besagt, dass ein System so strukturiert sein sollte, dass Zuständigkeiten innerhalb des Gesamtsystems separiert und eindeutigen Teilsystemen oder Komponenten zugeordnet werden können (vgl. ebd.). In der Systemarchitektur legt dieses Prinzip den Grundstein für den Aufbau verteilter IT-Infrastrukturen.

Der Entwurf und die Verteilung von Komponentenarchitekturen beruht in Anlehnung an Dyson and Longshaw auf vier wesentlichen Prinzipien (vgl. [DL04], S. 91ff.): Redundanz (*Redundancy*), Funktionale Identität (*Functionally-Identical Elements*), einfach gerichtete Abhängigkeiten (*One-Way Dependencies*) und Standardprotokolle (*Standard Protocols*). Diese sollen im Folgenden erklärt werden.

4.1.3.1. Redundanz

Das Prinzip der Redundanz lässt sich recht einfach beschreiben: „Add more capacity than you normally need, then use this capacity in abnormal situations such as when things fail or when there is more load on the system as usual“ [DL04], S. 92. Die Ziele, die mit dem Prinzip der Redundanz verfolgt werden, sind die Erhöhung der Verfügbarkeit und die Sicherstellung einer ausreichenden Performance des Systems. Redundanz kann in einer Architektur in zwei unterschiedlichen Erscheinungsformen auftreten: Vervielfältigung (*duplication*) und/oder durch Überkapazität (*over-capacity*). Durch Vervielfältigung einzelner Komponenten wird in einem Fehlerfall die Fortführung des Dienstes durch ein anderes Element garantiert, sodass ein Ausfall nicht bemerkbar ist. Im extremsten Fall wird das ganze System dupliziert[16]. Überkapazitäten dienen vornehmlich der Abschwächung von so genannten Lastspitzen, die ein System in Gefahr bringen können. Üblicherweise werden Überkapazitäten in Form von Sicherheitsfaktoren bei der Bedarfsermittlung von Ressourcen berücksichtigt, z. B. bei der Bestimmung des Hauptspeicherbedarfs für einen Datenbankserver oder der benötigten Bandbreite einer Internet-Verbindung. Sie dienen auch als Absicherung späterer Ausbaustufen und Erweiterungen. Die Entscheidung für die Umsetzung einer Redundanz ist für jeden Einzelfall durch eine Kosten-Nutzen-Analyse zu bewerten. „By definition, the extra capacity introduced [by redundancy] won't normally be required but does need to be paid for“ [DL04], S. 92.

[16] Die Ermittlung der Komponenten in einem System, die redundant ausgelegt werden sollten, wird durch eine Schwachstellenanalyse (*single point of failure analysis*) durchgeführt.

4.1.3.2. Funktionale Identität

Das Prinzip der Funktionalen Identität besagt, dass duplizierte Komponenten den gleichen Umfang an Funktionen bereitstellen müssen. Dies bedeutet jedoch nicht, dass auch ihre nicht-funktionalen Eigenschaften (Verfügbarkeit, Performance, etc.) identisch ausgeprägt seien müssen. Aus Kostengründen erscheint es in vielen Fällen nicht sinnvoll, gerade teure Spezialkomponenten mehrfach auszulegen. Hier würde es sich anbieten, für den während eines Ausfalls überschaubaren Zeitraum eine kostengünstigere Komponente mit einer geringeren garantierten Verfügbarkeit einzusetzen. Es gibt jedoch Situationen, in denen bei einem Ausfall in jedem Fall identische Komponenten bereitstehen müssen, da nicht garantiert werden kann, wie lange die Wiederherstellung des ursprünglichen Systems andauern wird und während dieser Zeit eine gleichwertige Dienstqualität sichergestellt werden muss. Dies gilt im Allgemeinen für geschäftskritische Anwendungen. Die Funktionale Integrität bietet damit gewisse Freiheitsgrade beim Architekturdesign. Hierbei gilt es jedoch zu berücksichtigen: „When we add redundancy through the use of duplications we need to be sure that the duplicate elements are all functional identical, but we have the option of varying the non-functional characteristics based on the circumstances“ [DL04], S. 92.

4.1.3.3. Einfach gerichtete Abhängigkeiten

In einer verteilten Architektur sind es keineswegs einzelne Elemente, mit denen nichtfunktionale Anforderungen wie Skalierbarkeit, Flexibilität etc. erfüllt werden, sondern das systematische Zusammenspiel mehrerer dieser Komponenten. Jedoch kann gerade das Zusammenwirken unterschiedlicher Elemente in einer Architektur die Weiterentwicklung des Gesamtsystems hemmen (vgl. [DL04], S. 210). Die enge Kopplung zwischen den Architekturelementen erschwert die Durchführung von notwendig gewordenen Änderungen an einer einzelnen Komponente. Die Abhängigkeiten zu anderen Komponenten sind oft so vielzählig, dass Veränderungen an einem Element zusätzliche Maßnahmen an anderer Stelle bedingen. Dies lässt sich an einem Beispiel aus Abbildung 4.4 verdeutlichen. Die Abhängigkeiten zwischen den Komponenten A, B und C, in der Abbildung dargestellt als schwarze Linien, sind jeweils doppelt gerichtet, hin zum Nachfolger und von diesem wieder zurück (*two way dependencies*). Ein Austausch oder Update der Komponente B durch die Komponente D wäre eine sehr zeitintensive und risikobehaftete Maßnahme. Es müsste sichergestellt werden, dass die neue Variante D mit A aber auch mit C zusammenarbeitet (gestrichelte Pfeile). In beide Richtungen sind Tests durchzuführen und unter Umständen sind an beiden betroffenen Komponenten Maßnahmen einzuplanen. So kann es sein, dass C nur mit D zusammenarbeitet, wenn C auf die neueste Version aktualisiert wird. Werden die Tests nicht umfassend und genau durchgeführt, so besteht das Risiko, dass Probleme erst später im produktiven Betrieb bemerkt werden. Die Komplexität steigt mit der Anzahl der Abhängigkeiten die auf eine Komponente wirken.
Nach Dyson und Longshaw sollten doppelt gerichtete Abhängigkeiten zwischen einzelnen Komponenten vermieden und das Prinzip der einfach gerichteten Abhängigkeit (*One-Way-Depenencies*) verfolgt werden. Der Austausch einer Komponente B durch D kann

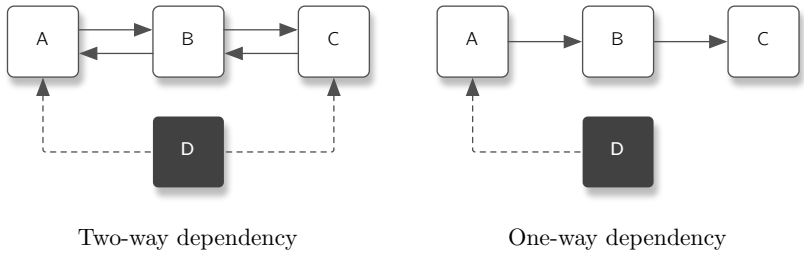

Abbildung 4.4.: Gerichtete Abhängigkeiten in einer Komponentenarchitektur

dann wesentlich einfacher vollzogen werden, da nur überprüft werden muss, ob D mit A zusammenarbeitet. Die Komponente C bleibt in diesem Fall unberührt. Die Testszenarien sind weniger umfangreich und die Folgen besser zu überblicken. Die Architektur kann flexibler und mit weniger Aufwand erweitert werden.

4.1.3.4. Einsatz von Standardprotokollen

Ist die Architektur nach dem Prinzip der einfach gerichteten Abhängigkeit durchgängig konzipiert, dann kann durch den Einsatz von Standardprotokollen zusätzliche Flexibilität gewonnen werden. Kommunizieren die Komponenten über Standardprotokolle wie z. B. HTTP, SOAP etc., dann können sie bei Bedarf ausgetauscht werden und durch ein anderes Produkt mit gleicher Funktionalität aber besseren nicht-funktionalen Charakteristiken ersetzt werden. Dyson und Longshaw lehnen die Verwendung von proprietären Protokollen ab: „The use of native protocols introduces a hard dependency that can make the replacement of a system element very difficult or even effectively impossible“ [DL04], S. 213.

4.1.4. Exkurs: Hochverfügbarkeit einer komponentenbasierten Infrastruktur

Nach Färber kann sich eine (System-) Komponente in den Zuständen *verfügbar* (funktionstüchtig) oder *ausgefallen* (nicht funktionstüchtig) befinden (vgl. [Fär94]). Im ausgefallenen Zustand findet in der Regel eine Reparatur der Komponente statt, die den Übergang in den Zustand *verfügbar* wieder ermöglicht (vgl. [Hel05]). Im weiteren Verlauf wird die Dauer der Verfügbarkeit eines Systems mit V(t) bezeichnet und die der Nicht-Verfügbarkeit mit N(t). Durch geplante und ungeplante Ausfallzeiten eines Systems findet über einen betrachteten Zeitverlauf ein mehr oder weniger häufiger Wechsel zwischen den Zuständen statt. Im Verlauf der Zeit setzt sich der Systemzustand einer Komponente aus Zyklen von Zeitabschnitten der Verfügbarkeit und Nicht-Verfügbarkeit zusammen. Die Zeitabschnitte zu denen ein System verfügbar ist, werden als *Time Between Failure* (TBF) und die Zeitabschnitte der Nicht-Verfügbarkeit als *Time to Repair* (TTR) bezeichnet. Die nachfolgende Abbildung stellt im zeitlichen Verlauf den Wechsel der Zustände eines Systems exemplarisch dar. Wird ein genügend langer Zeitraum betrachtet, so lassen sich hieraus zwei wichtige Kennzahlen ermitteln, mit denen die

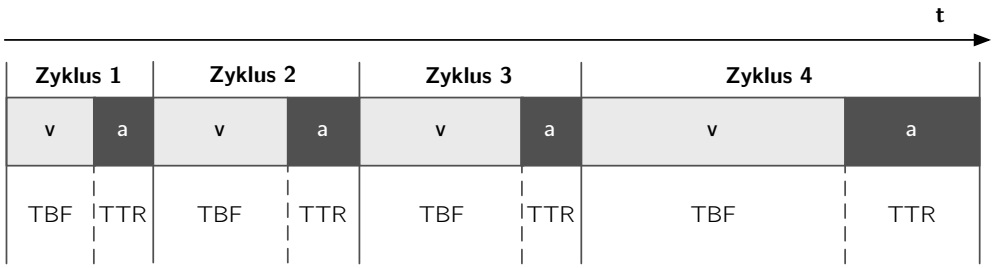

Abbildung 4.5.: Zeitverlauf des Systemzustandes (eigene Darstellung nach [Fär94])

Verfügbarkeit eines Systems bestimmt werden kann. Die durchschnittliche Verfügbarkeitsdauer (*Mean Time between Failures*, MTBF) wird als arithmetisches Mittel aus der Summe aller Zeitabschnitte, zu denen eine Komponente verfügbar war und der Anzahl der beobachteten Zyklen gebildet. Ein Zyklus wird dabei durch eine Folge der Übergänge V(t) nach N(t) nach V(t) gebildet. In gleicher Weise lässt sich die mittlere Dauer der Ausfallzeit (*Mean Time to Repair*, MTTR) bestimmen. Sie ist das arithmetische Mittel aus der Summe aller Zeitabschnitte, zu denen eine Komponente ausgefallen ist, und der Anzahl gemessener Zyklen.

Es gilt:

$$MTBF = \frac{\sum TBF}{Z} \tag{4.1}$$

und

$$MTTR = \frac{\sum TTR}{Z}, \tag{4.2}$$

wobei Z die Anzahl der in einem Zeitraum beobachteten Zyklen ist. „Die Verfügbarkeit eines Systems oder einer Systemkomponente ist definiert als prozentualer Zeitanteil, in dem dieses System bzw. die Komponente fehlerfrei funktioniert“ [Hel05], S. 36. Aus den beiden Kennzahlen MTBF und MTTR kann nun die prozentuale Verfügbarkeit (V) eines Systems mathematisch bestimmt werden.

Für die Verfügbarkeit eines Systems gilt:

$$V = \frac{MTBF}{MTBF + MTTR} \tag{4.3}$$

Die Nicht-Verfügbarkeit (N) eines Systems kann wie folgt bestimmt werden:

$$N = 1 - V = 1 - \frac{MTBF}{MTBF + MTTR} = \frac{MTTR}{MTBF + MTTR} \tag{4.4}$$

Die Gleichungen zeigen, dass durch die Erhöhung der MTBF und Reduzierung der MTTR die Verfügbarkeit einer Systemkomponente erhöht werden kann. Mögliche Maßnahmen zur Verbesserung der MTBF sind die Optimierung der Software- oder Hardwarekomponenten eines Systems. Dadurch wird zwar die Qualität einer einzelnen Komponente verbessert, jedoch führt dies nur zu einem gewissen Grad an Ausfallsicherheit

und Verfügbarkeit. Hinzu kommt, dass Ausfälle keineswegs zeitlich gleich verteilt sind (vgl. [Hel05], S. 39). Bei neuen Systemen kann es zu so genannten Frühausfällen kommen, die z. B. auf eine fehlerhafte Konfiguration oder eine funktionale Störung an der Komponente, die nur unter ganz bestimmten Bedingungen auftritt, zurückzuführen sind. Die Anzahl der Frühausfälle können durch umfangreiche Testszenarien, die insbesondere unter Last durchgeführt werden sollten, deutlich reduziert werden. Danach folgt eine Phase der statistischen Ausfälle. Zur Beurteilung des Ausfallrisikos während dieser Phase kann auf die ermittelten Angaben des Herstellers zurückgegriffen werden, sofern diese zur Verfügung stehen. Viele Hersteller können aus Erfahrungswerten oder Langzeittests die MTBF bzw. die Verfügbarkeit einer Komponente belegen. Durch Alterung und Verschleiß kann es in einer späteren Phase wiederum vermehrt zu Fehlern kommen. Dies gilt besonders für Bauteile mit einem hohen mechanischen Anteil, wie z. B. Festplatten oder Lüfter.

Ein komplexes System wie eine universitäre IT-Infrastruktur besteht jedoch nicht nur aus einer einzigen Komponente, sondern aus einer ganzen Reihe von Systemkomponenten, die in ihrer Zuverlässigkeit und Verfügbarkeit unterschiedlich ausgeprägt sein können. Hierbei gilt es zu beachten, dass eine Leistung von diesen Komponenten gemeinschaftlich als verteiltes System erbracht wird.

Ein wesentlicher Faktor für die Verfügbarkeit des Gesamtsystems ist dessen Architektur, d. h. die Strukturierung der Einzelkomponenten. In Abhängigkeit von der Anordnung der Komponenten kann die Gesamtverfügbarkeit eines Systems deutlich von den Verfügbarkeiten der Einzelkomponenten abweichen. Mit der Serien- und Parallelschaltung werden zwei Anordnungsvarianten unterschieden, auf denen die Formeln für die Berechnung der Verfügbarkeit von Gesamtsystemen beruhen. In der Serienschaltung sind Systemkomponenten in Reihe geschaltet und damit voneinander abhängig. Ein System, dass nur aus seriell angeordneten Elementen besteht ist nur verfügbar, wenn alle Einzelkomponenten verfügbar sind. Dementsprechend ergibt sich die Gesamtverfügbarkeit (V_g) als Produkt aus den Einzelverfügbarkeiten (V_i). In einer Serienschaltung von n Einzelkomponenten gilt:

$$V_g = \prod_{i=1}^{n} V_i \tag{4.5}$$

bzw. für $N << 1$

$$V_g = 1 - N_g = \prod_{i=1}^{n} (1 - N_i) = 1 - \sum_{i=1}^{n} N_i \tag{4.6}$$

„Die Fehlerquote eines Systems, das aus mehreren in Serie geschalteten Elementen besteht, steigt demnach mit der Zahl der Elemente. Betrachtet man hierfür nun die für die Sicherheit relevanten Elemente, so arbeitet ein solches System nur dann sicher, wenn alle Einzelkomponenten richtig arbeiten“ [Hel05], S. 40f. Durch Serienschaltungen wird die Ausfallzeit und Ausfallwahrscheinlichkeit des Gesamtsystems größer als die der Einzelkomponenten (vgl. [Hel05], S. 41; Abb. 4.6, Variante 1). In einem Architekturkonzept müssen Serienschaltungen zu Erhöhung der Gesamtverfügbarkeit bzw. zur Verringerung

der Ausfallwahrscheinlichkeit kompensiert werden. Erreicht werden kann dies durch die Parallelschaltung von Komponenten. Hierbei werden die Systemkomponenten parallel auf einer Ebene zusammengeschaltet und ergänzen sich somit redundant (vgl. Abb. 4.6, Variante 2). Solange eine Komponente in diesem Verbund zur Verfügung steht, kann die Funktionalität erbracht werden. Die Komponenten sind im Gegensatz zu einer Serienschaltung unabhängig voneinander. Die Gesamtausfallzeit einer Parallelschaltung ist das Produkt aus den Einzelausfallzeiten, so dass für die Gesamtverfügbarkeit gilt:

$$V_g = 1 - N_g = 1 - \prod_{i=1}^{n} N_{ija} \tag{4.7}$$

„Durch die redundante Anordnung wird zwar die Fehlerwahrscheinlichkeit im System aufgrund der größeren Zahl von Elementen erhöht. Doch durch die Art der Verknüpfung sinkt die Wahrscheinlichkeit eines Ausfalls des Gesamtsystems beträchtlich [...]“ [Hel05], S. 41.

Die vorgestellten mathematischen Formeln können Hinweise zur Bewertung der Anordnung von Komponenten innerhalb einer Architektur liefern und ermöglichen so eine zumindest theoretische Ermittlung der Verfügbarkeit eines Gesamtsystems. Die Abbildung 4.6 stellt drei Entwurfsentscheidungen gegenüber und beurteilt deren Auswirkung auf die Verfügbarkeit des Gesamtsystems. Es wird angenommen, dass die Komponente A mit einer Verfügbarkeit von 99,9 Prozent angegeben wird und die Komponente B mit 99,0 Prozent. In der zweiten und dritten Variante werden beide Komponenten jeweils durch Elemente mit identischen Verfügbarkeiten ausgetauscht.
Für die Implementierung verteilter Architekturen sind die Anforderungen der Systemkomponenten so vorzunehmen, dass die Verfügbarkeit des Gesamtsystems erhöht wird. Die Einordnung von Diensten in eine Verfügbarkeitsklasse kann mit den oben beschriebenen Formeln zumindest theoretisch abgestimmt werden. Ebenso lassen sich die Auswirkungen durch die Hinzu- oder Wegnahme einer Komponente auf die Verfügbarkeit des Gesamtsystems bestimmen. Um der Komplexität eines großen Systems in der Praxis vollständig gerecht zu werden, müssen jedoch weitere Aspekte berücksichtigt werden (vgl. hierzu [Hel05], S. 44).

4.1.5. Frameworks als semantische Referenzsysteme bei der Komponentenentwicklung

Neben der in Abschnitt 4.1.2.3 beschriebenen Funktionsintegration durch Middleware-Konzepte und Web-Services, die primär die Wiederverwendung von Programmcode – also die Portabilität einer Komponente – fokussiert, ist der Integrationserfolg von Anwendungssystemen und Komponenten auch stark von der Interoperabilität abhängig: Damit die Daten von mehreren Systemen korrekt verstanden werden können, muss Einigkeit über Syntax und Semantik der ausgetauschten Daten herrschen.

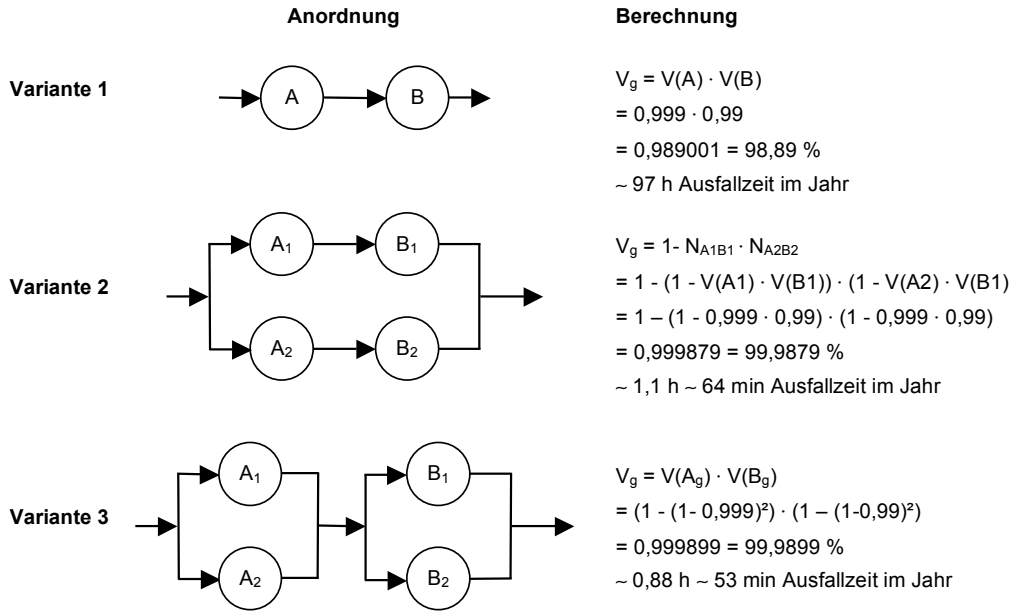

Abbildung 4.6.: Bestimmung der Verfügbarkeit bei unterschiedlicher Anordnung der Einzelkomponenten (eigene Darstellung in Anlehnung an [Hel05]).

Während die Syntax zum einen durch die Dienstbeschreibung, zum anderen durch das zur Kommunikation verwendete Rahmenwerk (bspw. SOAP) vorgegeben wird, kann die semantische Verbindung zwischen Systemen auf Basis neutraler bzw. genormter Austauschformate erfolgen. Dies ist oftmals dann notwendig, wenn die Systeme auf unterschiedlichen und getrennten Daten- und Speicherungsstrukturen basieren. Daten müssen dann vom Quellsystem über einen Pre-Prozessor in das Austauschformat und nach der Kommunikation im Zielsystem über einen Post-Prozessor in das Zielformat umgewandelt werden. Neben dem Aufwand der Modellierung stellt hierbei insbesondere die Integritäts- und Konsistenzsicherung eine Herausforderung dar, die nur durch eine übergreifende und ganzheitliche Betrachtungsweise gelöst werden kann (vgl. [Hel97], S. 38).

Wird die Semantik der in der Architektur verwendeten Konstrukte jedoch systemweit in eindeutiger Weise festgelegt, entsteht ein gemeinsames semantisches Referenzsystem mit einer allgemeinen für alle Anwendungen gültigen Daten- und Speicherungsstruktur[17]. Die Standardisierung erfolgt daher nicht mehr auf Ebene der Formate, sondern auf Ebene der Inhalte (vgl. ebd., S. 39). Der Beitrag zum Integrationsniveau ist offensicht-

[17] Unter Umständen sind dazu Konvertierungen der Zeichensätze (EBCDIC, ASCII, Unicode, etc.) und des Datenbankschemas notwendig. Middleware-Produkte können diese Aufgabe durch die Kapselung einer Reihe von standardisierten Funktionen jedoch unterstützen und so einen plattform- und technologieübergreifenden Zugriff zu erlauben. Neben nativen (anbieterspezifischen) Datenbankschnittstellen sind hier insbesondere sog. Call-Level Schnittstellen (*Call-Level Interfaces*, CLI) wie ODBC und JDBC zu nennen.

lich: „Komponenten, die miteinander kommunizieren oder kooperieren wollen, können dies mit geringerem Aufwand und zugleich auf einem höheren Sicherheitsniveau tun, wenn sie dazu in eindeutiger Weise auf Konstrukte verweisen können, deren Bedeutung systemweit bekannt ist“ [Fra94], S. 32.

Der Aufbau eines solchen integrierten semantischen Modells kann entweder schemabasiert erfolgen, bspw. auf Grundlage von Fach-Referenzmodellen, Branchen- oder domänenspezifischen Datenmodellen oder terminologiebasiert auf der Basis einer rekonstruierten Fachsprache eines Anwendungsgebietes (*Domain-Specific Language*).

4.1.5.1. Datenschema-basierte Integration

Die weite Verbreitung relationaler Datenbanksysteme nach der ANSI-SPARC-Architektur[18] hat dazu geführt, dass verteilte Systeme in den meisten Fällen über ein domänenweites, anwendungsübergreifendes Datenschema zusammengeführt werden. Ein solches Schema legt bei relationalen Datenbanken die Tabellen und ihre Attribute sowie Existenz- und Eindeutigkeitsbeziehungen fest.

Mit zunehmender Spezialisierung und funktionalen Erweiterungen der Anwendungen in der Domäne steigt jedoch der Aufwand für die Administration und Konsistenthaltung, da das konzeptuelle Schema zwar „ein gigantisches Entwicklungsergebnis auf der Anwendungsseite der Systeme darstellt“, aber nicht zum Entwicklungssystem respektive Entwicklungssprache an sich gezählt werden kann (vgl. [Ort00], S. 3). Es herrscht dann meist eine mehr oder weniger stark ausgeprägte Trennung von Daten- und Objektschicht vor[19]. Die Datenschicht stellt Funktionen zur Kontrolle und zur persistenten Speicherung der im System vorhandenen Informationen zur Verfügung. Die Objektschicht ermöglicht das Kontrollieren und Modifizieren der in der Persistenzschicht vorhandenen Daten.

Durch diese Differenzierung von Datenbank- und Anwendungsentwicklung sind das Design und die Pflege des zu entwickelnden (komplexen) Systems nur erschwert möglich oder mit hohen Kosten verbunden: Die Konsistenz des konzeptionellen Datenschemas muss zu jeder Zeit über alle Komponenten und Anwendungen hinweg aufrecht erhalten werden (vgl. [Oes04], [Ort00]).

Schienmann zählt in [Sch97a] weitere Defizite der Schema-basierten Integration auf, wie bspw. eine unzureichende Unterstützung der Modellierung globaler Systemdynamiken und objektübergreifender Funktionalität sowie die beschränkten Möglichkeiten der Wiederverwendung früherer Entwicklungsergebnisse.

4.1.5.2. Terminologiebasierte Entwicklung mit Objektframeworks

Die terminologiebasierte Integration von Systemen basiert zwar ebenfalls auf Datenbank- und Komponententechnologien, denn auch Termini sind – modellierungstechnisch

[18] Die Architektur wurde 1975 vom Standards Planning and Requirements Committee (SPARC) des American National Standards Institute (ANSI) beschrieben und garantiert durch eine Trennung verschiedener Beschreibungsebenen von Datenbankschemata (externe, konzeptuelle, und interne/physische Ebene) eine physische und logische Datenunabhängigkeit.

[19] Objektrelationales Mapping (O/R-M oder ORM) erlaubt zwar zum Zeitpunkt der Entwicklung eine objektorientierte Sicht auf ein relationales Datenbankschema. Der Aufwand für den Aufbau, die Administration und Weiterentwicklung des Datenschematas wird dadurch aber nicht gemindert.

gesehen – als „Schemata“ (sprachliche Repräsentation von Typen) aufzufassen. Anders als ein konzeptionelles Datenschema zählt eine Terminologie jedoch zum Entwicklungssystem und unterliegt darüber hinaus auch keinen strukturellen Beschränkungen durch eine Tabellen-orientierte Sichtweise (vgl. [Ort00]). Als Entwicklungssprache kann eine Terminologie mit geringerem Aufwand beispielsweise in Form eines Objektmodells organisiert und separat vom Anwendungsbetrieb – also verwendungsneutraler – administriert werden. Eine Abbildung des Gegenstandsbereiches der zu entwickelnden Anwendungen erfolgt dann durch die Rekonstruktion neuer Elemente mittels existierender Termini (siehe Unterkapitel 3.1).

Eine terminologiebasierte Komponentenentwicklung kann softwaretechnisch durch ein objektorientiertes Framework unterstützt werden, welches Wissen und Expertise in einem bestimmten Problembereich kapselt (*domain framework*, vgl. [Mat96], S. 57). Dieses Konzept wird seit den späten 80er Jahren eingesetzt, um ein domänenspezifisches objektorientiertes Design wieder zu verwenden und gleichzeitig Aspekte der Datenhaltung von der Anwendungsentwicklung zu separieren. Das Framework bietet dafür in der Regel ein domänenspezifisches Objektmodell generischer Klassen, auf das über Programmierschnittstellen zugegriffen und anwendungsspezifisch erweitert werden kann.

Die theoretischen Vorteile gegenüber der Datenschema-basierten Integration haben sich in der Praxis längst bestätigt: „In addition to the intuitive appeal of the framework concept and its simplicity from an abstract perspective, experience has shown that framework projects can indeed result in increased reusability and decreased development effort“ [Bos00], S. 241, vgl. hierzu auch [MN96].

In Kapitel 3 wurde das domänenspezifische Objektmodell der Wissensraummetapher bereits vorgestellt. Es wurde beschrieben, wie man mit diesem semantischen Bezugssystem Lern- und Arbeitsszenarien modellieren kann, ohne an ein festes Datenschema gebunden zu sein. Die technische Umsetzung mithilfe von Objektframeworks wurde in Abschnitt 3.2.3 ebenfalls aufgezeigt. Im Folgenden soll die spezielle Systemklasse dieser Frameworks für kooperatives Lernen und Arbeiten näher beschrieben werden, um damit die Grundlage für das Verständnis der Dienste der zu konzipierenden Rahmenarchitektur zu legen. Um dabei von bestimmten Technologien und Produkten zu abstrahieren, sollen primär die Variationspunkte der Frameworks skizziert werden.

4.1.5.3. Objektframeworks für CSCW/L-Systeme

Domänenspezifische Objektframeworks, die zur terminologiebasierten Entwicklung virtueller Umgebungen für kooperatives Lernen und Arbeiten herangezogen werden können, erfordern eine integrierte Kommunikation und eine sehr grundlegende Contentverwaltung. So genannte Groupware-Frameworks erfüllen diese Anforderungen. Auf Basis eines zumeist generischen Objektmodells unterstützen sie eine Bandbreite von Inhalts-, Gruppen- und Prozessstrukturen über offene Schnittstellen, mit denen verteilte kooperative Anwendungen erstellt werden können (vgl. [RG93]). Um eine klare Trennung (*one-way-dependency*, S. 124) zwischen dem Domänenmodell und dem Speichermodell zu

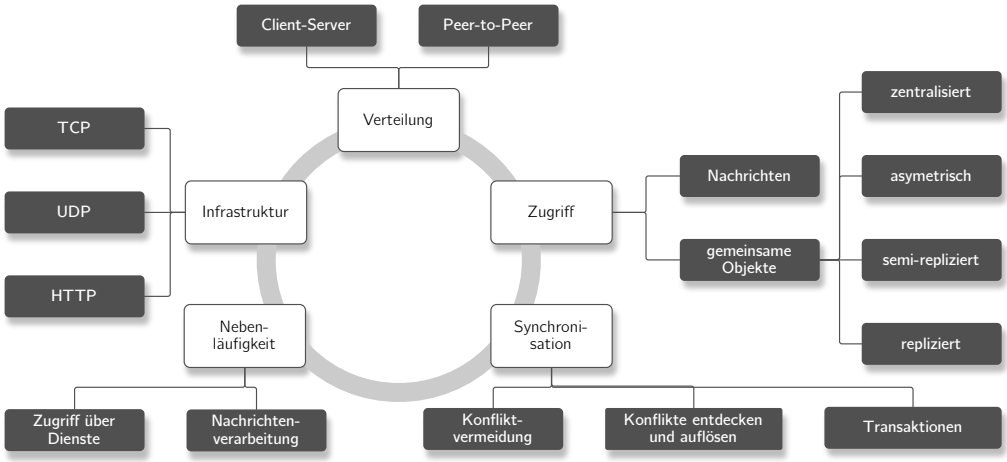

Abbildung 4.7.: Variationspunkte von Groupware-Frameworks nach [GTA05].

erreichen, werden in Groupware-Frameworks *Repositories* eingesetzt, die eine Anfrage-orientierte Schnittstelle implementieren. Somit können Objekte über das Repository domänenspezifisch angefragt werden, die dazu notwendigen Datenbankzugriffe geschehen durch das Repository gekapselt im Hintergrund[20].

Guicking et al. haben in [GTA05] die Variationspunkte synchroner Groupware-Frameworks analysiert und fünf essentielle Aspekte identifiziert, in denen sich einzelne Frameworks in puncto Softwarearchitektur und Funktion voneinander unterscheiden können (vgl. auch Abb. 4.7):

Infrastruktur Die Anbindung an eine technische Infrastruktur erfordert bestimmte grundlegende Schnittstellen zu Transport- und Kommunikationsprotokollen. Darauf können anwendungsnahe Protokolle aufsetzen, wie bspw. SOAP oder proprietäre Protokolle wie COAL[21].

Verteilung Die Verteilung der Architektur kann entweder nach dem Client-Server- oder nach dem Peer-to-Peer-Prinzip erfolgen. Jedes Prinzip hat seine Vor- und Nachteile. Während das Client-Server-Prinzip deutlich weniger komplex ist und Aspekte wie Konsistenz und die Behandlung von *Latecomers*[22] vereinfacht, vermeidet Peer-to-Peer den Server als Single-Point-of-Failure und „Flaschenhals“. Darüber hinaus bietet sich das Client-Server-Prinzip an, wenn eine Integration in eine heterogene Anwendungslandschaft gefordert ist oder mit verschiedenen Benutzungsober-

[20] Obwohl zu diesem Zweck objektorientierte Datenbanksysteme genutzt werden können, basieren viele Groupware-Frameworks häufig noch auf relationalen Datenbankmanagementsystemen.

[21] Das Groupware-Framework sTeam nutzt das COAL-Protokoll (*Client Object Access Layer*, vgl. [Ham02], S. 192) zur Kommunikation zwischen Server und clientseitigen Bibliotheken. Im Gegensatz zu standardisierten XML-basierten Protokollen wie SOAP weist das proprietäre COAL deutliche Geschwindigkeitsvorteile auf.

[22] Benutzer, die erst später zu einer Sitzung hinzustoßen, müssen den aktuellen gemeinsamen Bearbeitungsstand nachvollziehen können („What has happened here?“).

flächen auf die Anwendungen zugegriffen werden soll (mobile Endgeräte, Portale, etc.). Denkbar ist auch ein hybrides Konzept, bei denen ein Server einen Teil der Kommunikation zwischen den Clienten steuert, ein anderer Teil wie Audio- oder Videochat über eine Peer-to-Peer-Verteilung abgewickelt wird.

Zugriff Das kooperative Arbeiten und Lernen richtet sich an Inhalten und Strukturen aus, die dem gemeinsamen Zugriff unterliegen. Neben der Behandlung daraus möglicherweise resultierender Inkonsistenzen und Konflikte (s. u.) muss auch jeder Teilnehmer einer kooperativen Sitzung den gemeinsamen Arbeitsstand im Zugriff haben. Typische Verteilungsstrategien für gemeinsame Objekte sind zentriert oder repliziert (vgl. ebd.). Die Anwendungsmöglichkeiten sind dabei durch die Verteilung der Architektur geprägt. Eine replizierte Strategie setzt eine Anzahl kooperativer Instanzen einer Applikation voraus, die über das Netz eigenverantwortlich über einen gemeinsamen Datenbus kommunizieren. Eine zentrale Verteilung ist dagegen durch einen zentralen Prozess geprägt, der die Kommunikation zwischen den Instanzen steuert. Daneben können moderne Rich-Client-Architekturen durchaus auch hybride Verteilungsstrategien implementieren und einen Teil der Kommunikation mit anderen Clients eigenverantwortlich steuern. Damit die Realisierung der Objektverteilung nicht zum Bestandteil der Anwendungsentwicklung wird, muss das Framework hierzu eine entsprechende Abstraktion bieten.

Synchronisierung Bei der Herstellung des gemeinsamen Arbeitszustands durch den Abgleich gemeinsamer Objekte können Konflikte auftreten, die entweder vermieden oder aber entdeckt und aufgelöst werden müssen. In diesem Zusammenhang sichern Transaktionen die Bewahrung eines konsistenten Zustands im Konfliktfall. Danach wird der Zustand nur aktualisiert, wenn keine Konflikte bei der Synchronisation aufgetreten sind.

Nebenläufigkeit Abhängig von der Objektverteilung muss das Framework sowohl gleichzeitige Nachrichtenübermittlung als auch die nebenläufige Ausführung von Diensten potenziell unterstützen. Dazu müssen sowohl die Zugriffe auf Ressourcen der Infrastruktur koordiniert werden als auch das Zusammenspiel verteilter Komponenten, ohne nicht-funktionale Aspekte wie Performance, Robustheit und Skalierbarkeit negativ zu beeinträchtigen.

Darüber hinaus unterscheiden sich die auf dem kommerziellen und freien Markt vorhandenen Produkte zum Teil deutlich im Funktionsumfang. Die meisten Groupware-Frameworks werden mit einer Webschnittstelle ausgeliefert, die bereits verschiedene Kooperationsdienste wie Gruppen- und Dokumentenmanagement, Chat und Wikis implementieren. Zu manchen Frameworks existiert bereits eine abgestimmte Produktlinie[23].

[23] Die Hyperwave AG bietet bspw. auf Basis des Hyperwave Information Servers IS/6 eine ganzheitliche Infrastruktur zum *Collaboration Information Management* (CIM) an, bestehend aus Produkten zum Wissensmanagement, Teamräumen, E-Learning, E-Conferencing und zum Records Management. Die Lotus-Produktlinie von IBM unterstützt sogar noch weitere Anwendungsfelder.

In der Literatur finden sich ausführliche Analysen und Gegenüberstellungen von Groupware-Frameworks (bspw. [Ham02], S. 176ff., [GTA05], S. 51ff.). An dieser Stelle sollen daher mit der Tabelle 4.2 nur die bestehenden Aufstellungen um aktuelle Produkte ergänzt werden. Eine Gegenüberstellung der Produkte ist nicht Bestandteil dieser Arbeit, kann jedoch prinzipiell über die in diesem Abschnitt gelisteten Variationspunkte vorgenommen und – als Entscheidungsgrundlage für eine Softwareauswahl – konkreten Anforderungen gegenüber gestellt werden.

Produkt	**Hersteller**	**Weblink**
Alfresco Enterprise CMS	Alfresco Software, Inc.	`http://www.alfresco.com/`
Citadel Groupware Server	Uncensored Communications Group	`http/www.citadel.org/`
Clearspace	Jive Sofware	`http://www.jivesoftware.com/`
Confluence	Atlassian	`http://www.atlassian.com/software/confluence/`
DyCE	go4teams GmbH	`http://www.go4teams.com/`
Hyperwave IS	Hyperwave AG	`http://www.hyperwave.com/d/products/is6/`
Lotus	IBM	`http://www.ibm.com/software/de/lotus/`
Open sTeam	Heinz Nixdorf Institut	`http://www.open-steam.org/`
Open-Xchange	Open X-Change GmbH	`http://www.open-xchange.com/`
Scalix	Xandros Corporation	`http://de.scalix.com/`
SharePoint Services	Microsoft	`http://office.microsoft.com/de-ch/sharepointtechnology/`
teamXchange	VIPcom GmbH	`http://www.vipcomag.de/teamxchange/`

Tabelle 4.2.: Beispiele aktueller Groupware-Frameworks (ohne Anspruch auf Vollständigkeit).

4.2. Eine Rahmenarchitektur für verteilte Lern- und Arbeitsumgebungen

Nachdem in den vorangegangenen Unterkapiteln die notwendigen Grundlagen gelegt und entsprechende Architekturprinzipien aufgezeigt wurden, kann im Folgenden die eigentliche Konzeption einer Produktlinien-orientierten Rahmenarchitektur für universitäres E-Learning durchgeführt werden.

Dabei wird die Idee der in Abschnitt 2.2.4.3 vorgestellten komponentenorientierten E-Learning-Architekturen weiter konkretisiert. Die zwei wichtigsten Spezialisierungen gegenüber abstrakten Rahmenarchitekturen wie IEEE LTSA, ELF oder IAF (s. Abschnitt 2.2.4.3) sind (1) die Nutzung eines technischen CSCW/L-Rahmenwerks zur Abdeckung der Basisdienste und (2) eine Zweiteilung der Rahmenarchitektur in Produktstandard- und darauf aufbauender Produktarchitektur.

Diese Konkretisierungen werden in diesem Unterkapitel herausgearbeitet und um Aspekte des universitären Anwendungskontextes verfeinert. Somit wird eine Rahmenarchitektur geschaffen, die konkret auf den universitären Einsatz ausgerichtet ist.

4.2.1. Schichtenmodell

Grundlegend für die Einordnung von Standardprodukt- und Produktarchitektur ist eine logische Systemarchitektur, die in diesem Abschnitt anhand eines Schichtmodells konstruiert werden soll. Dabei wird nach dem OSI-Modell zwischen *Diensten*, *Schnittstellen* und *Protokollen* unterschieden: Die Gesamtheit der Dienste definiert den Funktionsumfang einer Schicht, eine Schnittstelle, wie auf einen Dienst zugegriffen werden kann und ein Protokoll definiert das Format und die Bedeutung der zu übertragenen Informationen.

Nach [Kar95], S. 12ff., können Dienste nach ihrer Wiederverwendbarkeit in die drei Klassen „Allgemeine Dienste“, „Domänen-spezifische Dienste“ und „Produktlinien-spezifische Dienste“ eingeordnet werden. In dieser Arbeit werden Allgemeine Dienste als Teil der Infrastruktur aufgefasst (Basistechnologien und Registrierungsdienste, vgl. Abb. 4.9). Eine verteilte Komponentenarchitektur für universitäres E-Learning lässt sich demnach durch vier Schichten repräsentieren: Infrastrukturschicht, spezifische Dienste der Domäne „Kooperatives Lernen und Arbeiten“ (Kooperationsdienste) sowie Produktlinien-spezifische Dienste hinsichtlich des universitären E-Learnings (E-Learning-spezifische Dienste). Obenauf liegt die Anwendungsschicht, die auf den Diensten basierende Benutzungsschnittstellen für bestimmte Anwendungszwecke des universitären E-Learnings umfasst. Die untersten drei Schichten stellen Dienste zur Verfügung, die sowohl innerhalb einer Schicht als auch von den darüber liegenden Schichten genutzt werden können. Die Dienste sollten dabei den Entwurfsprinzipien komponentenorientierter Architekturen entsprechen, insbesondere sollten die Dienste einer Schicht einfach gerichtete Abhängigkeiten zueinander aufweisen und Standardprotokolle nutzen (vgl. Abschnitt 4.1.3). Abbildung 4.8 stellt diesen Zusammenhang als UML-Komponentendiagramm dar.

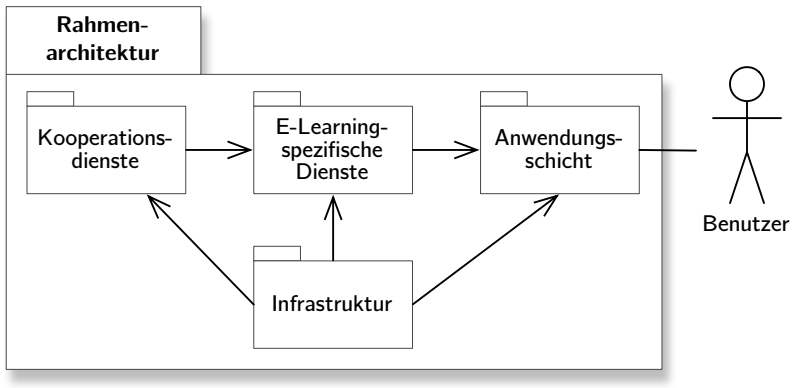

Abbildung 4.8.: Abhängigkeiten zwischen verschiedenen Dienst-Klassifizierungen

4.2.1.1. Infrastruktur

Die Netzwerk- und Transportschicht bildet die Basis jeder verteilten Komponentenarchitektur. Durch die Nutzung von *Netzwerkprotokollen* werden Transaktions- und Kommunikationsdienste für Dienste aller Schichten zur Verfügung gestellt. Die Kommunikation zwischen zwei Komponenten kann dabei in vielfältiger Art und Weise erfolgen, bspw. direkt über infrastrukturgebundene *Transportprotokolle* oder über darauf aufbauende XML-basierte *Nachrichteninfrastrukturen*, die einen umfangreichen Transport- und Kommunikationsservice anbieten[24].

Neben den Transportprotokollen sind in dieser Schicht auch komplexere *Basistechnologien* wie die Dienste eines Zertifikatsservers, Verzeichnisdienstes oder eines Mail-Servers zu verorten. Auch protokollnahe synchrone Kommunikationsdienste wie Instant Messaging, Audio- und Videochat können zu den komplexen Diensten einer Infrastruktur gezählt werden.

Ein wichtiger Aspekt einer komponentenorientierten Architektur ist die Möglichkeit zur losen Kopplung von Komponenten. In der konkreten Umsetzung auf verschiedenen Ebenen der Architektur hängt die Modularisierung stark von der verwendeten Technologie ab. In den meisten Fällen sieht das Konzept jedoch vor, eine Komponente gegen einen symbolischen Namen aufzulösen. Weit verbreitet ist das Entwurfsmuster *Dependency Injection*, bei dem die Auflösung dieser Abhängigkeiten von Frameworks übernommen und zur Laufzeit in ein aufrufendes Objekt hineininjiziert werden. Hierzu werden in der Infrastruktur *Registrierungsdienste* benötigt, die Komponenten mitsamt ihren symbolischen Namen verzeichnen und auflösen können[25].

[24] SOAP-Frameworks bspw. bieten die Verpackung in XML-Umschläge (*envelope*) an, einen verlässlichen Datentransport, *Publish-* und *Subscribe*-Mechanismen, Nutzung von „HTTP-GET"- und „HTTP-POST"-Mechanismen etc.

[25] Abhängig vom Softwaredesign und den eingesetzten Technologien können dies innerhalb der gleichen Softwarearchitektur durchaus unterschiedliche Mechanismen sein. Registrierungsdienste einer möglichen javabasierten Webarchitektur sind bspw. das *Java Naming and Directory Interface* (JN-

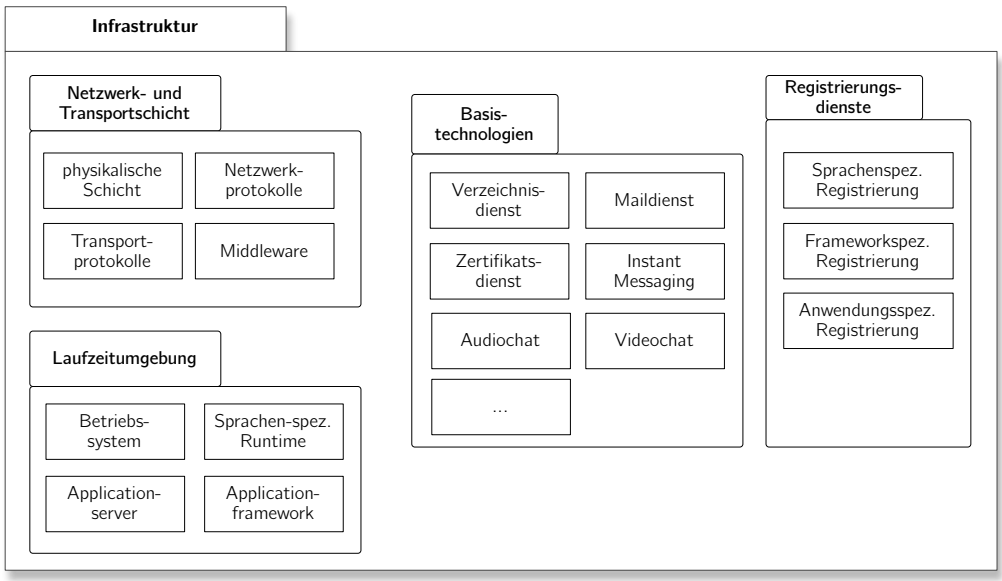

Abbildung 4.9.: Infrastruktur und Basisdienste

Die ausführende Komponente der Infrastruktur ist die *Laufzeitumgebung*, die den Anwendungskomponenten grundsätzliche Funktionen wie Schreiben und Lesen von Dateien, Steuerung von Ein- und Ausgabegeräten, Garbage Collector und Ähnliches zur Verfügung stellt. Sie kann – je nach eingesetzter Technologie – den Architekturstack entlang spezifiziert werden. Neben der Laufzeitumgebung der eingesetzten Sprache[26] können so auch Application Server und Programmierframeworks die Laufzeitumgebung mit definieren.

4.2.1.2. Kooperationsdienste

Spezifische Dienste der Domäne des kooperativen Arbeitens und Lernens sind nötig, um den gemeinschaftlichen Zugriff auf Wissensobjekte unterstützen zu können. Beispielsweise Dienste zum entfernten Zugriff auf Objekte, zur Unterstützung von Zugriffs- und Versionskontrolle, Dienste zur Notifikation und Unterstützung der Awareness. Insbesondere gehören Dienste zur Bereitstellung und Verwaltung virtueller Wissensräume zu dieser Klasse. Groupware-Frameworks wie die in Abschnitt 3.2.3 vorgestellten Systeme implementieren diese Dienste.

DI) als Teil des Java Enterprise Standards, die Service Registry der *Open Services Gateway Initiative* (OSGi) zur Nutzung in Application Frameworks wie Spring, oder anwendungsbezogene Modularisierungskonzepte wie die *Public Service API* des Enterprise CMS Alfresco.

[26] Gemeint ist hier bspw. das *Java Runtime Environment* (JRE), bestehend aus den Java Klassenbibliotheken und der *Java Virtual Machine* (JVM), ein PHP-Interpreter als Webserver-Modul oder für die Konsole oder Microsofts Common Language Runtime (CLR) für C#, Visual Basic.NET und C++.NET.

Kooperative Dienste können in unterschiedlichsten Anwendungsfeldern des kooperativen Arbeitens und Lernens verwendet werden, bspw. zur Unterstützung des Wissensmanagements in F&E-Projekten, virtuellen Communities of Practice oder eben auch des universitären E-Learnings (vgl. hierzu [PRH06], [HRKK05] und [RS05]).

Komponenten, die domänenspezifische Dienste des kooperativen Arbeitens und Lernens implementieren, können den Kategorien *Objektmanagement*, *Benutzermanagement*, *Kommunikation* und *unterstützende Dienste* zugeordnet werden.

Objektmanagement Das kooperative Arbeiten und Lernen auf Basis gemeinsam genutzter Artefakte impliziert architektonisch die persistente Verwaltung von Objekten. Dieses Objektmanagement umfasst Dienste zur Sicherung eines konsistenten Datenzustands sowohl in der Persistenzschicht als auch zur Laufzeit (*Persistenzmanager*, *Objektmanager*), die Sicherung der Konsistenz auch bei nebenläufigem Zugriff (*Locking*, *Versionskontrolle*), sowie die *Zugriffskontrolle* und ein internes *Benachrichtigungssystem*, das Zustandsänderungen an Objekten registriert und bei entsprechender Konfiguration andere Dienste darüber in Kenntnis setzen kann. Letzteres ist insbesondere für das synchrone Arbeiten essentiell, um Benutzer über Aktivitäten anderer Benutzer zu informieren (vgl. [HPD05]). Dienste zur *Archivierung* und *Paketierung* unterstützen die Langzeitaufbewahrung sowie den Export zur Übertragung der Daten in externe Systeme.

Benutzermanagement Das *Benutzermanagement* deckt die Verwaltung von Benutzern und hierarchischen Benutzergruppenstrukturen ab. Ein *Rollenmanager* kann dabei die explizite Zuweisung von Funktionsrechten zu Benutzern vornehmen und so verschiedene Akteure definieren, die mit dem System interagieren können. Darüber hinaus wird eine Komponente zur *Authentifizierung* benötigt, die Anmeldedaten verifiziert[27] und bei Erfolg eine Sitzung[28] initiiert. Um die Verwaltung von Sitzungen zur Laufzeit kümmert sich ein *Session-Manager*, der mit einer *Awareness-Komponente* interagiert, durch die angemeldete Benutzer im System für andere sichtbar sind.

Kommunikation Um einen vielfältigen Austausch von Informationen zu unterstützen, werden Komponenten gebraucht, die Kommunikationswerkzeuge implementieren. Neben den zur Infrastruktur zählenden synchronen Kommunikationsdiensten sind *Foren*, *Wikis* und *Weblogs* in kooperativen Lern- und Arbeitsumgebungen weit verbreitet. Insbesondere zum Informationsaustausch über vorhandene Wissensobjekte eignen sich *Kommentarfunktionen* und *Shared Whiteboards*, da sie – im Gegensatz zu den bisher aufgeführten Werkzeugen – die Objekte in den Mittelpunkt der Kommunikation stellen.

Unterstützende Dienste Unterstützende Dienste decken grundlegende Funktionen ab, die Instanzen von Komponenten zur Laufzeit benötigen, um untereinander und mit der Infrastruktur zu kommunizieren. Beispiele sind das *Modulmanagement*, über das neue Komponenten registriert werden können, *Logging* und *Fehlerbehandlung* und die

[27] In vielen Systemen umfasst diese Komponente auch Möglichkeiten zur Interaktion mit verschiedenen Basisdiensten zur Authentifizierung, wie Verzeichnisdienste oder Ticketsysteme.

[28] Eine Sitzung ist eine zeitweise bestehende Verbindung zwischen Client und Server.

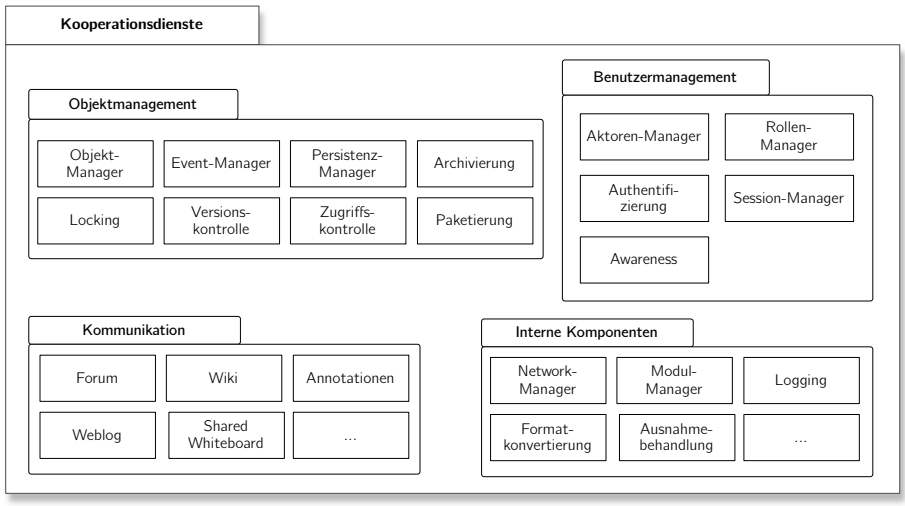

Abbildung 4.10.: Kooperationsdienste

Implementierung von Komponentenschnittstellen zu Diensten der Infrastrukturschicht (so genannte *Protokolladapter*, vgl. [SHB04]), bspw. zu Netzwerkprotokollen der Anwendungsschicht wie Jabber oder WebDAV. Eine weitere wichtige Komponente sollte Dienste zur *Formatkonvertierung* zwischen verschiedenen Inhaltstypen implementieren.

4.2.1.3. E-Learning-spezifische Dienste

Die dieser Schicht zugeordneten Dienste finden aufgrund ihrer Spezialisierung auf universitäres E-Learning nur innerhalb einer Produktlinie Wiederverwendung. Eine Übertragung auf thematisch verwandte Szenarien wie bspw. Wissensmanagement in F&E-Projekten ist i. d. R. nicht möglich.

Dienste dieser Schicht sind zum Beispiel ein auf Modulstrukturen beruhendes Kursmanagement, die Sequenzierung von Lerninhalten nach IMS Standards, Dienste zur Diskursstrukturierung oder ein den allgemeinen und bereichsspezifischen Regelungen zum Datenschutz genügendes Prüfungsmanagement.

Komponenten, die E-Learning-spezifische Dienste implementieren, können den Kategorien *Organisation*, *Campus Management*, *Lernszenarien* und *Community* zugeordnet werden.

Campus Management Grundlegend für die Unterstützung universitären E-Learnings ist eine geeignete Abbildung der Modul- und Prüfungsorganisation. Hierzu ist eine hierarchische Struktur von Modulen, Semestern, Moduldurchläufen und Unterveranstaltungen wie Übungsgruppen, Tutorien, Parallelveranstaltungen etc. durch ein *Veranstaltungsmanagement* zu unterstützen. Sehr eng mit dem Veranstaltungsmanagement verbunden ist das *Prüfungsmanagement*, welches auf Basis der existierender Prüfungsordnungen konfigurierbare Anmelde- und Belegverfahren für Veranstaltungen imple-

mentiert und die *Fortschrittskontrolle* von Studierenden innerhalb ihrer belegten Studiengänge unterstützt. An allen öffentlichen Universitäten werden diese Dienste bereits durch ein eigenständiges *Campus Management System* übernommen und können somit über Schnittstellen in die Komponentenarchitektur mit aufgenommen werden (vgl. hierzu Schnittstellen zum Campus Management, S. 152ff.).

Lernmanagement Komponenten dieser Kategorie zielen auf die Planung, Durchführung und Kontrolle von Lernprozessen ab. Dazu sind Dienste notwendig, welche die Entwicklung von Lernenden unter Berücksichtigung des Lernertyps anhand der Systemaktivität (*Tracking*) oder gezielten Überprüfungen (*Assessment*) registriert. Ein *E-Portfolio* unterstützt den Lernenden bei der Reflexion seiner Lernprozesse mithilfe eines persistenten Speichers für Artefakte, die im Verlauf seines Studiums erstellt werden. Bestandteile einer solchen „Sammelmappe" sind Sammlungen von Arbeitsergebnissen und Anmerkungen (bspw. von Tutoren oder Lehrenden), Feedback-Möglichkeiten, aber auch eigene Notizen zur persönlichen Selbstreflexion.

Lernszenarien Soll das Arbeiten und Lernen in virtuellen Räumen einer bestimmten Instruktion erfolgen, welche die Handlungsmöglichkeiten durch die Bindung an bestimmte Rollen, Reihenfolgen oder Medientypen einschränkt, müssen dazu entsprechende Restriktionen in der Softwareschicht vorgehalten werden. Die dieser Kategorie zugeordneten Komponenten unterstützen diese Einschränkungen bspw. durch konfigurierbare *Diskursstrukturen*, eine rollenbasierte *Ressourcenverwaltung* oder die *Sequenzierung* von Lektionen und Inhalten. Darüber hinaus wird oftmals eine *Workflow*-Komponente benötigt, die – basierend auf Zugriffsrechten und Bearbeitungsständen von Artefakten – Bearbeitungszyklen und Freigaben steuert.

Community Über die Gruppenverwaltung der kooperativen Dienste hinausgehend werden zur Unterstützung von sozialen Netzwerken weitere Funktionen benötigt. Die Verwaltung bestätigter und unbestätigter Kontakte wird durch eine *Kontaktmanagement*-Komponente übernommen. Durch diese Komponente können auch kürzeste Wege zu Benutzern ermittelt werden, zu denen kein direkter sozialer Kontakt besteht. Auf Basis des Kontaktmanagements können Benutzer individuelle Freigaben auf Daten des eigenen Profils einrichten (*Profildatenmanagement*).

4.2.1.4. Anwendungen

Die Anwendungen implementieren eine Benutzungsschnittstelle, durch die der Endbenutzer über eine Menge von Diensten mit bestimmten Artefakten interagieren kann. Einige typische E-Learning-Anwendungen wurden bereits in Tabelle 2.2 in dieser Arbeit vorgestellt. Eine abstraktes Klassifikationsschema kooperationsunterstützender Systeme anhand der Struktur des unterstützten Szenarien liefert Haake in [Haa99]. Danach bilden die drei Dimensionen Prozess-, Gruppen- und Inhaltsstruktur einen Strukturraum, in dem Anwendungen entsprechend ihrer Ausprägungen angeordnet werden können. Der Grad an möglicher Selbstorganisation stellt in jeder Dimension die ordinale Skala von selbststrukturiert, über semi-strukturiert bis hin zu vollständig vorstrukturiert.

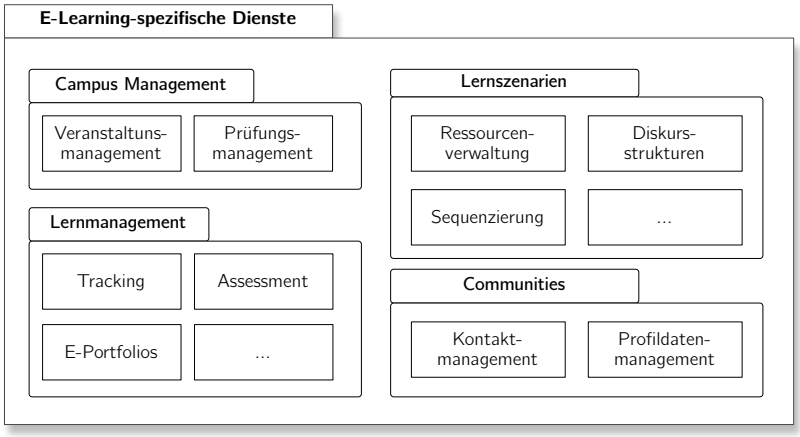

Abbildung 4.11.: E-Learning-spezifische Dienste

Die *Prozessstruktur* ist zentraler Aspekt von Workflow-Management-Anwendungen. Workflows, die fest vorgegeben sind, finden sich einerseits häufig in der organisatorischen Unterstützung des Lernmanagements, wie dem Einrichten von Veranstaltungen oder das Melden von Prüfungsergebnissen an die Verwaltung, andererseits auch im Learning Design oder Lernprotokollen wieder, die den Ablauf von Lerneinheiten genau spezifizieren oder den Ablauf von Diskursstrukturen genau vorgeben (vgl. dazu [PM02] und [HH05]).

Semi-strukturierte Workflows bieten neben einem vorstrukturierten Teil auch selbstorganisierte Abschnitte. Ein Beispiel hierfür ist die in [RHS06] vorgestellte Unterstützung der Projektmethode: In einem LMS-unterstützten Planspiel einer TV-Produktion wechseln sich Gruppenarbeitsphasen mit instruierenden Einführungsphasen und moderierten Reflexionsphasen ab (vgl. auch [Jan04b]).

Ad-hoc Workflows werden von Benutzern selbst organisiert. Ein Beispiel ist das Lernen in freien Lerngruppen im koaLA-System der Universität Paderborn (vgl. Kapitel 6). Studierende können dazu zwischen öffentlichen oder privaten Gruppen wählen, die Aufnahme neuer Mitglieder durch Passwortvergabe oder Bewerbungsverfahren regeln oder auch frei erlauben, sowie Rechte und Rollen im gemeinsamen Gruppenarbeitsraum frei vergeben.

Die *Gruppenstrukturen* in Lern- und Arbeitsumgebungen können ebenfalls stark vorstrukturiert sein, um bspw. mit einer rollenbasierten Zuordnung von Aktivitäten zu Benutzern/Benutzergruppen eine feste Teamzusammensetzung vorzugeben. Ebenso kann die *Inhaltsstruktur* ausschließlich nach festen Regeln erfolgen, d. h. Speicherort, Dateinamen, Versionierungssystem, Zugriffs- und Referenzmöglichkeiten sind dadurch festgelegt. Im Gegensatz dazu bilden virtuelle Dateiordner oder *Shared Whiteboards* selbstorganisierbare Handlungsräume, in dem Benutzer frei sind in der Gestaltung ihrer Inhaltsstrukturen.

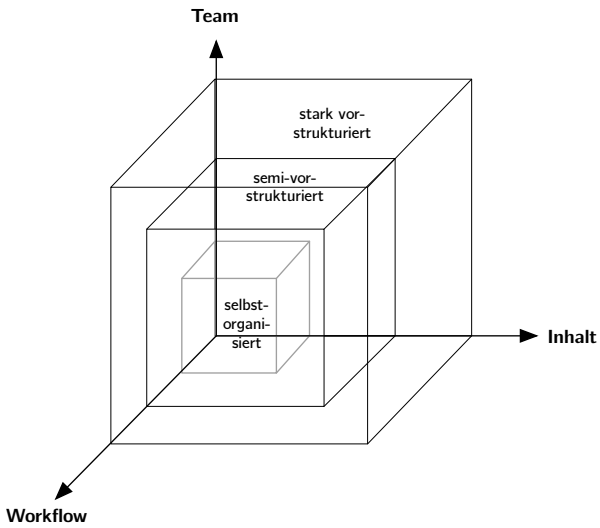

Abbildung 4.12.: Spezifikationsraum für kooperative Lern- und Arbeitskontexte (in Anlehnung an [RH04], S. 120)

Basierend auf dem Wiederverwendungsgrad der hier beschriebenen Dienste wird im Folgenden zunächst die Produktstandardarchitektur, dann die Produktarchitektur konzipiert.

4.2.2. Die Produktstandardarchitektur

Die Umsetzungsmöglichkeiten von kooperativen Diensten wurden im vorigen Kapitel 3.2.3 schon angedeutet. Als Alternative zur Implementierung der Dienste mithilfe der Basistechnologien wurden in Abschnitt 4.1.5.3 Groupware-Frameworks genannt, die zur Entwicklung und Unterstützung von geteilten Lern- und Arbeitsplattformen herangezogen werden können. Groupware-Frameworks stellen jeweils eine Reihe von Kooperationsdiensten für ein bestimmtes Objektmodell (*Domänenmodell*) zur Verfügung.
Domänenmodelle sind in der Regel generisch, und beinhalten – im Fall des kooperativen Arbeitens und Lernens – Abstraktionen zur Abbildung von Gruppen-, Prozess- und Inhaltsstrukturen[29]. Als Beispiel wurde im Kapitel 3.2.3 das in Groupware-Frameworks verbreitete Modell der Wissensraummetapher angeführt.
Im Folgenden soll aus softwarearchitektonischer Sicht beschrieben werden, wie auf Basis eines solchen generischen Objektmodells wiederverwendbare Modelle konfiguriert werden können, die Sachverhalte kooperativer Arbeits- und Lernumgebungen beschreiben. In Abhängigkeit ihres Anwendungsbezugs können diese Modelle entweder für einzelne Anwendungen oder aber – als semantisch integrative Komponente – für mehrere Anwendungen einer Softwareproduktlinie eingesetzt werden (vgl. S. 131).

[29] Damit gehen sie über Hypermedia-Modelle hinaus, die nur die reine Inhaltsverwaltung unterstützen (vgl. hierzu [HS90]).

Die in diesem Abschnitt beschriebene Kernarchitektur ist in allen Anwendungen grundsätzlich identisch und wird daher im Folgenden auch als „Produktstandardarchitektur“ bezeichnet. Darauf basierende anwendungsspezifische Erweiterungen werden unter der Bezeichnung „Produktarchitektur“ im nächsten Abschnitt erläutert.

4.2.2.1. Gemeinsame Modelle

Die Objekttypen des generischen Domänenmodells stellen die Komponenten zur Modellierung von Arbeits- und Lernkontexten im Rahmen des in Abbildung 4.12 aufgespannten Spezifikationsraums. Die normsprachliche Spezifizierung der dafür notwendigen Sichten wurde mit Kapitel 3.3 ausführlich beschrieben. Demnach können Kompositionsbeziehungen über Container und Gruppen als Behältnisse oder Kollektionen von Objekten und Benutzern umgesetzt werden. Eine weitere Möglichkeit besteht in der Speicherung von Objektreferenzen als Attribut eines Objektes (vgl. S. 101ff.).

Diese konstruierten Objektverbünde (*Composites*, vgl. [GHJV96], S. 213ff.) bieten per se spezifische Interaktionen mit ihren einzelnen Modellelementen, da durch das Repository jede Abstraktion mit Daten- und Verhaltensbeschreibungen assoziiert ist. Über eine Modell-Konfiguration kann definiert werden, mit welchen Kooperationsdiensten auf die Composites zugegriffen werden kann und welche zusätzlichen Eigenschaften das ganze Modell bzw. einzelne Modellelemente besitzen.

Zusätzlichen Eigenschaften wie Versionierbarkeit, Annotierbarkeit, Kategorisierbarkeit u. ä. werden daher klassenübergreifend benötigt. Das objektorientierte Design gelangt hier an seine Grenzen: Entweder muss jeder Objekttyp die Funktionalität neu implementieren oder es gibt eine „Überklasse“, die diese Funktionen bereitstellt, wie bspw. eine Klasse *Object*. Dies würde aber den objektorientierten Ansatz ad absurdum führen.

Moderne Groupware-Frameworks nutzen zur Behandlung von Aspekten, die eine Anforderung quer durch alle logischen Schichten „schneiden“ (*Cross Cutting Concerns*, vgl. S. 78), das im Kontext des Registrierungsdienstes bereits vorgestellte Entwurfsmuster *Dependency Injection*. Dieses Muster erlaub das „Injizieren“ von Fähigkeiten in die Objekte zur Laufzeit der Anwendung, so dass allgemeine Fähigkeiten einerseits nur einmal programmiert werden müssen, andererseits nur an den Objekten verfügbar ist, die sie auch wirklich benötigen.

Ein Beispiel für die Implementierung querschnittlicher Belange im Kontext eines generischen Objektmodells ist in Abbildung 4.13 zu sehen. Hier wird die Fähigkeit, kommentiert werden zu können, als Aspekt konfiguriert. Sie ist damit klasseninvariant und kann in verschiedenen Modellen wiederverwendet werden. So wird eine Möglichkeit geschaffen, systemweit ein einheitliches Verfahren für Kommentare zu implementieren.

Diese Möglichkeit der aspektorientierten Programmierung (AOP) kann in der Konfiguration eines Modells zur Definition wiederkehrender Funktionalität verwendet werden. Eine darauf basierende Konfiguration zur Beschreibung eines Weblogs findet sich im Anhang A.4 dieser Arbeit.

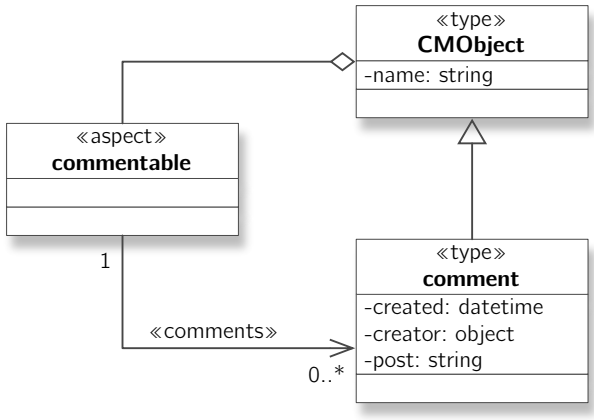

Abbildung 4.13.: Aspektorientierte Umsetzung von Kommentaren an Objekten auf Basis der Wissensraummetapher des Alfresco-Servers.

Die durch die Normsprache spezifizierten und durch die Konfiguration des Groupware-Frameworks definierten Modelle beschreiben Gegenstandsbereiche einer Lern- und Arbeitsumgebung, die kooperativ – also gemeinsam – über Softwareprodukte manipuliert werden können. Damit ein solches Modell zur Implementierung der Anwendungslogik verschiedener Produkte wieder verwendet werden kann, muss es in den einzelnen Produktarchitekturen lokal verfügbar gemacht werden. Pattersons Groupwarereferenzmodell beschreibt dazu die „Geteilte Architektur“ (vgl. [SS01], S. 298f.): Die Produktarchitektur implementiert eine eigene Sicht auf ein gemeinsames, zentral persistiertes Modell. Somit kapselt das gemeinsame Modell Daten über seinen allgemeinen Zustand, die darüber hinaus noch durch anwendungsspezifische Daten ergänzt werden können[30]. Dieses Entwurfsmuster ist in der Literatur auch unter der Bezeichnung *shared state* bekannt[31]. Die Herstellung eines gemeinsamen Arbeitszustands erfolgt dabei nicht durch den Austausch von Änderungsinformationen (Synchronisation), sondern ereignisbasiert durch die gemeinsame Nutzung von Kooperationsdiensten. Hierzu kann beispielsweise das *Observer*-Entwurfsmuster auf gemeinsame Modelle angewendet werden (vgl. [GHJV96], S. 257f.).

Aus softwarearchitektonischer Sicht kann durch ein Groupware-Framework die Persistenz der Modelle sowie der objektorientierte Zugriff darauf über die Kooperationsdienste gesichert werden (vgl. Abb. 4.14). Damit diese Dienste auch in den Produktarchitekturen verfügbar sind, bedarf es darüber hinaus entsprechender Schnittstellen.

[30] Beispiele für die so genannte dynamische Attributierung finden sich in [HR05].

[31] Das *Model-View-Controller*-Paradigma wird dabei durch eine Assoziation eines verschiedenen Anwendungen gemeinsames Modell durch lokale Modelle dahingehend erweitert, dass die Anwendungen auf gemeinsamen Daten arbeiten, d. h. lokale Modelle halten eine Referenz auf das gemeinsame Modell und sind gleichzeitig an dessen Änderungen interessiert.

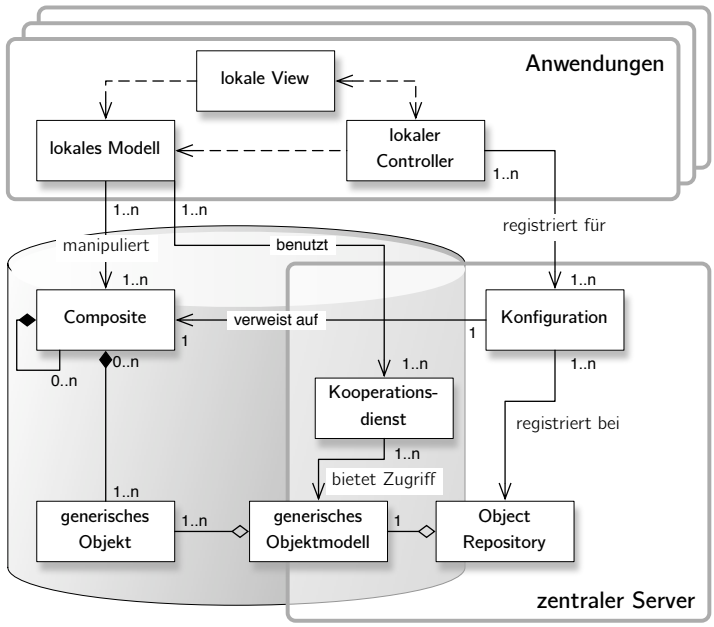

Abbildung 4.14.: Das verteilte MVC-Muster basiert auf zentral gehaltenen, gemeinsamen Modellen (Composites)

4.2.2.2. Schnittstellen zu Kooperationsdiensten

Um Instanzen gemeinsamer Modelle erstellen und manipulieren zu können, muss für die Zielsprache eine Bibliothek generischer Funktionen und Modellelemente existieren. In der technischen Ausgestaltung des Zugriffs auf Kooperationsdienste und Objekte gibt es jedoch architektonische Unterschiede, deren Vor- und Nachteile an dieser Stelle kurz skizziert werden sollen.

Remote Procedure Calls und Object Access Die Möglichkeit des entfernten Zugriffs (*remote access*) auf Funktionen und Objekte des Groupware-Frameworks über Web-Service-Schnittstellen ist neutral bzgl. der für die Anwendungsentwicklung verwendeten Programmiersprache. Die Anwendung kann – muss aber nicht – in der Laufzeitumgebung des Servers liegen. Der Kommunikationsaufwand zwischen Server und Client ist dabei abhängig vom verwendeten Zugriffsprotokoll. Wird bspw. ein SOAP-Framework eingesetzt, können Objekte aus der Laufzeitumgebung des Servers heraus mittels eines XML-Schemas serialisiert und als SOAP-Nachricht an den aufrufenden Client übertragen werden. Clientseitig werden die Daten mit Hilfe des gleichen Schemas i. d. R. wieder in Objekte transformiert, auf die innerhalb der Laufzeitumgebung des Clients dann zugegriffen werden kann. Sie fungieren dort allerdings nur als Datencontainer. Das Aufrufen von Methoden, die serverseitig an den jeweiligen Objekten verfügbar sind, ist nicht ohne weiteres möglich. Dies verhindert auch die Nutzung von softwaretechnischen Konzepten zur Spezialisierung und Komposition, mit denen ein Objektmodell auf der Klientenseite

erweitert werden kann. Während eine Erweiterung des ursprünglichen Modells auf dem Server problemlos möglich ist[32], muss auf Seite des Klienten ein zusätzlicher Aufwand betrieben werden, um aus den Datencontainern funktionale Fachobjekte zu generieren. Hierzu kann das Entwurfsmuster des Proxy-Objekts verwendet werden (vgl. [GHJV96], S. 227ff.).

Remote Proxy-Objekte Remote Proxy-Objekte leiten stellvertretend für ein Objekt innerhalb eines entfernten Adressraums Zugriffe auf Methoden und Attribute weiter an den Server. Sie kapseln also die Funktionen serverseitiger Objekte, und bieten `get`- und `set`-Methoden zum Auslesen und Schreiben von Attributen an. Die Kommunikation zwischen Proxy und Server kann dabei über standardisierte XML-basierte Protokolle laufen (bspw. SOAP). Von Vorteil ist bei der Verwendung von Proxy-Objekten die Verfügbarkeit des Objektmodells in Form von Konstrukten in der zur Entwicklung genutzten Programmiersprache. Die Funktionen sind damit direkt an den Objekten verankert und können auch in einer Klassenhierarchie vererbt werden. So kann selbst die Programmierung sehr intuitiv mithilfe der generischen Modellelemente erfolgen (vgl. [HR05]). Auch clientseitige Erweiterungen des Objektmodells durch Entwurfskonzepte wie Generalisierung und Komposition sind auf diesem Weg möglich, bleiben aus Sicht der Wiederverwendbarkeit jedoch auf das Produkt beschränkt.

An dieser Stelle muss daher abgewägt werden, ob die im Produkt zu implementierende Funktion in anderen Produkten der Softwarelinie wieder verwendet werden kann. Kann dies angenommen werden, ist es sinnvoller, ein gemeinsames Modell zu konfigurieren und die modellspezifischen Funktionen als höheren Dienst der Produktstandardarchitektur über Schnittstellen allen Produkten zur Verfügung zu stellen (s. nächster Abschnitt „E-Learning-spezifische Dienste“).

Klassenbibliotheken/Frameworks Braucht der Zugriff auf Kooperationsdienste nicht entfernt zu erfolgen, da zum Beispiel die zu entwickelnde Anwendung mit in der Laufzeitumgebung des Groupware-Frameworks ausgeführt werden kann, gibt es noch eine dritte Möglichkeit: Klassen und Funktionen des Objektmodells können auch in Klassenbibliotheken oder – spezieller – Frameworks zusammengefasst werden. Deren Verwendung kann auf zwei Arten erfolgen: (1) Objekte der Klassenbibliothek werden instanziert, (2) neue Klassen werden durch Vererbung von gegebenen Klassen der Bibliothek abgeleitet. Die Programmiersprache ist in diesem Fall fest vorgegeben. Dass die Kommunikation zwischen Anwendung und Server aber innerhalb derselben Laufzeitumgebung erfolgt, wirkt sich i. d. R. positiv auf den Kommunikationsaufwand und somit auch auf die Ausführungsgeschwindigkeit aus.

Aus technischer Sicht sind alle drei Ansätze gleich mächtig. Die Unterschiede liegen in dem der Entwicklung zugrunde liegenden Paradigma: Während die Verwendung von reinen Remote Procedure Calls und Object Access ein service-zentriertes Vorgehen

[32] Hierbei muss lediglich das die Schnittstelle beschreibende XML-Schema angepasst werden, um die Änderungen auch für Clients verfügbar zu machen.

impliziert, erlauben Proxy-Objekte und Klassenbibliotheken die Anwendung objektorientierter Konzepte und bieten somit auch Möglichkeiten der Modellerweiterung durch Spezialisierung und Komposition.

4.2.2.3. E-Learning-spezifische Dienste

E-Learning-spezifische Dienste können anhand ihrer Wiederverwendbarkeit entweder als höhere Dienste der Produktstandardarchitektur oder als Anwendungskomponenten der Produktarchitektur implementiert werden. Ersteres bietet sich an, wenn ein Dienst innerhalb der Softwareproduktlinie in mehreren Produkten wieder verwendet werden kann. Beispielsweise Komponenten zur Abfrage des Awareness-Status bestätigter Kontakte oder zur Unterstützung von Diskursstrukturen, um verschiedene, auf Diskursen beruhende Lernszenarien mit spezialisierten Anwendungen unterstützen zu können.

In beiden Fällen wird über die oben beschriebenen Schnittstellen auf kooperative Dienste zugegriffen, um gemeinsame Modelle auslesen und ggf. manipulieren zu können. Wird ein Dienst als höherer Dienst der Produktstandardarchitektur implementiert, entfällt jedoch die direkte Anbindung an eine Benutzungsschnittstelle; die Darstellungskomponenten und Komponenten zur Steuerung der Interaktion mit dem Benutzer sind Teil der Produktarchitektur. Statt dessen muss die Funktionalität über eine Komponentenschnittstelle zur Verfügung gestellt werden, damit sie für Anwendungskomponenten der einzelnen Produkte im Zugriff ist.

Wo E-Learning-spezifische Dienste in der Architektur verteilt sind, hängt zuerst von der Zuordnung zu Produktstandard- oder Produktarchitektur ab, jedoch auch von den eingesetzten Technologien. Lässt sich ein E-Learning-spezifischer Dienst im Rahmen der Produktstandardarchitektur über ein modulares Erweiterungskonzept des Groupware-Frameworks implementieren und sprechen keine anderen softwarearchitektonischen Prinzipien dagegen, ist diese Variante sicherlich zu bevorzugen (vgl. hierzu auch 151ff.).

Alternativ kann der Dienst unter Verwendung eines Application Frameworks realisiert werden. Entweder im Rahmen der Produktstandardarchitektur mit der skizzierten Komponentenschnittstelle oder – falls die Möglichkeit der Wiederverwendung nicht gegeben ist – als Anwendungskomponente der Produktarchitektur inklusive einer Benutzungsschnittstelle. Diese Möglichkeit wird im folgenden Abschnitt näher beschrieben.

4.2.3. Die Produktarchitektur

Softwareprodukte auf Basis der Produktstandardarchitektur implementieren eine Benutzungsschnittstelle für spezielle Anwendungskontexte des kooperativen Arbeitens und Lernens. Dazu werden Anwendungskomponenten benötigt, die über die Kooperations- und E-Learning-spezifischen Dienste eine synchron und asynchron kooperative Interaktion auf gemeinsamen, zentral persistierten Modellen implementieren.

Für die Entwicklung von Benutzungsschnittstellen kooperativer Systeme hat sich das Model-View-Controller (MVC)-Muster etabliert (vgl. [Ham02], S. 170f., [SS01], S. 306ff.). Dieses leitet aus der Interaktion mit einer Benutzungsschnittstelle drei verschiedene Rollen ab:

- ➢ Das *Modell* ist ein Teilbereich oder eine lokale semantische Interpretation eines gemeinsamen Modells. Im einfachsten Fall ein generisches Objekt, in Anwendungskontexten mit höherer Granularität kann ein lokales Modell auch mehrere gemeinsame Modelle umfassen. Aus softwarearchitektonischer Sicht muss die Produktarchitektur dazu ein lokales Modell von einem oder mehreren gemeinsamen Modellen instanzieren können.
- ➢ Eine *View* repräsentiert die Benutzungsoberfläche der Anwendung. Abhängig von den eingesetzten Technologien kann dies beispielsweise ein Fenster in einer Desktopanwendung sein oder eine dynamisch generierte Webseite, die über einen Web-Browser ausgeführt wird.
- ➢ Ein *Controller* steuert die Elemente der Benutzungsoberfläche, nimmt Benutzereingaben auf und manipuliert das Modell.

Das MVC-Muster impliziert eine Trennung von Zuständigkeiten an zwei Stellen der Produktarchitektur. Erstens trennt es die Belange des Modells von denen der Benutzungsoberfläche. Diese Trennung ist essentiell, damit verschiedene Sichten auf ein gemeinsames Modell geboten werden können. Sichten können dabei unterschiedliche Werkzeuge implementieren, die in Abhängigkeit von der jeweiligen Benutzerrolle und vom Anwendungskontext spezialisierte Funktionen der Modellmanipulation unterstützen.

Zweitens trennt das MVC-Konzept die Darstellung der Benutzungsschnittstelle von ihrer Steuerung. Diese separierende Betrachtung ist von Bedeutung, damit bspw. verschiedene Darstellungsformen mit ein und der gleichen Steuerung angeboten werden können. In Abschnitt 4.2.7.2 wird diese Trennung zum Entwurf einer Benutzungsschnittstelle für mobile Endgeräte benutzt (S. 164ff.).

Die Umsetzung des MVC-Musters im Softwaredesign ist dabei technologieabhängig. Moderne (Web-)Application-Frameworks unterstützen die Trennung von Modell, Darstellung und Steuerung mit unterschiedlichen Ansätzen.
Wendet man dieses Entwurfsmuster auf die Entwicklung verteilter Anwendungen auf Basis eines Objektframeworks an, lassen sich darüber die zentralen Komponenten der Produktarchitektur herleiten:

1. Ein entsprechendes *Application-Framework* zur Unterstützung der Anwendungsentwicklung. Als Teil einer Laufzeitumgebung bieten Application-Frameworks i. d. R. Trennung von Daten, Anwendungslogik und Präsentation (MVC-Konzept), einen Registrierungsdienst (dependency injection), Schnittstellen zu Basisdiensten der Infrastruktur wie Datenbanken, Middleware, E-Mail und Authentifizierung, sowie ein lokales Sitzungsmanagement und Lokalisierung,
2. *Schnittstellen* zu Kooperationsdiensten und E-Learning-spezifischen Diensten der Produktstandardarchitektur. Das kann – wie oben beschrieben – zum Beispiel ein Web-Service-Cient oder eine Klassenbibliothek sein, die die konfigurierten Modelle als Objektstruktur in der Laufzeitumgebung der Anwendung zur Verfügung stellen,
3. Eine *Instanz eines gemeinsamen Modells* (lokales Modell) als Grundlage kooperativer Manipulation,

4. *Anwendungskomponenten*, die auf Basis der Kooperations- und E-Learning-Dienste die lokale Instanz mit dem gemeinsamen Modell abgleichen und die Steuerung der Interaktion über die Benutzungsschnittstelle implementieren,

5. Eine *Darstellungsschicht*, die der Darstellung gemeinsamer Modelle und der durch sie repräsentierten Dokumente, Inhalte und Wissensstrukturen dient. Darüber hinaus erlaubt sie auch die synchrone und/oder asynchrone kooperative Interaktion mit den jeweils dargestellten Artefakten.

Das Application Framework als Teil der Laufzeitumgebung soll im Kontext der Produktarchitektur nicht näher spezifiziert werden. Auf die Instanzierung gemeinsamer Modelle in der lokalen Anwendungsschicht soll jedoch im Folgenden näher eingegangen werden. Ebenfalls auf architektonische Belange der Anwendungskomponenten und der Darstellungsschicht.

4.2.3.1. Konstruktion lokaler Modelle

Die generischen und konfigurierten Modelle des Groupware-Frameworks bilden in verschiedenen Anwendungen die Basis des kooperativen Arbeitens und Lernens. In der Produktarchitektur müssen diese gemeinsamen Modelle den Anwendungskomponenten als Objekte zur Manipulation zur Verfügung stehen, die Anwendungslogik also durch eine anwendungsspezifische Objektschicht fundiert werden.

Die dazu benötigten Schnittstellen zu höheren Diensten der Produktstandardarchitektur wurden im letzten Abschnitt bereits erläutert. Auf Basis der dadurch zur Verfügung stehenden Objekte und Funktionen können die gemeinsamen Modelle der Produktstandardarchitektur im Sinne eines lokalen Modells anwendungsspezifisch interpretiert und erweitert werden. Die Sichten verteilter Anwendungen können Modelle dabei in unterschiedlicher Granularität darstellen. Ein speziell auf ein bestimmtes Lernszenario zugeschnittenes Werkzeug bedient beispielsweise nur einen bestimmten Ausschnitt eines gemeinsamen Modells, während größere Lernplattformen verschiedene gemeinsame Modelle in einem lokalen Modell zusammenführen. Zwei Softwaretechniken können in diesem Kontext angewendet werden:

Anwendungsspezifische Erweiterung Neue Anwendungen dieser Domäne können über Instanzierung, Vererbung und Spezialisierung von Klassenbibliotheken oder Proxy-Objekte in der Ziel-Programmiersprache umgesetzt werden. Das bedeutet, dass die neuen Objektklassen vollständig auf den generischen Elementen der gemeinsamen Modelle basieren; sie erweitern sie und erben daher alle ihre Funktionen und Parameter, während die Möglichkeit besteht, sie um darüber hinausgehende Funktionen und Parameter zu ergänzen.

Anwendungsspezifische Interpretation Der andere Weg ist, ein neues, unabhängiges Set an anwendungsspezifischen Klassen zu entwickeln und darin bloß eine Referenz auf generische (Proxy-)Objekte zu verwalten. Auf der einen Seite bedeutet dieser Weg

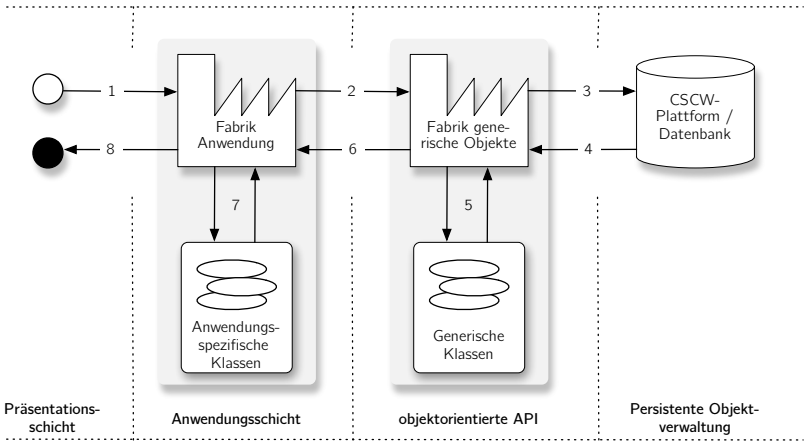

Abbildung 4.15.: Instanzierung von Klassen in der Objektschicht der Anwendung über Fabriken.

zunächst einen größeren Aufwand für Entwurf und Implementierung, die so genannten *Wrapper*-Methoden für die generischen Objekte zu schreiben, da die Methoden der Proxy-Objekte bei diesem Vorgehen nicht einfach geerbt werden können. Auf der anderen Seite erlaubt dieses Vorgehen die Gestaltung einer eigenen Vererbungshierarchie zwischen den anwendungsspezifischen Klassen. Ein solcher Vererbungsbaum kann vom Basis-Objekt angefangen neu geschaffen werden und erleichtert die Adaption spezifischer Domänen- oder Anwendungskonzepte.

Die Proxy-Objekte respektive die Klassenbibliothek umfasst in der Regel eine Fabrik, die jedoch ausschließlich Instanzen generischer (Proxy-)Objekttypen erzeugen kann. Soll das lokale Modell eine Interpretation oder Erweiterung eines gemeinsamen Modells sein, wird neben einem neuen Set an Klassen auch eine anwendungsspezifische *Fabrik* benötigt, an die die Erzeugung der neuen Objekte delegiert werden kann. Dieses klassenbasierte Erzeugungsmuster kapselt das Wissen um die zu erzeugenden Klassen und lagert es aus der Standardproduktarchitektur aus (vgl. Abbildung 4.15).

4.2.3.2. Ausführung und Manipulation von Modellen

Anwendungskomponenten erlauben eine automatisierte und interaktive Nutzung gemeinsamer Modelle im Kontext der Anwendung. Dazu implementieren sie bspw. Methoden, die die Strukturen und Mechanismen der Modelle verdeutlichen, oder solche, die eine aufgabenorientierte Manipulation realisieren.

Eine Anwendungskomponente zum Sitzungsmanagement baut dazu die Verbindung mit den höheren Diensten der Produktstandardarchitektur auf, meldet den Benutzer am Groupware-Framework an und koordiniert synchrone Funktionen und Darstellungen zwischen kooperierenden Benutzern.

Abhängig vom Typ der zu unterstützenden Darstellung übernehmen die Anwendungskomponenten dabei unterschiedliche Aufgaben der Interaktionskontrolle. Beispielsweise können in einer direktmanipulativen Anwendungsschnittstelle die Elemente lokaler Modelle rein visuell dargestellt werden. Bei diesem Ansatz werden Technologien ereignisbasierter, virtueller Welten mit Fähigkeiten zum netzgestützten, kooperativen Arbeiten und Lernen verknüpft. Die direkte Manipulation kann dann durch das *Object Action Interaction-Modell* (OAI) direkt innerhalb der Darstellung erfolgen (vgl. [Shn83]).

In Web-basierten Benutzungsschnittstellen liegt der Fokus der Interaktionssteuerung dagegen eher auf der Ansteuerung von HTML-Schablonen (der *View*) und der Auswertung von HTTP-Requests. Darüber können Interaktionen ausgewertet werden, die der Benutzer im Browser getätigt hat, und das Modell entsprechend manipuliert werden.

Durch das kooperative MVC-Muster ist die Objektschicht verteilter Anwendungen direkt oder indirekt[33] mit den Kooperationsdiensten des Groupware-Frameworks verbunden. Diese Assoziation von lokalen Modellen mit zentral verwalteten gemeinsamen Modellen erweitert das MVC-Konzept im Sinne einer zentralisierten Architektur, wodurch synchrone Kooperationen erst ermöglicht werden (vgl. [Ham02], S. 171f.).

Die gemeinsamen Modelle kapseln dazu sowohl domänenspezifische Modelldaten, die für alle Anwendungen von Interesse sind, als auch spezifische Daten einzelner Anwendungen, die diese zur Darstellung und Ausführung des Modells persistent benötigen. Wird ein Modell in einer gemeinsamen Sitzung durch eine Anwendung manipuliert, müssen alle verteilten Anwendungen der Sitzung über den neuen Zustand des Modells informiert werden, wenn allgemeine Daten geändert wurden. Darüber hinaus müssen alle verteilten Anwendungen des gleichen Typs informiert werden, wenn sich für diesen Anwendungstyp spezifische Modelldaten geändert haben, bspw. Koordinaten zur Darstellung eines Objektes in einem Shared Whiteboard.

Die Event-basierte Benachrichtigung über Zustandsänderungen von Objekten kann softwaretechnisch mithilfe des *Observer-Patterns* umgesetzt werden (vgl. [GHJV96], S. 257f.). Dazu können Anwendungskomponenten der Produktarchitektur so genannte *Listener*-Methoden implementieren, die ausgelöst werden, wenn sich das Modell ändert.

4.2.4. Verteilung

Die klare Trennung der Schichten und die Nutzung von Entwurfsmustern wie Dependency Injection und dem kooperativen MVC-Konzept begünstigen einfach gerichtete Abhängigkeiten zwischen den Komponenten. Dies lässt mehrere Verteilungsmöglichkeiten auf Laufzeitumgebungen zu. Denkbar ist eine so genannte „Fat Server/Thin Client"-Verteilung, bei denen alle Komponenten bis zur Darstellungsschicht in einer Umgebung laufen. Durch die klare Trennung der Schichten nach Wiederverwendungsgrad bieten sich jedoch insbesondere solche Verteilungsstrategien an, welche die Komponenten der Produktstandardarchitektur zentral bereitstellen und – entsprechend der zu erwarteten Mehrbelastung – in einer für die Last optimierten Laufzeitumgebung installieren.

[33] Neben der direkten Verbindung können Komponenten der Produktarchitektur wie beschrieben über die E-Learning-Dienste mit den Kooperationsdiensten verbunden sein.

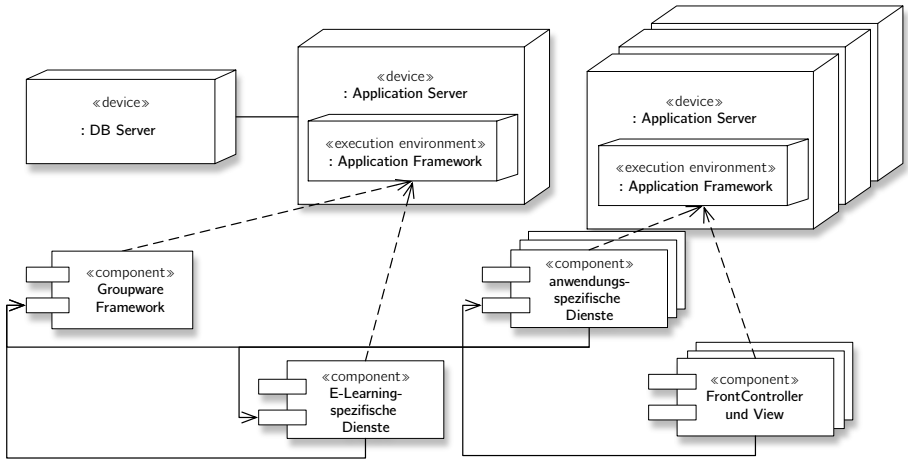

Abbildung 4.16.: Mögliche Verteilung der Architekturkomponenten

Im Sinne der Hochverfügbarkeit dieser in der Architektur relativ häufig nachgefragten Komponenten können die E-Learning-spezifischen Dienste auch in einer von dem Groupware-Framework separaten Laufzeitumgebung betrieben werden. So können sowohl die Kooperationsdienste als auch die E-Learning-spezifischen Dienste redundant ausgelegt und parallel betrieben werden.

Die individuellen Komponenten der darauf basierenden Anwendungen können nach Betreibermodell verteilt werden, beispielsweise dezentral auf Servern der einzelnen Fakultäten. Abbildung 4.16 beschreibt eine mögliche Verteilung, bei der die Komponenten der Produktarchitekturen auf mehrere Laufzeitumgebungen verteilt sind.

4.2.5. Integration in eine universitäre IT-Infrastruktur

Die aktuellen Bestrebungen nach einem integrierten Informationsmanagement an Hochschulen und die daraus resultierenden Anforderungen an Schnittstellen wurden bereits einleitend in Kapitel 2 motiviert (S. 17f., S.26ff.).

In diesem Unterkapitel sollen Lösungskonzepte für wichtige Schnittstellen auf Basis von Standardprotokollen erarbeitet und in die Rahmenarchitektur mit aufgenommen werden. Beispielhafte Szenarien sind die Anbindung an Campus Management- und Bibliothekssysteme. Unterkapitel 4.2.6 beschreibt darüber hinaus die Integration externer Knowledge Pools, die auch zur internen Anbindung von Content Repositories an die hier konzipierte verteilte Architektur genutzt werden kann.

Letztendlich ist die Anwendungsschicht der Ausgangspunkt für die Herstellung von Interoperabilität zwischen Systemen. Je mehr Dienste zwei oder mehr Systeme gemeinsam haben, desto umfassender können sie integriert werden.

Abbildung 4.17 zeigt die unterschiedlichen Schichten, die für eine Interoperabilität von verteilten Systemen eine Rolle spielen. Um Interoperabilität zu erreichen, muss es für jede Schicht mindestens eine gemeinsame Komponente geben. Die gewünschten Dienste

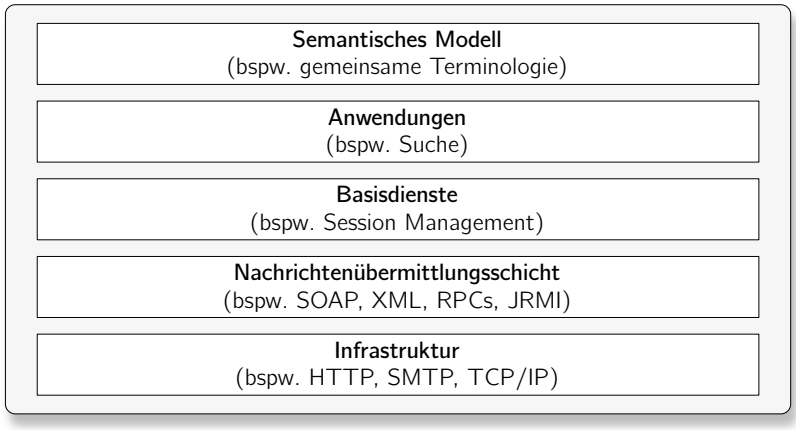

Abbildung 4.17.: Interoperability Stack

beeinflussen, welche Strukturen im semantischen Modell benötigt werden. Diese können bspw. in Form einer gemeinsam verwendeten domänenspezifischen Terminologie vorliegen oder durch ein anderes gemeinsames Datenschema wie standardisierte Anfrage- und Ergebnisformaten.

Auf der Schicht der Basisdienste muss ein gemeinsam zu nutzender Authentifizierungsdienst vorhanden sein. Die Anwendungsdienste sollten dabei aber so unabhängig gestaltet sein, dass sie von keiner speziellen Umsetzung dieser Komponenten abhängen, sondern nur die Existenz dieser Komponenten erfordern. Dadurch wird die Anpassbarkeit an andere Systeme erhöht.

4.2.5.1. Anbindung an Verwaltungssysteme

Bei der Planung und beim Prüfungsmanagement übernimmt die Hochschulverwaltung mit Hilfe einer zentralen, stark prozessorientierten Systemlandschaft unterstützende Funktionen, wie beispielsweise die Erstellung des Vorlesungsverzeichnisses oder Studienkontenverwaltung. Im Gegensatz dazu liegt die Durchführung der Lehre in der Hand von Lehrstühlen, die i. d. R. teilautonom über den Einsatz von Computerunterstützung zu diesem Zweck frei entscheiden können. Um zusätzliche Mehraufwände für Administration und Datenerfassung und damit verbundene Verzögerungen bei alltäglich wiederkehrenden Nutzungsszenarien zu vermeiden, müssen identische Daten z. B. von Personen und Lehrveranstaltungen zwischen Verwaltungs- und elektronischen Lernumgebungen konsistent gehalten werden.

Die Alternativen für die technische Umsetzung der Integration wurden in diesem Kapitel bereits vorgestellt. Die abzubildenden Nutzungsszenarien werden in [DEH+02], [KNW03], [BFH+04] und [GR07] skizziert und weisen auf eine asynchrone, lose Kopplung der Systeme hin, da zwischen den Systemen oftmals nur Veränderungen[34] abgeglichen werden müssen (vgl. Tabelle 4.3).

[34]Die Veränderung (Delta) lässt sich als Differenz der Datenbestände der abzugleichenden Systeme ermitteln, bspw. zur Ermittlung der seit der letzten Synchronisation neu erfassten Kursbuchungen $\Delta = AlleKursbuchungen - BereitsÜbermitteltеKursbuchungen$.

Eine synchrone Kommunikation über entfernte Funktionsaufrufe oder verteilte Objekte würde in dem Fall die Komplexität und den Kopplungsgrad der Schnittstelle unnötig erhöhen und zu Mehrkosten vor allem in der Wartung und Weiterentwicklung führen (vgl. [RHG05], S. 70, [HPP07], S. 3). Eine direkte Kopplung sollte daher vermieden werden.

Verwaltungssystem	**virtuelle Lernumgebung**
Veranstaltung anlegen	➥ Kursraum und Gruppen anlegen und beschreiben
Dozenten zuordnen	➥ Berechtigungen für Dozenten auf Kursraum einrichten
Teilnahmebeschränkung einrichten	➥ Gruppenzutritt beschränken
Studenten zu Veranstaltung zulassen	➥ Studenten in Teilnehmergruppe aufnehmen
Studenten von Veranstaltung abmelden	➥ Studenten aus Teilnehmergruppe ausschließen

Tabelle 4.3.: Beispiele gemeinsamer Nutzungsszenarien von Verwaltungs- und Lernsystemen

Daher bietet sich eine nachrichtenbasierte Kommunikation über eine Integrationsplattform respektive über ein Messagebussystem an, wobei das für einen Prozess führende System – also das, in dem die Daten erhoben werden – definiert sein muss. Das Messagebussystem implementiert eine Warteschlange für Nachrichten und garantiert damit die lose Kopplung in der 1:n- bzw. n:m-Kommunikation zwischen den beteiligten Systemen. Nachrichtenerzeugung und -verarbeitung können als Prozesse somit zeitlich getrennt werden, was sowohl eine synchrone als auch asynchrone Kommunikation ermöglicht. Die Anbindung der Systeme an den Messagebus wird in der Regel durch Konnektoren hergestellt, die im Einzelfall individuell implementiert werden müssen.

Zunächst wird für das führende System die Schnittstelle definiert. Dadurch wird der Funktionsumfang festgelegt, sowie das Format der auszutauschenden Daten und das Kommunikationsmodell (synchron/asynchron). Gleichzeitig werden somit die Vorgaben an die Gegenseite spezifiziert. Die Schnittstelle sollte dabei fachlich weit genug gefasst sein, um nicht nur den spezifischen Anforderungen einer einzigen Plattform zu genügen, sondern ein möglichst breites Spektrum an Szenarien abbilden[35]. Dies minimiert die Wahrscheinlichkeit einer Änderung der Schnittstelle bei Anbindung weiterer Systeme.
Ist das sendende System auf eine Antwort vom empfangenden System angewiesen, lässt sich ein asynchrones Kommunikationsmodell implementieren. Diese Art der Schnittstelle zeichnet sich durch eine hohe Performance aus, da der sendende Thread sofort

[35] Das gilt auch für den Fall, dass im ersten anzukoppelnden System Änderungen nötig sind oder interne Prozesse geändert werden müssen. Dies sind einmalige Aufwendungen, die kalkulierbar sind (vgl. [RHG05], S. 70).

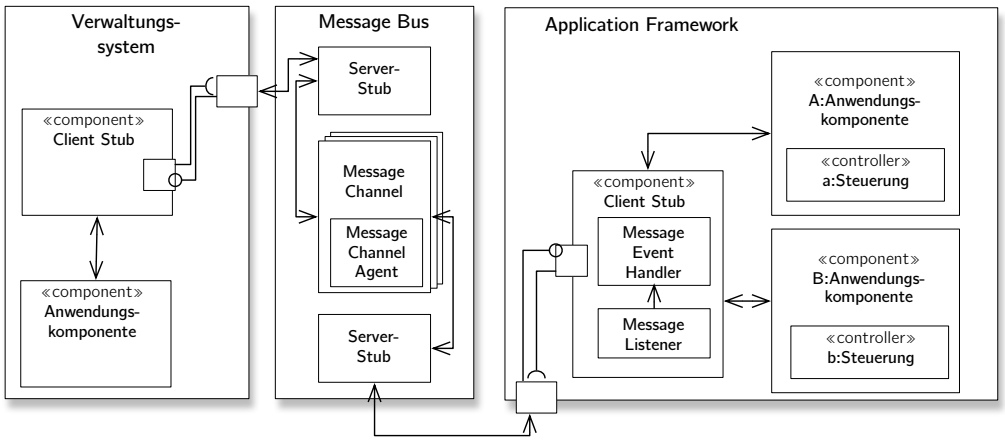

Abbildung 4.18.: Die Messagebus-Schnittstelle

weiterarbeiten kann. Weiterhin stabilisiert eine zwischengeschaltete Warteschlange den Kommunikationsweg. Moderne Messagebussysteme garantieren auch dann eine Zustellung der Nachricht, wenn das zu empfangende System zum Zeitpunkt der Sendung nicht erreichbar ist. Die Nachricht wird dann in der Warteschlange vorgehalten und zugestellt, wenn das System wieder empfangsbereit ist.

Ein synchrones Kommunikationsmodell muss hingegen gewählt werden, wenn der sendende Thread zum Weiterarbeiten ein Ergebnis vom Empfänger benötigt, wie bspw. eine differenzierte Statusmeldung über die Transaktion.

Die Definition der Schnittstelle ist produktspezifisch und somit innerhalb der Rahmenarchitektur der Produktarchitektur zuzuordnen. Der Konnektor zum Messagebussystem bzw. zur Integrationsplattform kann hingegen der Produktstandardarchitektur zugeordnet werden (vgl. Abb. 4.18). Dieser Konnektor implementiert einen so genannten *Message Listener*, der anhand einer Analyse des Nachrichtenkopfes eine empfangene Nachricht an einen für den Nachrichtentyp zuständigen *Message Event Handler* weiterleitet. Da Nachrichtentyp und -behandlung anwendungsspezifisch sind, muss diese Zuordnung aus einer Konfiguration ausgelesen werden.

Die eigentliche Verarbeitung der Nachricht findet im *Message Event Handler* statt. Hier werden aus der Nachricht Objekte erzeugt und darauf Funktionen aufgerufen. Im Falle einer synchronen Kommunikation generiert ein *Event Handler* nach der Verarbeitung eine Antwort für das aufrufende System, die über den Konnektor auf den Messagebus gelegt wird. Damit der ursprüngliche Sender die Antwort der gesendeten Nachricht zuordnen kann, muss die Antwort einen entsprechenden Schlüssel referenzieren.

Grundsätzlich können über eine nachrichtenorientierte Middleware nicht nur Lernumgebungen an Verwaltungssysteme angebunden werden. Eine solche Integrationsplattform kann als zentraler Dienstvermittler innerhalb einer gesamtuniversitären IT-Infrastruktur Datenbanken, Web-Dienste und zum Teil proprietäre Anwendungsschnittstellen prozess-

gesteuert zusammenschalten. Durch das *Publish-Subscribe*-Verfahren auf Basis von spezifischen Nachrichtenkanälen bietet sich darüber hinaus – neben einem zentralen Identitäts- und Rechtemanagement – ein zentraler Ansatzpunkt für das Rechtemanagement auf Dienstebene. Hier kann gesteuert werden, welche Systeme über welche Nachrichten in welchem Umfang informiert werden[36].

4.2.5.2. Anbindung an Bibliothekssysteme

Ein wichtiger Bestandteil des wissenschaftlichen Arbeitens an Universitäten ist die Literaturrecherche, also das Suchen, Auswählen, Beschaffen und Verarbeiten wissenschaftlicher Quellen. Insbesondere das Suchen in Onlinedatenbanken und das Auswählen und Speichern relevanter Ergebnisse in einer eigenen Literaturliste sollte ohne Systemwechsel durch die alltäglich genutzte Lern- und Arbeitsumgebung unterstützt werden.

Dazu soll im Folgenden – gemäß den Entwurfsprinzipien verteilter Architekturen – ein geeignetes Standardprotokoll identifiziert und eine dazu geeignete Schnittstelle in die Rahmenarchitektur aufgenommen werden.

Das Standardprotokoll im Bibliotheksbereich zur Abfrage von bibliographischen Informationssystemen ist die internationale Spezifikation ANSI/NISO Z39.50-2003: „Information Retrieval (Z39.50): Application Service Definition and Protocol Specification“ (vgl. [NIS02]). In den 90er Jahren förderte der Bund und der DFG unter der Federführung der Deutschen Bibliothek das Verbundprojekt „DBV-OSI-II“, das auch viele deutsche Datenbanksysteme um die Z39.50-Schnittstelle erweitert hat (vgl. [Luc96]). Das hatte zur Folge, dass heute – mehr als zehn Jahre später – viele Internetportale und Anwendungen über diese Schnittstelle den Zugriff auf bibliographische Online-Kataloge realisieren[37].

Aus technischer Sicht setzt Z39.50 als sitzungsbasiertes Anwendungsprotokoll auf dem Internet-Protokoll TCP/IP auf. Die Spezifikation legt die Funktionen fest (siehe Tab. 4.4), sowie die Operatoren und Übertragungsformate.

Funktion	**Erklärung**
`INITIALIZE`	Authentifizierung des Clienten, Eröffnung einer Z39.50-Session
`SEARCH`	Suchanfragen, Query, Name von Result-Sets und Format
`PRESENT`	Übertragung von Suchergebnissen
`DELETE_RESULT_SET`	Löschen von Result-Sets beim Zielsystem
`RESOURCE_REPORT`	Abfrage der angefallenen Kosten/Abrechnung
`SCAN`	Suche in geordneten Term-Listen
`CLOSE`	Schließen der Z39.50-Session

Tabelle 4.4.: Basisfunktionen des Z39.50 Protokoll

[36] Integrationsplattformen bieten z. T. die Möglichkeit, Nachrichteninhalte für bestimmte Systeme gezielt aufzubereiten und daher Informationen für bestimmte Adressaten auch einzuschränken.

[37] Beispiele: Digitale Bibliothek des Hochschulbibliothekszentrum NRW (`http://rhea.hbz-nrw.de/Digibib`), das Portal p7 des Gemeinsamen Bibliotheksverbund (`http://p7.gbv.de`), oder die Literaturverwaltung im Web 2.0 namens LibraryThing (`http://librarything.de`). Letzter Zugriff: 31.07.2008.

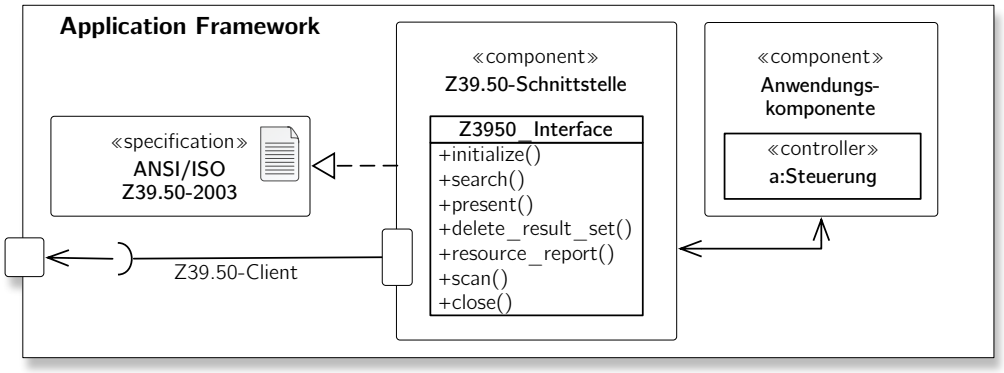

Abbildung 4.19.: Die Z39.50-Schnittstelle in der Rahmenarchitektur

Da der Zugriff auf Online-Kataloge in verschiedenen Lern- und Arbeitsanwendungen nützlich sein kann, ist die Z39.50 Schnittstelle in der Produktstandardarchitektur verortet. Somit kann sie in den darauf basierenden Implementierungen unterschiedlich genutzt werden. Die Schnittstelle impliziert architektonisch einen clientseitigen *Stub*[38], der die Komplexität der Z39.50-Spezifikation verbirgt und als Funktionen den Anwendungskomponenten lokal verfügbar macht. Die Konfiguration der Endstellen, d. h. ob bibliographische Kataloge im Kontext der Anwendung fest vorgegeben oder manuell durch den Benutzer ergänzt werden können, muss in der Produktarchitektur vorgenommen werden. Ebenso anwendungspezifisch ist die Art der Weiterverarbeitung der Ergebnisse. Auch sie muss in der Anwendungslogik der einzelnen Produkte individuell programmiert werden.

4.2.6. Anbindung an interne und externe Content-Provider

Die in Kapitel 2 bereits beschriebene internationale und nationale Bildungspolitik hat mit ihren Förderprogrammen viele Verbundprojekte unterstützt, die Lerninhalte hochschul- und sogar länderübergreifend zur allgemeinen Verwendung erstellt und ausgetauscht haben. Darüber sind über die Jahre hinweg große Knowledge-Pools erwachsen, die aus Netzwerken verteilter Lernobjekt-Repositories mit Inhalten und Metadaten bestehen. Zwei der größten Projekte sind in diesem Kontext die *ARIADNE Foundation for the European Knowledge Pool*[39] und das internationale *Global Learning Objects Brokered Exchange* Consortium (GLOBE)[40].

[38] Ein Stub (deutsch: Stummel, Stumpf) kapselt Funktionalität, die auf einem anderen Rechner oder in einem anderen Speicherbereich ausgeführt wird und sonst nur über komplexe Protokolle zu erreichen ist. Somit entspricht ein Stub dem in dieser Arbeit bereits vorgestellten Entwurfsmuster „Remote Proxy“ (Stellvertreter, vgl. S. 146).

[39] Mehr Informationen unter `http://www.ariadne-eu.org/`. Letzter Zugriff: 31.07.2008.

[40] Mehr Informationen unter `http://globe.edna.edu.au`. Letzter Zugriff: 31.07.2008.

Mithilfe dieser verteilten Architekturen lässt sich einerseits das eigene Angebot um Lerninhalte erweitern. Ein föderierter Suchdienst [41] kontaktiert das oder die ausgewählten Netzwerke mit der gewünschten Suchanfrage und liefert Metadatensätze über die Treffer zurück. Diese können dann abgerufen und in das eigene Content Repository übernommen werden. Andererseits können über die Netzwerke auch eigene Inhalte zur freien Wiederverwendung anderen Hochschulen und Dozierenden zur Verfügung gestellt werden. Dazu muss das eigene Repository über definierte Schnittstellen an den Verbund angeschlossen werden, damit lokale Inhalte auch für entfernte Benutzer über die föderierte Suche aufgefunden und geladen werden können.

Diese beiden Szenarien – der Betrieb des lokalen Repository als Client und als Provider eines föderierten Knowledge Pools[42] – sollen durch die Produktstandardarchitektur unterstützt werden können. Dazu wird die Rahmenarchitektur im Folgenden um eine entsprechende Schnittstelle nach dem SQI-Standard (*Simple Query Interface*) erweitert[43].

4.2.6.1. Das Simple Query Interface (SQI)

Gerade bei der Unterstützungen im E-Learning gibt es unterschiedlichste aktuelle Ansätze zur Verwaltung von Wissens- bzw. Lernobjekten, wie bspw. die Verteilung über Peer-to-Peer- oder semantische Netze. Dies führt zu einer hohen Heterogenität von Lernrepositories und damit zu erschwerter Interoperabilität.
Aus diesem Grund wurde im Rahmen von CEN/ISSS Learning Technology Workshops mit SQI eine Schnittstelle zur Konfiguration und Übertragung von Anfragen an Lernrepositories beschrieben[44] (vgl. [S+05]). Auf Ebene der Basisdienste spezifiziert SQI dazu ein Sitzungsmanagement, welches sich um Authentifizierung kümmert und das von höherwertigen Diensten genutzt werden kann. Auf Ebene der E-Learning-spezifischen Dienste beschränkt sich SQI auf die Spezifikation eines Query-Dienstes, der die Interoperabilität für Anfragen zwischen heterogenen Systemen herstellt. Dabei ist weder das semantische Modell der Anfrage näher spezifiziert, noch wird eine Aussage über Infrastruktur und Nachrichtenübermittlungsschicht getroffen. SQI ist daher kompatibel mit unterschiedlichsten Netzwerkprotokollen und Datenformaten. Diese semantische Neutralität führt aber dazu, dass zusätzliche Vereinbarungen zwischen kooperierenden Systemen getroffen werden müssen, wie über die Menge an Attributen und Vokabularien, die

[41] Zum Beispiel die föderierte Suche SILO (*Search & Index Learning Objects*) unter `http://ariadne2.unil.ch/SiloAriadne/`. Letzter Zugriff: 31.07.2008.

[42] Dabei ist zu beachten, dass es sich durchaus auch um hochschulinterne Knowledge Pools und Content Repositories handeln kann, wie bspw. Bibliotheks- und Dokumentenserver oder Learning Object Repositories.

[43] Obwohl die IMS Spezifikation *Digital Repositories Interoperability* (IMS DRI) die Z39.50-Nachfolger *Search and Retrieve Web Service* (SRW) und *Search and Retrieve via URL* (SRU, s. `http://www.loc.gov/z3950/agency/`) sowie das Protocol for Metadata Harvesting (OAI_PMH, vgl. `http://www.openarchives.org/OAI/openarchivesprotocol.html`) empfiehlt, scheint sich der SQI-Standard zumindest für Europäische Knowledge Pools durchzusetzen.

[44] Zwar wurde SQI für die Domäne E-Learning entwickelt, kann jedoch problemlos auf andere Inhalte wie bspw. Videos adaptiert werden. So lassen sich bspw. Inhalte der Plattform Youtube.com über SQI finden und zurückgeben. Das entsprechende WSDL-Binding für SQI ist online unter `http://sqi-wsdl.sourceforge.net/` verfügbar.

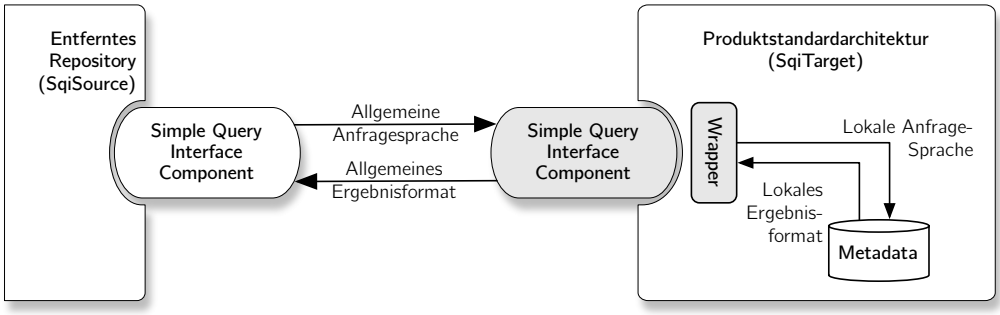

Abbildung 4.20.: Die SQI-Schnittstelle der Produktstandardarchitektur

in einer Anfrage genutzt werden können, die Anfragesprache und deren Repräsentation, sowie die Metadatenformate der Ergebnisses. Ein Mechanismus zur Abfrage oder Aushandlung dieser Parameter wird von der SQI-Spezifikation nicht angeboten.

Sind diese Vereinbarungen getroffen, wird u. U. noch eine Transformation zwischen den vereinbarten Formaten und den im lokalen Repository genutzten Formaten benötigt (*Wrapper*). Der grundsätzliche Ablauf einer Anfrage würde somit wie in Abbildung 4.20 dargestellt aussehen.

Eine weiterer Bestandteil der SQI-Spezifikation ist die Möglichkeit der asynchronen zusätzlich zur synchronen Anfrage. Bei einer asynchronen Anfrage wird deren Beantwortung vom angefragten System initiiert. Dieses ruft – sobald eine bestimmte Anzahl an Ergebnissen vorliegt – eine bereitgestellte Methode des aufrufenden Systems auf und übermittelt ihr die vorliegenden Ergebnisse. Das aufrufende System muss nach dem Absenden der Anfrage nicht auf das Ergebnis warten, sondern braucht erst beim Eintreffen neuer Ergebnisse darauf zu reagieren. Dies hat vor allem Vorteile bei einer gleichzeitigen Anfrage mehrerer Repositories oder ganzer P2P-Netzwerke, da die Antwortzeiten hierbei stark variieren und auch der Ausfall eines Knotens in einem P2P-Netzwerk nicht ungewöhnlich ist.

4.2.6.2. Anfragesprachen und Ergebnisformate

Suchanfragen können mithilfe von Anfragesprachen formuliert werden. Sie stellen Syntax und Semantik zur Verfügung, mit denen anzufragenden Systemen mitgeteilt werden kann, welche Daten in welchem Zusammenhang relevant sind. Eine Anfragesprache gilt dabei als ausdrucksstärker, je klarer relevante von nicht relevanten Daten abgegrenzt werden können. Dabei muss die Mächtigkeit und die Benutzbarkeit, d. h. die Verständlichkeit und Erlernbarkeit, gegeneinander abgewogen werden. Für die Auswahl der Anfragesprache spielt zusätzlich zur Ausdrucksstärke und Benutzbarkeit vor allem ihre Verbreitung eine entscheidende Rolle, wenn mit anderen Systemen kommuniziert werden soll. Beispiele für Anfragesprachen sind XQuery, CQL und VSQL (s. Anhang A.1).

Für das Ergebnisformat gilt Ähnliches wie für die Anfragesprache. Auch in diesem Bereich lässt SQI gewollt offen, in welcher Form die Ergebnismenge beschrieben wird. Für eine Implementierung gibt es demnach mehrere Alternativen.

Bekannte Metadatenformate wie *Dublin Core* oder *IMS LOM* (vgl. S. 2.2.4.2) unterscheiden sich in Art und Menge der Metadaten, als auch in Syntax und Semantik der Repräsentation. Entscheidungskriterien bei der Auswahl des Ergebnisformats sind daher neben einem hohen Verbreitungsgrad (1) die Übertragbarkeit auf systemintern genutzte Repräsentationsformate, sowie (2) die Möglichkeit der Weiterverarbeitung:

1. Wird das lokale Repository angefragt, muss das Ergebnis in das gewünschte Austauschformat transformiert werden. Bei dieser Zuordnung müssen nicht alle internen Attribute zwingend eine Entsprechung im Austauschformat finden. Es sollten jedoch genug Informationen geboten werden, damit interne Objekte von außerhalb zugreifbar sind,

2. Soll die Ergebnismenge einer föderierten Suche lokal dargestellt werden, also über Produkte der Standardarchitektur, so sollte das Austauschformat entsprechend der jeweiligen Benutzungsschnittstelle genügend Informationen bieten, damit entfernte Objekte analog zu den lokal vorgehaltenen Inhalten angezeigt werden können.

4.2.6.3. Architekturkonzept

SQI spezifiziert eine Schnittstelle im Sinne einer Funktionsintegration. Das bedeutet, dass alle SQI-konformen Systeme die definierten Funktionen anbieten und nutzen. Somit wird die Portabilität von Funktionen respektive Interoperabilität im Sinne des Informations- und Dokumentenaustauschs gesichert. Besitzt das System also per se die von SQI geforderte Funktionalität[45], müssen diese nur noch durch Ausimplementierung der Schnittstellenspezifikation und eines Transformators (*Wrapper*) explizit verfügbar gemacht werden (vgl. Abb. 4.21).

Zur Implementierung einer SQI-Unterstützung muss die Produktstandardarchitektur sowohl als Web-Service Client als auch als Provider auftreten können. Um einen SQI-Client zu realisieren, müssen eine Reihe von SQI-Methoden aufgerufen werden. Der grundsätzliche Ablauf bei der Nutzung von SQI teilt sich dabei in drei Abschnitte[46] (vgl. [SMD04], [SMVAD05]):

Session Management Für den Methodenaufruf wird eine eindeutige Session-ID benötigt, die auf Basis eines Berechtigungsnachweises (*Credentials*, bspw. Login-Daten) erzeugt wird[47].

Query Parameter Configuration Anfrageparameter gelten bis zum Ende einer Sitzung oder bis sie neu gesetzt werden. Parameter, die sowohl für synchrone als auch asynchrone Anfragen gelten, sind die Anfragesprache, die Anzahl der Ergebnisse in einer

[45] Also die Suche in Metadaten und Rückgabe von Ergebnissen (*Search/Expose*), sowie das Abrufen und die Auslieferung von Objekten (*Request/Deliver*). Nach [IMS03a] sind dies Standardfunktionen von Content-Repositories und somit in der Regel vorhanden.

[46] Zusätzlich dazu sind mit so genannten SQIFaults Codes für den Fehlerfall spezifiziert.

[47] Es besteht auch die Möglichkeit einer anonymen Session-Erzeugung mit vom Zielsystem oftmals eingeschränkten Berechtigungen.

Ergebnismenge, die Gesamtzahl der Ergebnisse, die maximale Ausführungszeit der Suche und das Ergebnisformat.

Synchronous/Asynchronous Query Interface Bei der eigentlichen Anfrage wird unterschieden zwischen der synchronen Anfrage, die als Rückgabewert das Anfrageergebnis hat, und der asynchronen Anfrage. Bei dieser zweiten Variante wird eine Rückantwort asynchron an eine bei der Anfrage angegebene Methode des aufrufenden Systems gesendet.

Notwendig für die Implementierung von asynchronen Abfragen ist die Bereitstellung eines Ergebnis-Listeners in der Produktstandardarchitektur, sowie eine eindeutige Identifikation jeder Suchanfrage, um die später eintreffenden Antworten zuordnen zu können[48].

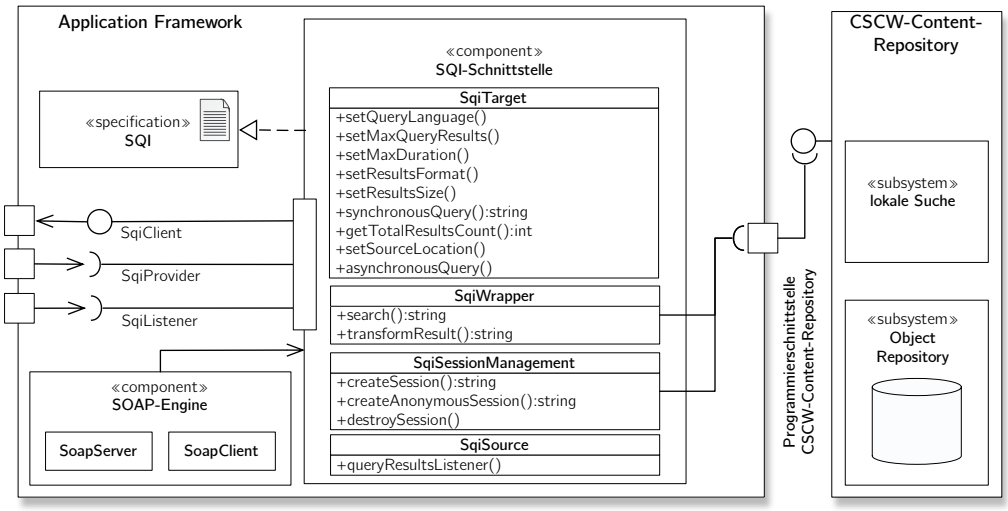

Abbildung 4.21.: Komponentenstruktur der SQI-Schnittstelle

Die Anwendungsfälle werden im Kontext des für die Produktarchitektur verwendeten Application Frameworks als Web-Service umgesetzt. Der Service muss drei Arten von Aufrufen unterscheiden: Das Stellen einer synchronen Anfrage mit direktem Rückgabewert, das Beauftragen einer asynchronen Anfrage und das Rückmelden des Ergebnisses. Die Rückmeldung wird dabei vom angefragten Repository ausgelöst und kann im Web Service dem auslösenden lokalen Objekt zugeordnet und an dieses weitergereicht werden.

Um für Anfragen über SQI zur Verfügung zu stehen, muss die Produktstandardarchitektur als SQI-Provider fungieren. Die dazu spezifizierten Methoden werden als Web-Service aus dem Kontext des Application-Frameworks heraus veröffentlicht.

[48] Eine asynchrone Beantwortung der Anfragen erfordert jedoch keine asynchrone Nachrichtenübermittlung auf der Infrastrukturebene.

Zur Implementierung des Sitzungsmanagements wird über die Programmierschnittstelle des Groupware-Frameworks auf dessen Authentifizierung zurückgegriffen. Mit der Erzeugung der Sitzung wird die Anmeldung des entfernten Benutzers vorgenommen und bei Erfolg eine Sitzungs-ID zurückgegeben. Die Zerstörung der Sitzung impliziert die Abmeldung vom System und das Trennen der Verbindung. Zur Sitzungsverwaltung kann in der Regel auf Funktionen des Application-Frameworks zurückgegriffen werden.

Die Methoden zur Konfiguration der Anfragen (*Query Parameter Configuration*) beeinflussen die lokale Suche im Object Repository. Zur Speicherung der Suchparameter kann die Session genutzt werden. Nach Eingang der Anfrage wird diese in einen Aufruf zur lokalen Suche transformiert. Das Ergebnis muss dann je nach gewünschten Ergebnisformat aufbereitet werden. Bei einer synchronen Anfrage wird es direkt zurückgegeben, bei einer asynchronen muss für die lokale Suche ein eigenständiger Thread gestartet werden, der diese analog zur synchronen Anfrage durchführt und das Ergebnis an die übermittelte Adresse sendet.

4.2.7. Erweiterte Benutzungsschnittstellen

Wie Kapitel 2 bereits deutlich gemacht hat, sind die Anforderungen an universitäre E-Learning-Umgebungen gestiegen. Benutzer fordern komfortable Benutzungsschnittstellen, sie möchten mit mobilen Endgeräten auf die aktuellsten Informationen zugreifen oder genau diese nach ihrem Login in ein hochschulweites Portal personalisiert und aggregiert einsehen wollen.

Der folgende Abschnitt zeigt daher Erweiterungsmöglichkeiten der Benutzungsschnittstellen auf Basis der Produktarchitektur auf.

4.2.7.1. AJAX/Rich Client Applications

Rich Client Applications wurden schon auf S. 55 skizziert. Gemeint ist damit die clientseitige Anreicherung von Benutzungsschnittstellen mit Javascript[49]. Hauptsächlich wird Javascript auf Programme angewendet, die innerhalb des Browsers ausgeführt werden. Ist der Quelltext geladen und vollständig interpretiert, läuft er innerhalb des Browsers in einem so genannten „Sandkasten“, von wo aus nicht auf Ressourcen außerhalb des Browsers (z. B. lokale Dateien) zugegriffen werden darf. In Javascript programmierte Funktionen und Objekte können also nur die mit der Webseite geladenen Inhalte verändern. Über ein spezielles Objekt (*XMLHttpRequest Object*) kann dann die eigentliche Kommunikation mit dem Server vom Benutzer entkoppelt und beispielsweise als Reaktion auf die Änderung eines Steuerelements durchgeführt werden. Dazu muss dem Objekt ein Event-Listener angehängt werden, der auf vom Server kommenden Statusänderungen einen Event-Handler ansteuert. Die Aktualisierung der Präsentation kann dann über das *Document Object Model* (DOM)[50] erfolgen.

[49] Eine objektbasierte Skriptsprache, die durch die European Computer Manufacturers Association international als *ECMAscript* spezifiziert wurde (ECMA-262) und für die alle modernen Browser eine Laufzeitumgebung bereitstellen.

[50] DOM ist eine vom W3C spezifizierte Programmierschnittstelle für den Zugriff auf HTML- und XML-Dokumente, die im Sinne der Objektorientierung die einzelnen Elemente als Klassen definiert. Zur Änderung von Attributen stehen Methoden bereit. So kann mittels Javascript sowohl Inhalt, Struktur

Der Server liefert die Daten – asynchron zu dem eigentlichen Laden der Webseite – an den Klienten aus, was insgesamt zu einem Benutzungsverhalten führt, das sich dem von Desktop-Anwendungen sehr ähnelt. Neben einem erhöhten Benutzerkomfort hat diese Technik auch den Vorteil, dass synchrone Komponenten wie bspw. ein Chat oder Awareness-Meldungen in die sonst asynchronen Web-Klienten eingebaut werden können. Dabei können AJAX-Frameworks wie „Prototype“ verwendet werden[51].

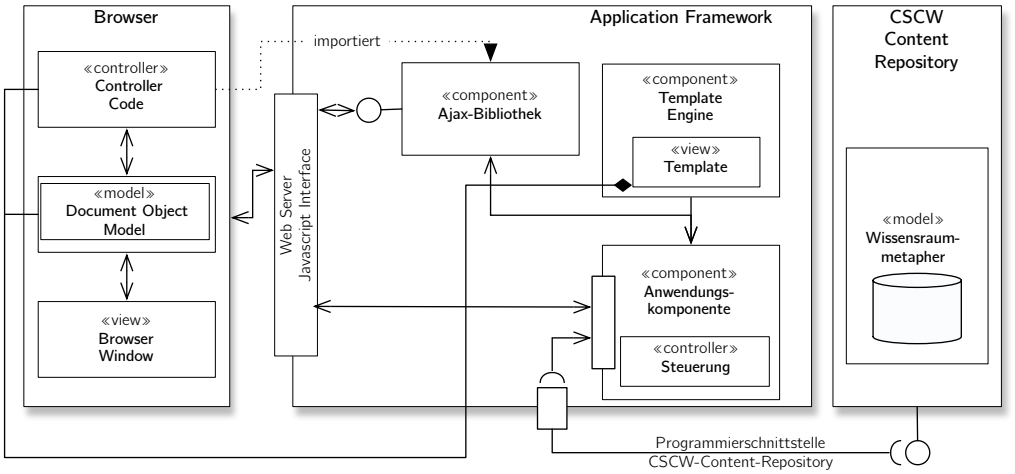

Abbildung 4.22.: Model-View-Controller-Konzept der AJAX-Schnittstelle

Von dem eigentlichen Anwendungsszenario abstrahiert soll die Rahmenarchitektur nun um diese Elemente erweitert werden. Dabei wird die Zweiteilung der Produktarchitektur in eine sowohl server- als auch clientseitige Behandlung der Anwendungs- und Präsentationsschicht durch das MVC-Entwurfsmuster verdeutlicht (vgl. Abb. 4.22). Seitens der Produktarchitektur wird das lokale Objektmodell durch eine Steuerungslogik kontrolliert, die in Anwendungskomponenten definiert ist. Zur Ausgabeerstellung kann eine *Template-Engine*[52] verwendet werden, wobei die zu konstruierende Sicht durch eine oder – geschachtelt – mehreren XHTML-Schablonen beschrieben wird. An dieser Stelle wird auch der Javascript-Code eingebettet, der die eingesetzte Ajax-Bibliothek referenziert. Das MVC-Pattern kann für die clientseitige Ausführung sowohl auf die gesamte Seite als Ganzes angewendet werden, jedoch auch separate Modelle und Controller für einzelne Ausgabeelemente implementieren. Der um den Javascript-Code angereicherte XHTML-Quelltext wird über den Web-Server an den Browser ausgeliefert und dort ausgeführt.

und Layout des übertragenen Dokumentes dynamisch verändert werden.

[51] Diese Bibliothek ist am weitesten verbreitet und im Internet unter `http://prototypejs.org` zu finden. Eine bekannte Effekt-Erweiterung zu Prototype ist „Scriptaculous“, zu finden unter `http://script.aculo.us`. Letzter Zugriff: 31.07.2008.

[52] So genannte Template Engines sind meistens fester Bestandteil von (Web-)Application Frameworks. Sie verarbeitet Text-Dateien (Templates) und ersetzt die dort definierten Platzhalter durch (Inhalts-)Daten.

4.2.7.2. Mobile Endgeräte

Grundsätzlich kann *Mobile Learning* immer dann zum Einsatz kommen, wenn kein Personal Computer zur Verfügung steht, d. h. wenn andere Formen wie WBTs nicht zur Verfügung stehen. Dies ist insbesondere in spontanen Situationen, zum Nachlesen oder in kurzen Lernphasen der Fall, aber auch um die Aktualität der Informationen sicherzustellen. (vgl. [KS05b], S. 9).
Mobile Lernszenarien können dabei entweder als eigenständige Anwendung umgesetzt werden, die auf dem Endgerät lauffähig sind (vgl. [Str07]). Diese Art der Anwendung kann mit der hier skizzierten Produktarchitektur jedoch nicht umgesetzt werden und wird an dieser Stelle daher nicht näher betrachtet. Eine andere Möglichkeit liegt in der Erweiterung der Benutzungsschnittstelle existierender web-basierter Anwendungen, die auf der Produktstandardarchitektur aufsetzen. Letzteres eignet sich insbesondere dazu, existierende Anwendungslogik und Informationen auf mobilen Endgeräten verfügbar zu machen.

Im Folgenden wird die Rahmenarchitektur um eine entsprechende Schnittstelle erweitert, die diese Funktion erfüllt und einen mobilen Zugriff auf Lernräume und Inhalte bietet. Dazu muss jedoch zunächst die erweiterte Infrastruktur eines solchen Szenarien analysiert werden, um daran Anforderungen abzuleiten.

Die in Deutschland angebotenen Übertragungsdienste GSM, GPRS und UMTS unterscheiden sich vor allem im Übertragungsverfahren und in der Geschwindigkeit. Der paketorientierte Datendienst GPRS ist eine abwärtskompatible Erweiterung des ursprünglich auf konstante Datenraten ausgelegte GSM-Netzes und erreicht eine reale Übertragungsgeschwindigkeit von ungefähr 50 kbit/s. Zum Datenaustausch wird das Internetprotokoll (IP) verwendet, weshalb jedes mobile Endgerät eine individuelle IP-Adresse erhält.

Der wirkliche Nachfolger des GSM-Netzes ist das *Universal Mobile Telecommunications Systems* (UMTS), das sich durch eine leistungsfähigere Funktechnik[53] auszeichnet und spezifizierte Datenübertragungsraten umsetzen kann, die von 144 kbit/s für den hochmobilen Nutzer[54] und bis zu 2 Mbit/s für den quasistationären Betrieb reichen.

Studien gehen jedoch davon aus, dass UMTS im Jahr 2010 erst eine Marktdurchdringung von 50 % erreicht hat. In Anbetracht der Multimodefähigkeit[55] der UMTS-Endgeräte sollten Benutzungsschnittstellen von mobilen Geräten kurz- bis mittelfristig auf die Übertragungstechnologie GPRS ausgerichtet sein. Das bedeutet für eine konkrete Implementierung, dass die Struktur der Anwendung für eine Transaktionsgeschwindigkeit von ungefähr 50 kbit/s konzipiert wird. Überflüssige Interaktionen und überladene Inhalte sollten daher vermieden werden.

Unabhängig von der Übertragungstechnologie wird das *Wireless Application Protocol* (WAP) benötigt, um Internetinhalte für die langsameren Übertragungsraten und für

[53] Im Einzelnen wird die höhere Leistungsfähigkeit durch eine größere Bandbreite und ein neueres Übertragungsverfahren (Wideband-CDMA) erreicht.
[54] Die maximale Geschwindigkeit der Empfangseinheit beträgt 500 km/h.
[55] Die Geräte können für Sprach- und Datenverbindungen auch das GSM-Netz nutzen.

die kleineren Displays verfügbar zu machen. Der Standard besteht aus einer Sammlung von Protokollen. In den Versionen WAP 1.0 bis 1.2 verläuft die Übertragung nach dem Client-Gateway-Server-Prinzip: Ein WAP-Gateway übersetzt die Anfragen des Klienten in HTTP und nimmt eine Verschlüsselungskonvertierung von WTLS nach SSL/HTTPS vor. Dem Webserver der Anwendung wird jedoch über den MIME-Type mitgeteilt, dass es sich um eine Anfrage in WML handelt. Die Rückantwort des Servers wird durch das WAP-Gateway zurückkonvertiert und kompiliert an den Klienten übertragen[56]. Ab der Version WAP 2.0, die auf einem geänderten Protokoll-Stack basiert, wird die Web-Anwendung direkt über HTTP/HTTPS angesprochen und der Umweg über einen WAP-Gateway eingespart[57].

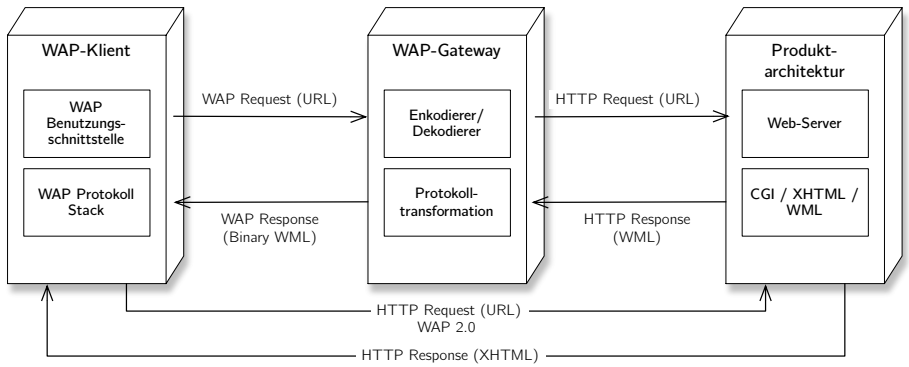

Abbildung 4.23.: Client-Gateway-Server-Prinzip der Schnittstelle für mobile Endgeräte

Auf der Serverseite unterscheidet sich WAP somit kaum von anderen Dokumententypen im WWW. Um eine Schnittstelle für mobile Endgeräte im Kontext der Rahmenarchitektur anzubieten, wird daher lediglich ein Webserver benötigt, der abhängig von der bevorzugten Protokoll-Variante WAP 1.x oder 2.0 entsprechende MIME-Types unterstützen muss.

Zur Beschreibung der Inhalte wird aktuell die *Extensible HyperText Markup-Language* (XHTML) verwendet[58]. Diese kann dynamisch zur Laufzeit der Anwendung analog zur normalen browserbasierten Ausgabe erzeugt werden. Somit kann im günstigsten Fall sogar auf eine eigene Anwendungsfallsteuerung (*Controller*) für den mobilen Zugriff verzichtet werden, wenn die Präsentation (*View*) in großen Teilen identisch bleiben soll. Die Auswahl eines geeigneten XML-Templates kann dann durch den Controller anhand

[56] Die Seiteninhalte werden in das kompaktere Format WBXML (*WAP binay XML*) umgewandelt, um die bestehende Bandbreite besser auszunutzen. Die Übertragung verläuft im MIME-Type WMLC (*Wireless Markup Language Compiled*).

[57] Allerdings werden die Inhalte dadurch auch nicht mehr für das Endgerät aufbereitet. Für den Endbenutzer bedeutet diese datenintensivere Internetnutzung auch einen Kostenanstieg. Mobilfunkbetreiber kommen dem aber mit entsprechenden Produkten wie Datenflatrates entgegen.

[58] Die Weiterentwicklung von HTML wurde nach der Version 4.01 zugunsten von XHTML eingestellt und somit auch das auf HTML beruhende und für den mobilen Zugriff optimierte *Compact HTML* (cHTML). Im Zuge der Spezifizierung von WAP 2.0 wurde ebenfalls auf die auf XML beruhende *Wireless Markup Language* (WML) zugunsten von XHTML verzichtet.

des HTTP-Headers `HTTP_ACCEPT` getroffen werden, der die vom Browser akzeptierten MIME-Types beinhaltet. Weitere Header-Informationen – sollte eine feinere Differenzierung nötig sein – sind `HTTP_USER_AGENT` und `X-WAP-PROFILE`.

Die Auswertung des Accept-Headers übernimmt in der Rahmenarchitektur die Klasse „AcceptDispatcher“. Sie ermittelt den MIME-Type, den der anfragende WAP-Klient verarbeiten kann. Mit Hilfe dieser Informationen bestimmt der „TemplateDispatcher“ die passende XML-Schablone, die durch die Anwendungsfallsteuerung mit dynamischen Daten gefüllt und über den Web-Server an das WAP-Gateway – bzw. im Fall von WAP 2.0 direkt an den Klienten – zurückgeschickt wird.

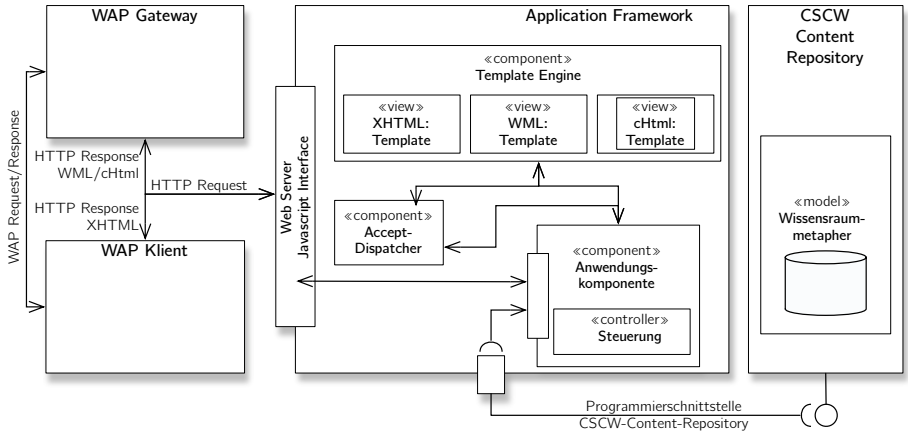

Abbildung 4.24.: Die Auswahl der Sicht erfolgt in der Anwendungssteuerung über einen Dispatcher, der den HTTP-Request analysiert und eine entsprechende Schablone auswählt.

4.2.7.3. Hochschulportale

Im Mittelpunkt einer weiteren Ausbaustufe kann ein gemeinsames Portal von Fachbereichen, Bibliothek, Verwaltung und IT-Diensten stehen, das eine einheitliche, systemübergreifende Benutzungsschnittstelle für determinierte Abläufe bietet (*Präsentationsintegration*, vgl. [A+07])[59]. „Ziel hinter dem Portal ist es, die in der Regel getrennten, inneren Systeme nach außen hin zu vereinheitlichen, so dass dem Benutzer ein einziger Zugang, das Portal, das gesamte Unternehmen, oder zumindest einen wohl definierten Ausschnitt daraus, präsentiert wird“ [Mas05], S. 53. Ein Hochschulportal ist dabei eine horizontale Anwendung, d. h. dass sich eine Portalintegration nicht vertikal an den Funktionalbereichen orientiert, sondern horizontal entlang der Prozesse. Hochschulportale integrieren dabei typischerweise Inhalte und Funktionen verschiedener, zumeist heterogener Applikationen der Universität.

[59]Zum Thema Hochschulportale siehe auch [DEH+02], S. 4, [Ker04b], [MJ05], S. 21, [AJ05], S. 31f.

Die Integration orientiert sich dabei an den verfügbaren Benutzungsschnittstellen der einzubindenden Anwendungen: *Client-Side Content Fetching* ist dabei eine Technik der URL-Integration, bei der im Portal nur die Quell-Adresse des Backend-Systems hinterlegt ist und der Browser die Daten von diesem direkt abholt. Beim *Server-Side Content Fetching* werden die Daten zunächst vom Portal aus dem Backend-System geladen und dann an den Browser weitergegeben[60]. Weiterhin kann eine Portalsoftware natürlich auf die existierende Infrastruktur von Web-Services und Nachrichten-orientierter Middleware zugreifen, um verteilte Dienste zu integrieren oder Transaktionen einzubinden.

Eine Präsentationsintegration verschiebt somit per se keine Anwendungsgrenzen und muss daher nicht zwangsweise zu einer Kopplung führen. Gerade aus diesem Grund eignet sie sich für Anwendungsfälle, in denen mit mehreren Anwendungen parallel gearbeitet werden muss, eine Integration auf Daten- oder Funktionsebene aber nicht möglich oder nicht gewollt ist, bspw. aus sicherheitstechnischen oder datenschutzrechtlichen Gründen (vgl. [OWZ+05], S. 47).

Kirchhoff et al. unterteilen in [KGHV04] die Portalsoftware in zwei Bestandteile – den Portal-Basisdiensten und den Portal-Anwendungen, den so genannten Portlets:

Portal-Basisdienste bündeln eine Reihe übergreifender Dienste wie Layout-Management, Struktur- und Content-Management als auch Infrastrukturdienste zur Authentifizierung, Rechte- und Nutzerverwaltung, Suche etc. Diese Dienste sind fest im Portal verankert und in ihrer Ausprägung sehr herstellerspezifisch. Sie sollen daher in diesem Kontext nicht weiter betrachtet werden. Eine allgemeine Beschreibung findet sich in [KGHV04].

Portlets ermöglichen die Aggregation von Content sowie die Ausführung von Funktionalität verschiedener Anwendungen in der Präsentationsschicht. „Damit integrieren Portlets geschäftsprozessspezifische Funktionalität und bieten dem Anwender einen homogenen Zugang zu unterschiedlichen Datenquellen und Applikationen“ [Pus04], S. 118. Sie realisieren also die betreiberspezifischen Anwendungsfunktionalitäten für das Portal und sollen daher in die Rahmenarchitektur mit aufgenommen werden.

Portlets werden aus einer allgemeinen Portal-Anwendungsklasse abgeleitet. Im Sinne des objektorientierten Gedanken deklariert eine abstrakte Portal-Oberklasse eine Reihe von Methoden und Eigenschaften, die von den Anwendungsklassen geerbt und zum Teil definiert werden müssen.

Es existieren verschiedene Formen von Anwendungsklassen am Markt, wie z. B. Portlets, Webparts, Remote Portlets, I-Views, I-Lets oder Gadgets, die zum Teil herstellerabhängig und nicht kompatibel zueinander sind[61]. Ein Portlet muss daher eine be-

[60] Bei dieser Variante kann häufig ein Cache verwendet werden, so dass auch bei Nicht-Verfügbarkeit des Backends Inhalte ausgeliefert werden können. Ohne den Einsatz von Cache-Mechanismen kann Server-Side Content Fetching den Portalserver erheblich belasten, da in der Regel personalisierte Sichten abgerufen werden müssen.

[61] Im Java-Umfeld sind Portlets nach dem JSR168-Standard weit verbreitet. Die in 2008 von Sun neu definierte Spezifikation JSR286 (*Portlet Specification 2.0*) erlaubt als Weiterentwicklung von JSR168

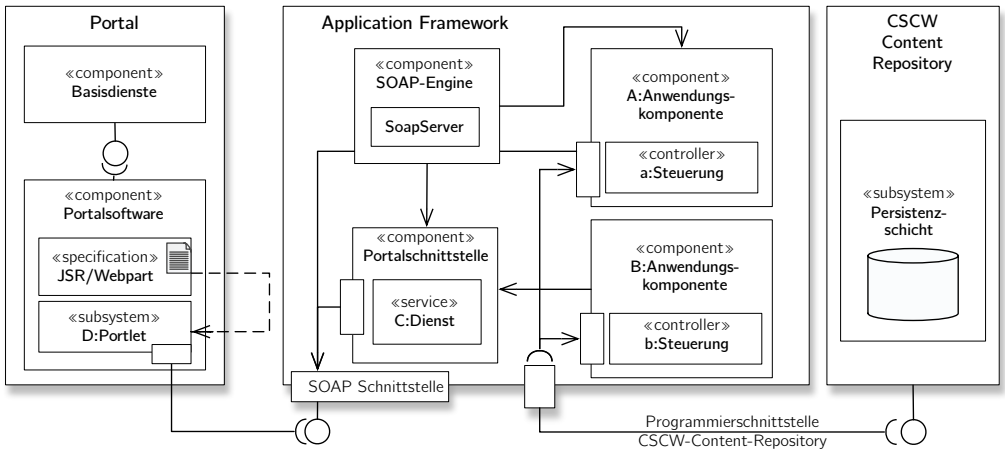

Abbildung 4.25.: Zugriff eines Portlets über die SOAP-Schnittstelle der Produktarchitektur

stimmte Portal-API implementieren, um z. B. auf die Portal-Basisdienste wie Authentifizierung, Caching, Content-Management etc. zugreifen zu können.

Die zu präsentierenden Daten aus der Anwendung werden vom Portlet über eine SOAP-Schnittstelle aus der Produktarchitektur ausgelesen. Der von der Produktarchitektur dazu zur Verfügung zu stellende Dienst kann – abhängig von den benötigten Daten – von einer Anwendungskomponente (hier: A) direkt implementiert werden oder jedoch durch eine eigenständige Portalschnittstelle (hier: C), die auf einer oder mehreren Anwendungskomponenten (hier: B) basiert. Somit können sowohl einfache Basisdienste der Architektur wie bspw. die bibliographische Katalogsuche über die Z39.50-Schnittstelle als Portlet verfügbar gemacht werden als auch gekoppelte Komplexdienste aus der Anwendungsschicht der Produkte.
Für die Darstellung ist in der Regel das Portlet selbst zuständig. Die Vielfalt der technischen Lösungen reicht dabei von einzelnen JSP-basierten Portlets bis hin zu frameworkbasierten Lösungen auf Basis von Struts[62] und *JavaServer Faces* (JSF) im Java-Umfeld, bzw. ASP.NET und den Sharepoint Services im Windows Umfeld.

4.3. Proaktive Wiederverwendung von Komponenten

Der statische Teil der Rahmenarchitektur wurde durch die Definition logischer Komponenten und der Beschreibung ihres Zusammenspiels im letzten Unterkapitel behandelt. Es wurde ein Rahmenwerk konzipiert und die dazugehörigen Komponenten nach

durch die Inter-Portlet-Kommunikation eine auf einen Kontext besser abgestimmte Anwendungsoberfläche. In der Microsoft-Welt (Sharepoint Services) werden standardmäßig Webparts eingesetzt.

[62] Struts ist ein Open-Source-Framework, das sowohl für die Anwendungssteuerung als auch für die Präsentationsschicht von Java-Webanwendungen verwendet werden kann. Informationen unter: `http://struts.apache.org`. Letzter Zugriff: 31.07.2008.

Grad der Wiederverwendung in Produktstandard- und Produktarchitektur aufgeteilt. Dadurch wurde der Rahmen für eine einheitliche Plattform spezifiziert, die für verschiedenste Anwendungen in der Domäne des universitären E-Learnings eine geeignete Basis darstellt.

Nach Dern beschreibt der komplementäre dynamische Teil einer Anwendungsarchitektur, wie die einzelnen Komponenten auf Grundlage der statischen Sicht beschrieben, erstellt und eingeführt werden (vgl. [Der03], S. 18f.). Damit die Wiederverwendungspotenziale der Produktstandardarchitektur zugunsten einer kostengünstigeren und weniger fehlerbehafteten Anwendungsentwicklung realisiert werden können, sollte das Vorgehen auch hier die Wiederverwendung von Komponenten gezielt unterstützen. Dabei hat sich in der Praxis ein Produktlinien-orientiertes Vorgehen bewährt, das in den folgenden Abschnitten beschrieben wird.

4.3.1. Softwareproduktlinien

„A software product line is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.“ Diese Definition des Carnegie Mellon Software Engineering Institutes (SEI) impliziert bereits das Ideal einer proaktiven Wiederverwendung bei der Entwicklung einer Softwareproduktlinie (SPL). Nämlich die zielgerichtete Entwicklung mehrfach verwendbarer Komponenten für einen bestimmten Anwendungsbereich unter vorhersehbaren technischen Rahmenbedingungen (vgl. hierzu auch [CE00], S. 21, [MA07], S. 180).

Dieses vorausschauende Vorgehen geht dabei über die reine Wiederverwendung von Komponenten aus früheren Entwicklungen oder Entwicklung abstrakter generischer Komponenten hinaus[63]. Dazu ist der Entwicklungsprozess ganzheitlich ausgerichtet und umfasst sowohl technische als auch methodische, organisatorische und strategische Aspekte[64]. In Analogie zur softwaretechnischen Unterscheidung von Produktstandard- und Produktarchitektur werden methodisch konzeptionell auch zwei Lebenszyklen unterschieden, nämlich die Plattformentwicklung (*Domain Engineering*) für den Bau wiederverwendbarer Artefakte, und die Produktentwicklung (*Application Engineering*) für die Erstellung darauf basierender Anwendungen (vgl. [CE00], S. 19ff., [MA07], S. 181f., [Sch08], S. 111f.).

Im Folgenden werden diese beiden Phasen näher erläutert. Die Untergliederung lehnt sich an [CE00] an. Eine detaillierte Beschreibung der Tätigkeiten findet sich im Vorgehensmodell, dass im folgenden Kapitel 5 konzipiert wird und im Anhang S. 259ff. beschrieben ist.

[63] Die reaktive Wiederverwendung wird auch als Baukasten-orientierte Softwareentwicklung bezeichnet. Der Schwerpunkt liegt dabei bei der Anwendungsentwicklung (vgl. [MA07]).

[64] Im Detail sind dies (1) die strategische Ausrichtung der Produktlinie an die Ziele der Organisation, (2) das systematische Managen von Varianten respektive Systemunterschieden bei der Komponentenentwicklung, (3) die Implementierung einer Produktstandardarchitektur als gemeinsame Basis und (4) die konzeptionelle Trennung zwischen der Entwicklung wiederverwendbarer Komponenten und der von Produkten (vgl. [Sch08], S. 110f.).

4.3.2. Separate Plattform- und Produktentwicklung

Der grundsätzliche Aufbau bei beiden Phasen ist identisch: Analyse, Entwurf, Implementierung und Test, mit jeweils entsprechenden Entwicklungs-Ergebnissen für Plattform oder Produkt. In der Plattformentwicklung liegt der Fokus auf der Umsetzung der Standardarchitektur für die Produktfamilie mitsamt ihren wiederverwendbaren Artefakten[65]. Die Produktentwicklung umfasst die Erstellung konkreter Applikationen aus der Plattform mit dem Ziel, einen hohen Wiederverwendungsgrad zu erreichen. Wie in Abbildung 4.26 zu sehen ist, sind beide Prozesse miteinander eng verzahnt. Jede Produktentwicklung bedeutet einen neuen Zyklus zur Weiterentwicklung der Plattform. Die Zwischenergebnisse der Plattformentwicklung wie Spezifikation, Entwurf und Implementierung der Standardkomponenten fließen in den jeweils korrespondierenden Abschnitt der Produktentwicklung mit ein. Dieser ist durch eine Feedbackschleife wieder rückgekoppelt, was eine iterativ-evolutionäre Prozesssteuerung begünstigt.

Die konzeptionelle Ausrichtung der beiden Phasen ist jedoch unterschiedlich. Während bei der Plattformentwicklung die Struktur und die Auswahlregeln für die Komponenten der Plattform konzipiert und verwaltet werden müssen, um die Variabilität innerhalb des Produktraums zu beherrschen, sind bei der Produktentwicklung die notwendigen Standardkomponenten zu konfigurieren und gegebenenfalls mit zusätzlichen Anwendungskomponenten zu erweitern.
Der Übergang von einer Einzelsystemsicht hin zu einer integrierten Entwicklung einer Systemfamilie rückt demnach die Gemeinsamkeiten und Unterschiede in den Produktvarianten in den Mittelpunkt, was das so genannte Variantenmanagement notwendig macht (vgl. [Sch08], S. 112, [Beu05]). Entscheidend ist dabei der Überblick darüber,

- welche Anforderungen für alle Systeme gelten (*Gemeinsamkeiten*),
- welche Anforderungen nur für einen Teil der zu entwickelnden Systeme gelten, aber nicht für alle relevant sind (*Variabilitäten*) und
- welche Anforderungen nur für bestimmte Produkte relevant sind (*produktspezfische Anforderungen*).

Ziel der Domänenanalyse (*Scoping*) als Teil der Plattformentwicklung ist es, diesen Überblick herzustellen. Dabei wird der Produktraum von einer übergeordneten Strategie – in diesem Fall bspw. die hochschulweite E-Learning- und IT-Strategie – abgeleitet und im Sinne eines Portfolios präzisiert. Aus der Strategie können wesentliche Anforderungen an die Produktlinie abgeleitet und im Rahmen von weiteren Analysen ergänzt werden. Diese Anforderungen können in Form textueller Beschreibungen verfasst werden. Hierzu bietet sich die in Kapitel 4 vorgestellte normsprachliche Modellierung an. Die Ergebnisse der Domänen-Analyse sind eine Produktraumbeschreibung, ein Domänenlexikon und Konzeptmodelle.

[65] „Domain Engineering is the activity of collecting, organizing and storing past experiences in building systems or parts of systems in a particular domain in the form of reusable assets (i.e., reusable work products), as well as providing an adequate means for reusing these assets (i.e., retrieval, qualification, dissemination, adoption, assembly, and so on) when building new systems“ [CE00], S. 20.

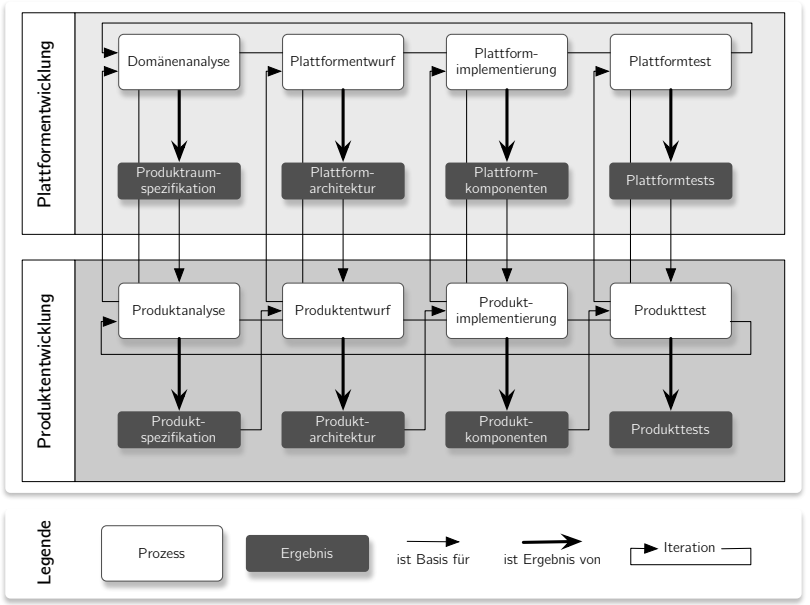

Abbildung 4.26.: Zwei separate Iterationszyklen: Die Entwicklung der Produktstandardplattform und die Produktentwicklung.

Wechselseitig mit der Domänenanalyse wird in der Produktanalyse das traditionelle *Requirements Engineering*[66] weiter ausgedehnt, um die Variabilitäten zu analysieren und zu dokumentieren[67]. Das Ergebnis dieser beiden Prozesse in der ersten Iteration sind so genannte Featuremodelle zur Modellierung von Variationspunkten und Varianten (vgl. Abb. 4.27), die immer weiter verfeinert werden können. Sie bilden die Entscheidungsgrundlage bei der Fragestellung, welche Komponenten der Plattform wiederverwendet werden können, welche angepasst, neu hinzugefügt oder individuell für ein Produkt neu entwickelt werden müssen. Darüber hinaus unterstützen Featuremodelle in folgenden Iterationen eine werkzeuggestützte Anforderungserhebung und Komponentenauswahl (vgl. [CE00], S. 38ff., [Beu05], S. 75ff.).

Die Entscheidungen, welche Requirements-Artefakte SPL-spezifisch und welche produktspezifisch sind, beeinflussen in einem nächsten Schritt „Entwurf“ die Produktarchitektur und das produktübergreifende Architekturdesign der Plattform. Hier werden die Anforderungen mit technischen Lösungen verknüpft. Bei diesem Schritt kann die in diesem Kapitel konzipierte Rahmenarchitektur als Orientierung für ein Architekturdesign verwendet werden.

[66] Also das Finden und Verstehen von Anforderungen, die Anforderungsanalyse und -dokumentation, sowie die Verifizierung und das Management derselben.

[67] Weiss und Lai schlagen dazu [WL99] weitere Analysen vor: (1) die *Commonality Analysis* zur Analyse der gemeinsamen Anforderungen aller Systeme der Produktlinie, (2) die *Variability Analysis* zur Analyse von Anforderungen, die sich in den Systemen der Produktlinie unterscheiden und (3) das *Variability Modelling* zur Modellierung von Variationspunkten, möglichen Varianten und ihren Beziehungen.

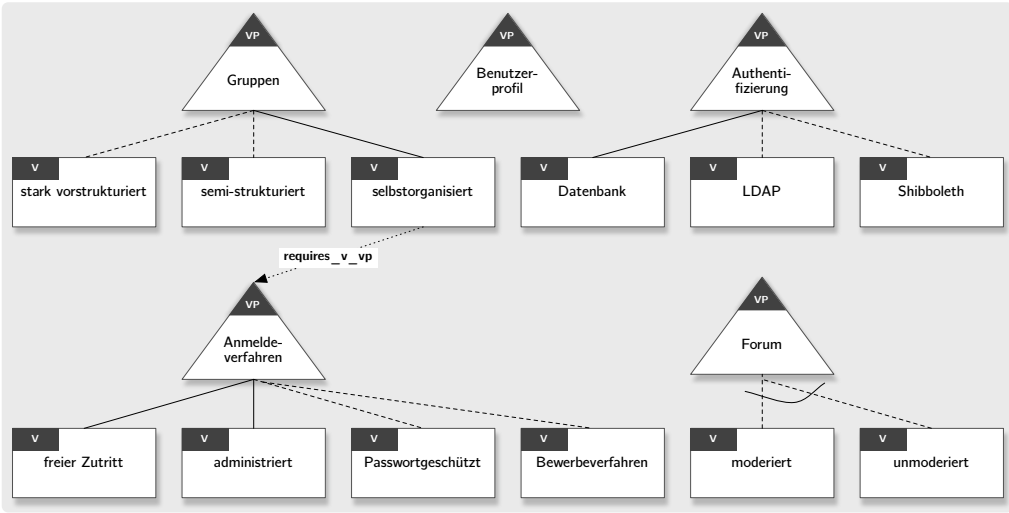

Abbildung 4.27.: Featuremodell mit notwendigen, optionalen, alternativen und exklusiven Varianten (Ausschnitt).

Der Schritt „Implementierung“ bedeutet für die Plattformentwicklung die Umsetzung oder Erweiterung von Standardkomponenten, die gemeinsame oder variable Features realisieren. Im Gegensatz dazu besteht ein großer Teil der Produktimplementierung darin, einen Verbund von Standardkomponenten zusammenzustellen und diese anwendungsspezifische Konfiguration um individuelle Komponenten zu erweitern. Das Ergebnis beider Phasen ist eine funktionierende, erweiterte Produktplattform mit anwendungsspezifischen Komponenten, die gemeinsam das neue Produkt zur Realisierung der erhobenen Anforderungen implementieren.

In einem letzten Schritt wird zum einen das Zusammenspiel neuer Standardkomponenten mit den bisherigen Standardkomponenten der Produktplattform getestet („Plattformtest“), zum anderen die Integration der anwendungsspezifischen Komponenten mit der Produktplattform („Produkttest“).

4.3.3. Anwendbarkeit des Produktlinien-orientierten Vorgehens

In der Praxis hat sich gezeigt, dass im universitätsweiten Kontext eine genaue Vorab-Spezifizierung des Produktraums ohne eine zentrale Koordination nur schwer möglich ist[68]. Auch wenn eine E-Learning-Strategie vorhanden ist, umfasst sie aus softwaretechnischer Sicht oftmals nur die hochschulweite Einführung und den Betrieb einer

[68] Diese Erkenntnis spiegelt die Erfahrung der Autors dieser Thesis aus der mehrjährigen Mitarbeit in universitätsinternen und -übergreifenden Projekten und Arbeitskreisen zur Infrastrukturentwicklung an Hochschulen wider.

oder mehrerer (monolithischer) Lernplattformen[69]. Komponentenbasierte Individualentwicklungen im Kontext einer dienstorientierten Infrastruktur finden zwar an amerikanischen Universitäten mit der *Open Knowledge Initiative*[70] immer mehr Verbreitung. Im deutschsprachigen Raum hat sich dieses Vorgehen jedoch noch nicht durchgesetzt[71]. Allerdings sind in den letzten Jahren an einigen Universitäten zentrale Dienstleistungszentren im Bereich „Neue Medien“ eingerichtet worden, die neben der Beratung, Schulung und Betrieb von Lernplattformen auch Software-Anpassung und -Entwicklung anbieten. Weiterhin existieren an vielen Universitäten bereits Foren und Arbeitskreise zum Thema E-Learning. Oftmals engagieren sich auch dort Dozenten, die sich schon Ende der 90er Jahre im Rahmen der BMBF-geförderten Projekte mit der Anwendungsentwicklung im Bereich der computerunterstützten Lehre auseinander gesetzt haben. Auf diesen Erfahrungen aufbauend kann – koordiniert durch eine zentrale Stelle wie ein Dienstleistungszentrum, ein Arbeitskreis oder ein initiales Pilotprojekt – eine Produktraumspezifikation vorgenommen und mit der E-Learning-Strategie der Universität abgestimmt werden.

Aufgrund der hohen Initialkosten, die bereits vor Fertigstellung der ersten Anwendung anfallen und somit schwer auf alle zu entwickelnden Produkte aufzuteilen sind, bedeutet das aus der Softwareindustrie stammende Produktlinien-orientierte Vorgehen im universitären Kontext sicherlich eine Hürde. Schon bei der Entwicklung des ersten Produktes ist eine stabile und wiederverwendbare Produktstandardarchitektur zu realisieren.

Dem muss man jedoch entgegenhalten, dass gerade an einer Universität wie nirgendwo anders die Möglichkeit besteht, wiederverwendbare Komponenten der Standardarchitektur im Rahmen der Lehre und Forschung proaktiv und kostengünstig zu entwerfen, zu implementieren und zu testen. Das dazu notwendige Softwaredesign kann sich an der in dieser Arbeit spezifizierten Rahmenarchitektur orientieren. Der Erfahrung nach eignen sich zur Umsetzung der Komponenten insbesondere Projektseminare und Abschlussarbeiten (Bachelor-, Master- u. Diplomarbeiten). Notwendig ist dazu jedoch ein standardisiertes Vorgehen, wie es im nächsten Kapitel mit der Instanzierung eines Referenzmodells für Qualitätsmanagement und Qualitätssicherung bei der Entwicklung von Bildungsangeboten erarbeitet wird.

[69] Strategie „Verwendung von Standardsoftware“, siehe dazu S. 114. Weitere mögliche Inhalte einer E-Learning-Strategie sind in Kapitel 2 als organisatorische und technische Maßnahmen aufgeführt (S. 25ff.).

[70] Informationen unter `http://www.okiproject.org/`, siehe auch S. 46. Letzter Zugriff: 31.07.2008.

[71] Es gibt jedoch Vorreiter, die früh auf eine service-orientierte Architektur gesetzt haben und auf dieser Basis auch Eigenentwicklungen umsetzen, bspw. das *Karlsruher Integrierte InformationsManagement* (KIM, `http://www.kim.uni-karlsruhe.de/`). Viele Universitäten befinden sich derzeit aber in Abwartehaltung.

5. Entwicklungsmethodik komponentenbasierter Lernumgebungen

In diesem Kapitel wird als drittes Gestaltungsziel dieser Arbeit eine Methode entwickelt, die als Grundlage für die normsprachliche Spezifizierung (Kapitel 4) und komponentenorientierte Umsetzung (Kapitel 5) universitärer Lern- und Arbeitsumgebungen herangezogen werden kann. Zunächst werden die Konzepte von Methoden und Vorgehensmodellen erläutert und voneinander abgegrenzt. Auf dieser Basis wird eine Entscheidungshilfe zur Auswahl einer Prozesssteuerung für die Softwareentwicklung von E-Learning-Werkzeugen im universitären Kontext erarbeitet (6.1). Anschließend werden verschiedene Referenzmodelle zur Entwicklung von E-Learning-Komponenten vorgestellt, um eine für den Kontext dieser Arbeit passende Prozessarchitektur auszuwählen (6.2). Auf Basis der Prozessarchitektur des Referenzmodells für Qualitätsmanagement und -sicherung bei der Planung, Entwicklung und Durchführung von Bildungsangeboten (DIN PAS 1032-1:2004) wird eine Anpassung an das im letzten Abschnitt vorgestellte Produktlinien-orientierte Vorgehen vorgenommen. Die Anwendung des Referenzmodells auf die spezifischen Aspekte der Entwicklung von Lern- und Arbeitsumgebungen im universitären Kontext basiert dabei auf den eigenen mehrjährigen Erfahrungen des Autors aus Verbundprojekten und Arbeitsgruppen im Bereich Neuer Medien in der Bildung, insbesondere E-Learning, sowie aus Projekten zur IT-Infrastrukturentwicklung an Hochschulen.

5.1. Grundlagen der methodischen Entwicklung

5.1.1. Begriffsdefinition einer Methode

Die Entwicklung von Informationssystemen (und ihren Bestandteilen) bedingt ein ingenieurmäßiges Vorgehen, damit es planbar und wiederholbar ist. Bereits seit den 80er Jahren befasst sich die Wissenschaft daher mit dem Methoden-Engineering als rechnergestützten, methodischen Entwurf sowie zur rechnergestützten, kontrollierten Ausführung von Methoden. Mittlerweile hat es einen hohen Stellenwert in der angewandten Informatikforschung (vgl. [Gut94], S. 11).

Das Konzept der Methode bzw. das Verständnis, welches hinter dem Begriff der Methode in der Forschung zu Informationssystemen und ihrer Entwicklung steht, unterliegt in der Literatur unterschiedlichen Auffassungen und Definitionen. Der klassischen Definition aus dem Griechischen folgend, ist eine Methode „der Weg zu etwas“. In den

verschiedenen Definitionen von Nonnemacher, Haberfellner[1] und Lorenz[2] steht insbesondere die Planmäßigkeit, die einen systematischen Ansatz erkennen lässt, sowie der Zweckbezug im Vordergrund. Stahlknecht und Hasenkamp propagieren darüber hinaus in ihrer Definition die Verwendung von Prinzipien[3].

Braun et al. haben unterschiedliche Methodendefinitionen daraufhin untersucht, inwieweit verschiedene Merkmale zu erkennen sind. Sie sind dabei zu dem Schluss gekommen, dass in der Literatur insbesondere die folgenden Merkmale im Vordergrund stehen (vgl. [BWHW05], S. 1296f.):

Zielorientierung Methoden sind zielorientiert. Sie vereinbaren Regeln, wie vorzugehen ist, um definierte Ziele zu erreichen und/oder Probleme zu lösen.

Verfolgung eines systematischen Ansatzes Wenn Methoden Regeln und Instruktionen vorgeben, wie Ziele erreicht und Probleme gelöst werden können, dann müssen sie auch eine systematische Struktur besitzen. Dies ist nötig, um spezifische Arbeitsschritte und Aufgaben zur Zielerreichung abzuleiten[4].

Einsatz von Prinzipien Methoden sowie Methodenspezifikationen können eng mit Design-Prinzipien, wie z. B. generellen Konstruktionsrichtlinien und/oder -strategien verbunden sein. Ein Beispiel ist das Prinzip der Objektorientierung. Ein Prinzip ist dabei allgemein gültig, abstrakt und bildet die theoretische Grundlage, die einer Handlung zugrunde gelegt werden kann (vgl. [Bal01], S. 36).

Wiederholbarkeit Methoden, insbesondere in der Software-Entwicklung, sollten nicht nur einmalig in einem bestimmten Kontext nutzbar sein und werden deshalb relativ offen gestaltet. Einige Autoren motivieren in ihren Definitionen zusätzlich, dass Methoden kontextübergreifend wiederholbar sein sollten[5].

In diesem Kapitel werden die Bestandteile einer Methode nach dem Methoden-Engineering dargestellt. Grundlage dafür bilden die Ausführungen von Gutzwiller (vgl. [Gut94]), nach dem eine Methode durch die Konzepte Entwurfsaktivität, Rolle, Technik, Entwurfsergebnis und Metamodell beschrieben wird. Diese konzeptionelle Methodenbeschreibung dient als Darstellungsrahmen. Kernpunkt der weiteren Ausführungen wird als Erweiterung der Methodenkonzepte die Vorgehensmodellierung sein, dessen Elemente im nächsten Abschnitt erklärt werden.

[1] Nonnemacher gibt in Anlehung an Haberfellner (vgl. hierzu [Hab80], S. 1701) eine relativ knappe Definition des Methodenbegriffs, die vielmehr einer kurzen Erklärung gleicht. Dort heißt es, dass „eine Methode [...] allgemein eine in der Art des Vorgehens festgelegte Arbeitsweise“ beschreibt (vgl. [Non94]).

[2] Nach Lorenz ist eine Methode, „ein nach Mittel und Zweck planmäßiges (=methodisches) Verfahren“ [Lor95], S. 876.

[3] „Vorschriften, wie planmäßig nach einem bestimmten Prinzip (oder einer Kombination von Prinzipien) zur Erreichung festgelegter Ziele vorzugehen ist“ (vgl. [SH99], S. 234).

[4] Dies ist insbesondere vielen Methoden zur Entwicklung von Informationssystemen zu Eigen, die auf ein ingenieurmäßiges Vorgehen setzen, um eine gewisse Plan- und Wiederholbarkeit zu gewährleisten (vgl. [Gre03], S. 955).

[5] Balzert spricht in [Bal01] in diesem Zusammenhang von der Anwendungsneutralität von methodischen Vorgehensweisen.

5.1.2. Bestandteile eines methodischen Vorgehens

Grob und Seufert beschreiben in [GS96] ein Vorgehensmodell als ein „[...] ablauforganisatorisches Konzept der Software-Entwicklung [...], bei dem ein komplexer Prozess in klar definierte, überschaubare Einheiten gegliedert wird. Durch Vorgehensmodelle wird Handlungswissen für die Erstellung von Software in Form von Prinzipien, Methoden und Werkzeugen zur Verfügung gestellt. Aus diesem allgemein gültigen Wissen soll ein Softwareentwicklungsprozess für konkrete Anwendungen abgeleitet werden“.

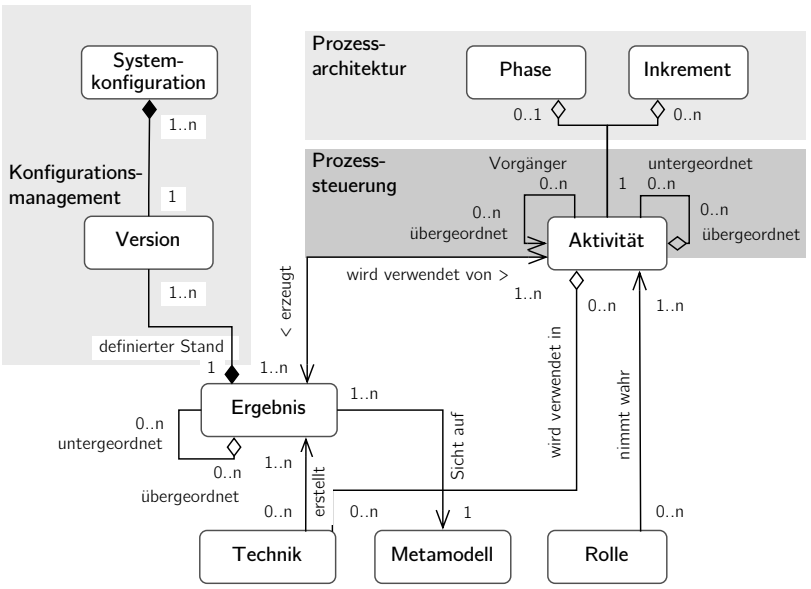

Abbildung 5.1.: Konzepte eines methodischen Vorgehens

Vorgehensmodelle beschreiben primär die Prozessarchitektur und -steuerung, beantworten also die Fragen, wann eine Aktivität durchgeführt und was für ein Entwurfsergebnis erstellt wird. Darüber hinaus können sie Konzepte des Konfigurationsmanagements implizieren, die im Wesentlichen auf der Versionierung von Ergebnissen und deren Zuordnung zu einer Systemkonfiguration basieren. Demnach beschreiben Vorgehensmodelle das Vorgehen im Großen, bieten jedoch oftmals keine konkreten Handlungsanweisungen für die konkrete Umsetzung. Die Betrachtungsweise einer Methode geht über das Vorgehensmodell hinaus und erweitert dieses um die fehlenden Aspekte. Nach dem Methoden-Engineering ist eine Methode damit die Erweiterung eines Vorgehensmodells um Rollen, Techniken und dem Metamodell. Eine Methode dient während eines gesamten Projektes als roter Faden und gibt zusätzlich konkrete Handlungsmuster vor. Dadurch wird es einfacher, komplexen Anforderungen zu genügen und diese in einem standardisierten Prozess zu erfüllen.

Da diese Elemente von Vorgehensmodellen und Methoden grundlegend sind für die in diesem Kapitel durchzuführende Methodenkonzeption, sollen sie an dieser Stelle kurz erläutert werden:

Prozessarchitektur Eine Prozessarchitektur gibt einen groben Überblick über die Phasen des Entwicklungsprozesses sowie der zugehörigen Aktivitäten (vgl. [NS99], S. 171).

Phase Eine Phase ist eine Kategorisierung des Fortschrittszustands einer Entwicklung anhand der Erreichung von Meilensteinen, welche die Phasen voneinander abgrenzen. Abhängig davon sind einer Phase die zur Erreichung des Meilensteins notwendigen Aktivitäten und Ergebnisse zugeordnet.

Inkrement In risikogetriebenen Vorgehensmodellen wird der Entwicklungsprozess als iterativ aufgefasst. Dabei ist eine zyklische Wiederholung der einzelnen Phasen vorgesehen, die jeweils eine Menge an Anforderungen realisiert. Jeder Zyklus beschreibt dabei ein Inkrement, für das die Risiken separat abgeschätzt, reduziert und evaluiert werden können.

Prozesssteuerung Das Prinzip, das dem Vorgehensmodell zur Definition der Ablaufreihenfolge zugrunde liegt (vgl. [NS99], S. 175, sowie [Dow87]). Dieses kann je nach Zielsetzung des Softwareentwicklungsprozesses variieren und klassifiziert die Zugehörigkeit eines Vorgehensmodells zu einer der im nächsten Abschnitt vorgestellten Familien von Vorgehensmodellen.

Aktivität Unter einer Aktivität wird eine Verrichtungseinheit verstanden, die ein oder mehrere definierte Ergebnisse erzeugt. Eine Aktivität kann in Subaktivitäten bis hin zu elementaren, nicht weiter teilbaren Arbeitsschritten[6] zerlegt werden. „In diesem Zusammenhang wird von einer Aktivitäten-Struktur, -Hierarchie oder -Dekomposition gesprochen“ [Gut94], S. 13. Diese Entwurfsaktivitäten sind aggregierte Aktivitäten und werden danach klassifiziert, ob sie iterativ oder nicht iterativ durchführbar sind. Manche Aktivitäten sind beliebig oft wiederholbar. Hierzu zählen z. B. Testprozesse oder Verifikationen von Funktionalität. Elementare Aktivitäten lasen sich nicht weiter unterteilen und liefern bei der Ausführung immer das gleiche Ergebnis.

Rollen Rollen repräsentieren in diesem Kontext die Aufgabe zur Ausführung bestimmter Aktivitäten, die durch Einzelpersonen oder Gruppen übernommen werden. Eine Rolle ist also – bezogen auf den Aufgabenträger bzw. Rolleninhaber – als Zusammenfassung spezifischer Aktivitäten zu sehen. Bsp.: Nutzer, Architekt, Programmierer, Support (vgl. [KLT98], S. 21).

[6] In der Literatur oft verwendete Begriffe für elementare Arbeitsschritte sind auch *task* oder *workstep*.

Techniken und Werkzeuge Techniken stellen Anleitungen oder Handlungsanweisungen bereit, wie Einzelergebnisse und/oder logisch zusammenhängende Gruppenergebnisse innerhalb von Aktivitäten erzielt werden können. Unter einem Werkzeug ist eine Software zu verstehen, die zur Planung und Generierung des Informationssystems verwendet wird (vgl. [FBML98], S. 17ff.). Eine Einteilung kann dabei in Werkzeuge für das Objekt- und Prozessmanagement sowie in Werkzeuge für die Softwareentwicklung erfolgen (vgl. [CG98], S. 219).

Ergebnisse Ergebnisse[7] werden durch die Aktivitäten entweder erzeugt oder verwendet/modifiziert und bilden somit eine „Input-Output"-Beziehung zwischen den verschiedenen Aktivitäten. Ein Entwurfsergebnis ist hierarchisch strukturiert und kann wiederum in seine Bestandteile zerlegt werden. Eine Beschreibungsstruktur legt fest, wie ein Ergebnis dokumentiert wird. Logisch zusammengehörende Ergebnisse können aggregiert und unter einem Bezeichner angesprochen werden. Ein Ergebnis kann z. B. ein Netzplan der Systemarchitektur oder ein Regelwerk einer Firewall sein. Die Gesamtheit aller Ergebnisse eines Vorgehensmodells kann auch als Dokumentationsmodell verstanden werden (vgl. [Gut94], S. 14).

Version Eine Version ist ein ursprüngliches Ergebnis oder eine Veränderung respektive Weiterentwicklung eines Ergebnisses. Die Anzahl erforderlicher Versionen eines Ergebnisses in einem Softwareentwicklungsprozess erhöht sich beispielsweise dann, wenn mehrere Entwickler ein Ergebnis bearbeiten oder im Entwicklungsprozess bereits abgearbeitete Tätigkeiten erneut durchgeführt werden müssen. Dies kann bspw. im Fehlerfall oder bei geänderten Systembedingungen vorkommen.

Systemkonfiguration Eine Systemkonfiguration fasst die aktuellen Versionen von Ergebnissen an einem bestimmten Zeitpunkt der Entwicklung zusammen (vgl. [BK02], S. 6). Jede neue Version eines Ergebnisses führt zu einer neuen Systemkonfiguration.

Metamodell Das Metamodell bringt die Entwurfsergebnisse auf konzeptioneller Ebene in Beziehung. „Es stellt die atomatisierten Bestandteile aller Entwurfsergebnisse in der Form eines Datenmodells [...] übersichtlich dar" [Gut94], S. 14. Jedes Ergebnis hat dabei seine eigenen Attribute oder Ausprägungen. Die Komponenten eines Metamodells, ihre Beziehungen zueinander und ihre Attribute können durch ein Entity-Relationship-Modell (ERM) abgebildet werden (vgl. [Gut94], S. 24).

Diese Komponenten von Vorgehensmodellen werden in Abb. 5.1 in einem Ordnungsschema noch einmal graphisch miteinander in Verbindung gesetzt.

Anhand des Grades der Detaillierung, also ihrer Abstraktion bzw. Spezialisierung, lassen sich die Modelle des Softwareentwicklungsprozesses unterscheiden (vgl. [Bre98], [FBML98]): Während *Vorgehensstrategien* auf einer sehr abstrakten Ebene nur Beschreibungen zur Entwicklungsphilosophie, den Software-Werkzeugen und der Projektorgani-

[7] Bei Braun et al. lautet dieser Eintrag *specification document*; für die Darstellung in dieser Arbeit wurde er in den Eintrag *Entwurfsergebnisse* abgewandelt, da sich Braun et al. in [BWHW05] auf [Gut94] beziehen und dieser den hier verwendeten Begriff benutzt.

sation enthalten, also keine konkreten Arbeitsabläufe definieren, liefern *Vorgehensmodelle* auf dieser Basis das dazugehörende Muster zur Beschreibung des Entwicklungsprozesses. Vorgehensmodelle sind also Instanzen, die von einer oder mehrerer Vorgehensstrategien abgeleitet wurden. Auf dieser Ebene lassen sich Modelle einordnen, die nur für spezifische Projekttypen verwendet werden können[8]. Ebenso wie diejenigen, die allgemein gültig sind, also generisch verwendet werden können. Werden die in einem Vorgehensmodell generisch verwendete Komponenten (also Rollen, Ablauf, Werkzeuge) für ein reales Projekt festgelegt, ist dies ein so genanntes *Projektmodell*.

5.1.3. Entwicklungsschemata von Vorgehensmodellen

Da in der Praxis unterschiedliche Zielsetzungen bei Softwareentwicklungsprojekten existieren, sind aus dem Softwareengineering die verschiedensten Vorgehensmodelle hervorgegangen, die auf bestimmte Ziele hin optimiert wurden. Abhängig von ihrer Prozesssteuerung – aktivitäts-, ergebnis-, oder entscheidungsorientiert – können drei Familien von Vorgehensmodellen unterschieden werden (vgl. [BK02], S. 3ff., auch [Dow87]):

1. Phasen-, Wasserfall- und Schleifenmodelle,
2. prototypische Vorgehensmodelle und die
3. inkrementellen, evolutionären, rekursiven und iterativen Verbesserungsmodelle.

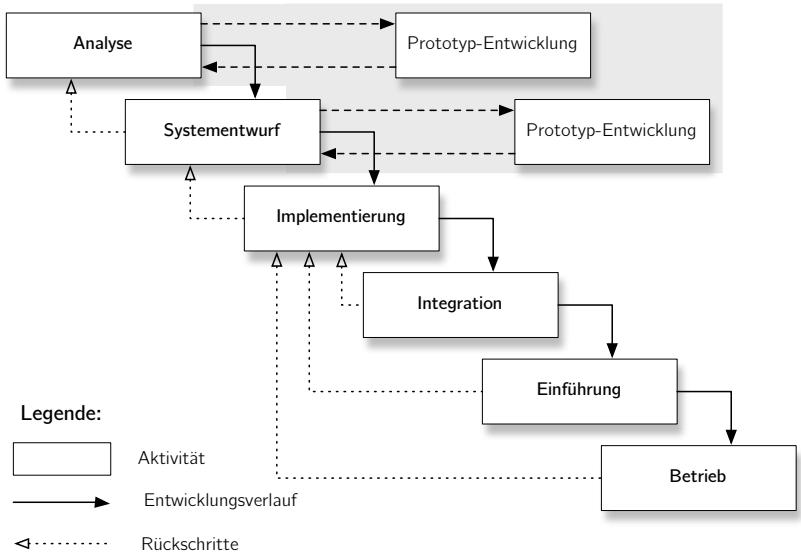

Abbildung 5.2.: Prozessarchitektur phasenorientierter Vorgehensmodelle (in Anlehnung an [BK02], S. 7).

[8] Man spricht hierbei auch von Entwurfsobjektbindung (vgl. [Fil05], S. 13).

Phasen-, Wasserfall- und Schleifenmodelle Das Phasenmodell wurde 1956 erstmals veröffentlicht. Hierbei wird der Softwareentwicklungsprozess in bestimmte Phasen aufgeteilt, denen Aktivitäten zugeordnet sind.

In der ersten Phase – der *Analysephase* – werden die Anforderungen an das System möglichst umfassend ermittelt und dokumentiert. Das Ziel dieser Phase ist ein gemeinsames und genaues Verständnis aller am Entwicklungsprozess beteiligten Rollen hinsichtlich des gewünschten Systems. Auf Basis dieser Anforderungen wird in der *Entwurfsphase* eine IT-Architektur entworfen, welche die Anforderungen auf ein realisierbares Softwaresystem abbildet. Dies beinhaltet bspw. den Entwurf der Benutzungsoberfläche, die Datenmodellierung und den Funktionsentwurf, sowie den Entwurf der Klassen, Methoden und Objektstrukturen. Das Ergebnis der Entwurfsphase ist der Systementwurf als Abschlussdokument, der einzelne Komponenten des Systems identifiziert und dokumentiert. In der *Implementierungsphase* wird dieser Systementwurf durch eine Programmiersprache in lauffähige Komponenten umgesetzt und getestet. Die Zusammensetzung der Komponenten zu einem lauffähigen Gesamtsystem folgt in der *Integrationsphase*. In der *Einführungsphase* wird das integrierte Gesamtsystem dann in die Informationsarchitektur des Auftraggebers übertragen. Während des *Systembetriebs* erfolgt die Aufrechterhaltung der Systemfunktionen durch Fehlerbehebung, sowie die Anpassung des Systems an sich ändernde Kundenwünsche und Umgebungsbedingungen.

Diese Phasen bauen konsequent aufeinander auf und werden sequenziell abgearbeitet (vgl. Abb. 5.2). Am Ende jeder Phase kann ein fertig gestelltes Ergebnis (bspw. Anforderungskatalog, Systementwurf, lauffähiges System) eine Qualitätssicherung durchlaufen.

Im Phasenmodell ist ein Rücksprung zu bereits abgeschlossenen Phasen, wie er unter bestimmten Bedingungen vorkommen kann, wie z. B. im Fall von fehlerhaften Ergebnissen aus früheren Phasen, nicht abgebildet. Im Gegensatz dazu erlaubt die Familie der Wasserfallmodelle eine begrenzte rückwärts gerichtete Iteration. Ob dabei ein Rücksprung nur zu der direkt vorhergehenden oder zu früheren Phasen erlaubt ist, wird durch die konkrete Ausprägung eines bestimmten Vorgehensmodells dieser Familie definiert (vgl. [BK02], S. 5).

Prototypische Vorgehensmodelle Vorgehensmodelle dieser Familie erlauben, wie auch Wasserfall- oder Schleifenmodelle, wiederholte Durchläufe bereits abgearbeiteter Phasen. Diese begründen sich entweder durch Rückschritte aus anderen Phasen aufgrund von Fehlern o. ä. oder durch Prototypen, die zu verschiedenen Zeitpunkten als fester Bestandteil im Entwicklungsprozess vorgesehen sind (vgl. [GS96]):

Explorative Prototypen dienen der Überprüfung unklarer oder schwieriger Teile der Anforderungen, die bereits im frühen Entwicklungsstadium stattfinden kann. Sie unterstützen die Kommunikation zwischen Anwendern und Entwicklern. Somit können die Anforderungen auf Basis der mit dem Prototypen gesammelten Erfahrungen angepasst und neu entwickelt werden.

Experimentelle Prototypen können zur Überprüfung unklarer oder schwieriger Designaspekte herangezogen werden, wie z. B. der Überprüfung der gewählten Architektur hinsichtlich der Anforderungen an die Performance.

Evolutionäre Prototypen erweitern und verbessern ein anfängliches Testsystem kontinuierlich. Diese Form des Prototyping steht im Widerspruch zu der Prozesssteuerung der phasenorientierten Modelle und gehört somit eher der Familie evolutionärer Modelle an (s. nächster Abschnitt).

Sowohl die Analyse- als auch die Designprototypen sollten möglichst schnell und kostengünstig umgesetzt werden können: Für die Analyse kann auch eine einfache Notation verwendet werden, die eine Ausführung der dokumentierten Anforderungen erlaubt (vgl. [BK02], S. 8). Oberflächen können auf dem Papier skizziert und für Designprototypen können auch existierende Softwaresysteme aus ähnlichen Projekten unter Vernachlässigung softwaretechnischer Standards angepasst und verwendet werden.

Inkrementelle, evolutionäre, rekursive und iterative Verbesserungsmodelle Basili und Turner stellten 1975 das iterative Verbesserungsmodell vor, welches sich im Vergleich zu den bisher vorgestellten Modellen einer anderen Prozessarchitektur bedient: Das eindimensionale phasenorientierte Vorgehen wird durch das Konzept der Inkremente abgelöst. Ein Inkrement wird dabei definiert als eine Teilmenge von Anforderungen, die durch Aktivitäten in den Bereichen Analyse, Entwurf, Implementierung, Integration, Installierung und Einsatz in sequenzieller Ordnung erstellt werden. Ein Inkrement besteht somit aus allen Ergebnissen für die durch ihn definierte Teilmenge der Systemanforderungen. Also aus einem lauffähigen installierten System, das aber u. U. noch nicht alle Anforderungen abdeckt. In Iterationsschritten werden dann immer weitere Anforderungen umgesetzt, womit Folgeinkremente Erweiterungen bereits bestehender Inkremente darstellen und die lauffähige erste Systemkonfiguration immer weiter ausbauen. Dieser Prozess endet erst, wenn alle Anforderungen erfüllt sind und sich auch keine neuen mehr ergeben durch geänderte Anforderungen oder veränderte Systembedingungen[9]. Der Entwicklungsprozess wird hierbei also in mehreren Entwicklungszyklen geplant, die jeweils gleiche Phasen durchlaufen. Zwar liegt diesem Modell ebenfalls ein Top-down-Ansatz zugrunde, die zunehmende Konkretisierung geht dabei jedoch nicht von Phasen, sondern von Entwicklungszyklen aus.

Bei den evolutionären Modellen wird der Softwareentwicklungsprozess als risikogetrieben angesehen. Hier findet nach jeder Iteration eine Verifikation der Entwicklungsziele und eine Verifikation der Entwicklungsergebnisse statt. Diese Analyse bildet dann die Entscheidungsgrundlage zur Auswahl der Anforderungen für das nächste Inkrement. Rekursive Vorgehensmodelle verwenden darüber hinaus das Konzept der Rekursion für ihren Systembegriff, der damit auch Teilsysteme und deren Teilsysteme usw. umfasst.

[9] Unter Umständen also erst, wenn das System deinstalliert wird.

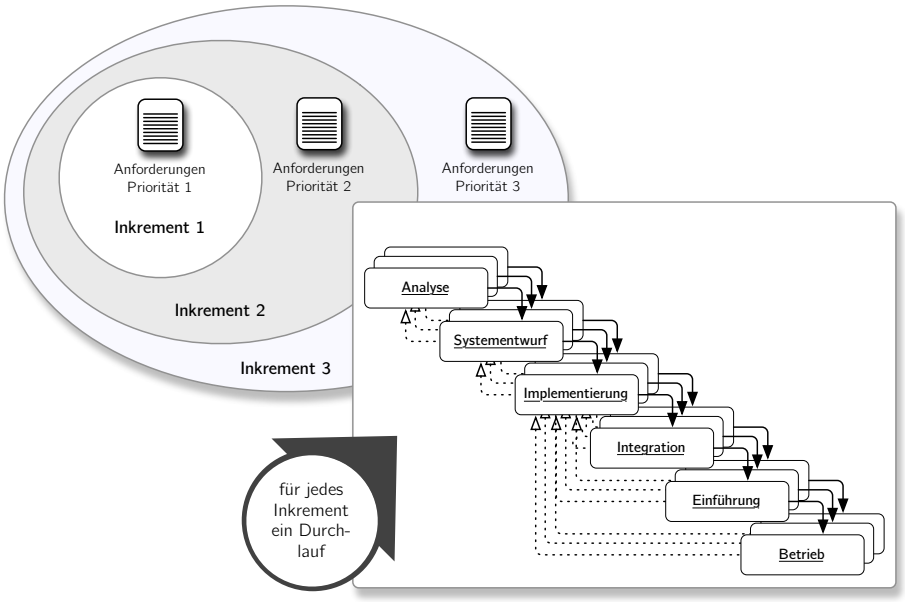

Abbildung 5.3.: Schrittweise Umsetzung von Anforderungen in inkrementellen Vorgehensmodellen (eigene Darstellung).

5.1.4. Entscheidungshilfe zur Auswahl der Prozesssteuerung für die Softwareentwicklung im universitären Kontext

Ein Entwicklungsprozess von Softwarekomponenten zur Unterstützung virtueller Lern- und Arbeitsszenarien ist innerhalb einer Universität oftmals mit einem festen Termin zur Einführung verbunden, nämlich mit dem Beginn eines neuen Semesters. Bis dann muss das neue System im Regelfall getestet und lauffähig in der universitären Informationsarchitektur integriert sein.

Für solche termingebundene Softwareentwicklungsprozesse eignen sich inkrementelle, evolutionäre Vorgehensmodelle grundsätzlich eher als phasenorientierte, bei denen es erst sehr spät im Projektverlauf zu einem ausführbaren Ergebnis kommt: Wenn sich die Anforderungen priorisieren lassen, können die Wichtigsten in den ersten Inkrementen zusammengefasst werden, in den Folgenden die weniger wichtigen. Somit wird ein Scheitern des Projektes verhindert, selbst wenn der Zeitplan nicht eingehalten werden kann: Die wichtigsten Anforderungen sind durch die ersten lauffähigen Inkremente bereits umgesetzt, getestet und installiert, womit auch zusätzlich das Risiko einer Fehlentwicklung gemindert wird.

Da in phasenorientierten Modellen die Implementierung auf einen vollständigen Entwurf aufsetzt, kann bei Terminüberschreitung eines Meilensteins der Auslieferungstermin der Software nicht unbedingt durch ein solches Weglassen von Teilfunktionalitäten gesi-

chert werden[10]. Auch bedingt die Notwendigkeit eines vollständigen Entwurfes ein sehr gutes Verständnis des zu entwickelnden Softwaresystems seitens der Architekten/Programmierer, weiterhin eine grundlegende Beschreibbarkeit der Anforderungen und ihre Stabilität, denn Weiterentwicklungen und Änderungen sind – anders als in inkrementellen Vorgehensmodellen – nicht als natürlicher Bestandteil der Softwareentwicklung im Modell vorgesehen.

Ein weiterer Vorteil von inkrementellen Vorgehensmodellen sind die Möglichkeiten, die im Projekt gesammelten Erfahrungen im gleichen Projekt weiter zu nutzen. Hier wiederholen sich die durchzuführenden Aktivitäten für jedes Inkrement, so dass neu Gelerntes im folgenden Inkrement direkt zur Anwendung kommen kann. In geringerem Maße bieten zwar auch Prototypen oder außerplanmäßige Rücksprünge im Projekt bei phasenorientierten Modellen diese Möglichkeiten. Zumindest Letzteres geht jedoch zu Lasten der Projektlaufzeit.

Dies lässt auch weitere Schlüsse auf die Kompetenzprofile einiger im Vorgehensmodell definierter Rollen zu: Phasenorientierte Modelle sollten aus diesem Grund bei einem festen Termin zur Fertigstellung nur von einem Team eingesetzt werden, das bereits viel Erfahrung mit ihren Werkzeugen und Methoden in vorhergehenden Projekten gemacht hat.

Aus softwaretechnischer Sicht bieten die phasenorientierten Modelle allerdings auch Vorteile: Die Implementierung basiert auf einem abgeschlossenen Entwurf, der die vollständigen Anforderungen enthält. Somit können Komponenten besser identifiziert, parallel entwickelt und integriert werden. Die Schnittstellenbeschreibungen zwischen den Komponenten erfolgen in diesen Vorgehensmodellen zur gleichen Zeit und sind damit per se aufeinander abgestimmt. Der Komplexitätsgrad einer Integration verschiedener Inkremente hängt im Gegensatz dazu allerdings davon ab, wie gut die Anforderungen partitionierbar waren[11], zum anderen von der Erweiterbarkeit der zugrunde liegenden Systemarchitektur. Inkrementelle Modelle begünstigen diesbzgl. allenfalls das Testen der Schnittstellen, da hier nicht alles zur gleichen Zeit integriert und getestet wird. Außerdem werden die wichtigsten Systemschnittstellen häufig als erstes umgesetzt und dadurch auch am häufigsten getestet.

Auch der Aufwand für das Versions- und Konfigurationsmanagement ist bei inkrementellen Vorgehensmodellen höher, da hier die Anzahl an Konfigurationen und Versionen bedeutend größer ist als bei den anderen Vorgehensmodellen: Zu jedem Inkrement existiert mindestens eine Konfiguration, die alle dazugehörenden Versionen der Ergebnisse enthält. Jede neue Version eines Ergebnisses führt gleichzeitig zu einer neuen Konfiguration. In phasenorientierten Modellen ist die Anzahl der Versionen (und somit auch der Konfigurationen) eher klein, denn sie entstehen nur dann, wenn einmal abgearbeitete Tätigkeiten erneut durchgeführt werden, also planmäßig nur bei Prototypen oder außerplanmäßigen Rückschritten. Im besten Fall liegt eine Version pro Ergebnis vor und eine Konfiguration für das lauffähige und installierte Gesamtsystem.

[10] Zwar wird in prototypischen Vorgehensmodellen ein ausführbarer Prototyp von den Endbenutzern leicht als Endsystem erkennt, dieser sollte aber keinesfalls so eingesetzt werden.

[11] D. h., wie viele Anhängigkeiten zwischen den verschiedenen Teilmengen der Anforderungen existieren.

5.2. Referenzmodelle zur Entwicklung von E-Learning-Komponenten

Vorgehensmodelle in der Domäne des computergestützten Lernens haben – je nach Charakteristika des Endproduktes (z. B. Lerninhalt als virtuelles Modul, Lernwerkzeug, abgeschlossene Lernumgebung oder gar ein ganzer virtueller Studiengang) – unterschiedliche Aktivitäten, Techniken und (Zwischen-) Ergebnisse zum Inhalt. Unabhängig der vorgenommen Klassifizierung von Modellen hinsichtlich ihrer Prozesssteuerung können in diesem Anwendungskontext daher neben den eher technisch orientierten Modellen des Softwareentwicklungsprozesses auch Vorgehensmodellen mit didaktischer Schwerpunktsetzung respektive hybride Modelle unterschieden werden (vgl. [Blu98], S. 153f, [Ker01], S. 325 und [Paw01], S. 79). Letztere gewichten die technischen und didaktischen Aspekte gleichermaßen[12].

Im Folgenden sollen bekannte Referenz- und Vorgehensmodelle zur Entwicklung von E-Learning-Komponenten erläutert und gegenübergestellt werden, um ein passendes Vorgehensmodell für die Softwareentwicklung im universitären Kontext zu identifizieren. Davon soll eine Prozessstruktur abgeleitet und um die Besonderheiten der normsprachlichen Spezifizierung und der Produktlinien-orientierten Umsetzung sowie um Rollen, Techniken und ein Metamodell ergänzt werden.

5.2.1. Didaktisch-orientierte Modelle

Die Strukturierung des Entwicklungsprozesses in didaktisch orientierte Vorgehensmodelle zur Entwicklung von Lernumgebungen geht zurück auf die Anwendung des generischen *systemic approach*[13] auf Bildungssysteme in den fünfziger Jahren (vgl. [Iss97], S. 201).

Daraus entstand eine Vielzahl von präskriptiven Vorgehensmodellen, die zur Erreichung definierter Lehrziele den gesamten Prozess – von der Analyse und Konzeption von Lernsystemen bis hin zur Umsetzung und Einsatz – abbilden[14]. Zwar unterscheiden sie sich primär durch einen individuellen Grad der Detaillierung, Linearisierung und Interaktivität (vgl. [Blu98], S. 150) – so gibt es ziel- und inhaltsspezifische Modelle für

[12] Einen anderen Ansatz zur Kategorisierung von Vorgehensmodellen zur Entwicklung von Bildungsangeboten mit E-Learning-Komponenten wählt Hambach in [Ham05a]. Für ihr dreistufiges Schema betrachtet sie zusätzlich Modelle der Designtheorie für die Gestaltung von Produkten (Produktdesign), Medien (Mediendesign) und Kommunikation (Kommunikationsdesign). Allerdings merkt sie an, dass einerseits nur wenige Belege für Vorgehensmodelle in den verschiedenen Teildisziplinen des Designs ermittelt werden konnten (S. 56), und andererseits ihr Ansatz zur Kategorisierung für die praktische Einordnung von Vorgehensmodellen anhand der Literatur zu feingliedrig ist. Aus diesem Grund werden in dieser Arbeit die Modelle der Designtheorie außer Acht gelassen.

[13] Der *systemic approach* bzw. das *general systems design* beschreibt ein heuristisches Verfahren zur Entwicklung von Systemen, welches auf verschiedene Problembereiche angewandt werden kann, wie z. B. zur Softwareentwicklung, Organisationsentwicklung oder zum Management. Es besteht aus den Schritten Analyse, Planung, Entwicklung sowie Evaluation und Revision (vgl. [Iss97], S. 200).

[14] Für diese Modelle haben sich in der Literatur unterschiedliche Begriffe etabliert: *models of instructional design*, *Instructional Systems Development* (IDD), *Instructional Design and Development* (IDD), *Instructional Systems Design* (ISD), *Instructional Systems Technology* (IST), oder im Deutschen auch *systematisches Instruktionsdesign* (ID).

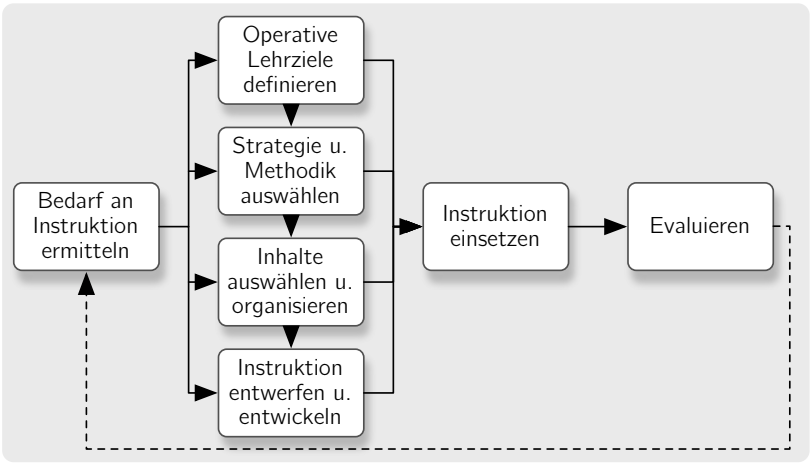

Abbildung 5.4.: Evolutionäres Prozessmodell des Instructional System Design

Lehrpläne und Schulkonzepte, Unterrichtseinheiten oder konkrete Lehr-/Lernprozesse[15] – ihnen zugrunde liegt aber der gleiche, durch behavioristische und kognitivistische Lernparadigmen geprägte Modellansatz: Auf Basis operativ definierter Lehrziele (also dem gewünschten Ergebnis) und einer *Instruktionssituation* wird unter Zuhilfenahme eines probabilistischen Regelsystems[16] eine optimale Medienwahl abgeleitet und eine entsprechend optimierte Medienproduktion vorgeschlagen. Aktuelle Modelle des Instructional Designs sind diesbzgl. komplexer. Sie leiten darüber hinaus auch eine *Lehrstrategie* bzw. eine Spezifikation von Lehrzielen ab (vgl. [Ker01], S. 331).

Viele ID-Modelle sehen umfangreiche Evaluationsaktivitäten während und nach dem Prozessdurchlauf vor (vgl. Abbildung 5.4), deren Ergebnisse Rückkopplungen mit bereits abgeschlossenen Phasen oder Aktivitäten begründen können[17]. Um schon zu einem frühen Zeitpunkt Schwächen erkennen und beheben zu können, ist bereits der Entwurf der Instruktionsmaterialien mit Personen der Zielgruppe zu prüfen. Neben dieser formativen Evaluation untersucht eine summative Evaluation nach Durchführung der Instruktion die relative und absolute Qualität der Bildungsmaßnahme. Weiterführende Ansätze betrachten die Evaluation systematisch als durchgehende Aktivität für alle Phasen und nutzen diese als Mechanismus zur Steuerung und Kontrolle des gesamten Projektes (vgl. Tabelle 5.1).

Die generelle Kritik am Objektivismus und Instruktionalismus wurde in dieser Arbeit bereits im Kontext der lerntheoretischen und kognitionspsychologischen Grundlagen thematisiert (S. 82). Diese Kritik greift auch beim Instruktionsdesign. In den Regelwerken wird nämlich davon ausgegangen, dass das Ergebnis von Instruktion für alle Lernenden

[15] Flechsig und Haller beschreiben in [FH77] diese einzelnen Anwendungsbereiche der Didaktik anhand eines Modells von fünf Ebenen.

[16] Die Methoden des ID sind nicht deterministisch, sondern bieten auf Basis empirischer Erkenntnisse wahrscheinliche Beziehungen zwischen Wenn-Dann-Komponenten an.

[17] Somit ist die Prozesssteuerung von Modellen des Instruktionsdesigns als evolutionär zu klassifizieren.

im Wesentlichen gleich ist (deterministische Orientierung, vgl. [Blu98], S. 151f., [Ker01], S. 132). Das bedeutet, dass eine objektiv wahrnehmbare Realität bzw. objektives Wissen außerhalb des individuellen Lernens nicht anerkannt wird. Neuere ID-Modelle wurden jedoch bereits durch konstruktivistisch geprägte Lerntheorien beeinflusst, in denen darstellend erklärende Lehrstrategien (Anleitung und Expertenwissen) mit einem aktiv-explorierenden Lernen und Reflexionsphasen verschränkt werden (bspw. *cognitive apprenticeship*, siehe S. 84). Von radikalen Vertretern des Konstruktivismus werden diese Modelle trotzdem kaum akzeptiert (vgl. [Sch97b], S. 88ff.).

Analyse	**Design**	**Produktion**	**Implementation**
Bedürfnisse und Bedarf erfassen, Merkmale des Lerners analysieren, Rahmenbedingungen und allgemeine Ziele feststellen	Lehrziele und Lehrinhalte, Lernerfolgskontrolle, Trägermedium, Managementstrategie	Programmierung, Dokumentation, Planung der Distribution	Management der Bildungsmaßnahme, Durchführung
Evalutation der Machbarkeit	Formative Evaluation	Summative Evaluation	Evaluation der Implementation

Tabelle 5.1.: Elemente gängiger ISD-Modelle (aus [Ker01], S. 331).

5.2.2. Vorgehensmodelle für die Web-Entwicklung

Im Laufe der letzten Jahre wurden viele Methoden zur Entwicklung Web-basierter Systeme konzipiert. Dabei wurde Methoden in Anlehnung an traditionelle Modellierungstechniken wie beispielsweise dem ER-Diagramm (ERM) oder der *Object Modelling Technique* (OMT) entwickelt, oder aber auf Basis der Stärken existierender Methoden weiterentwickelt (vgl. [SK03], S. 70).

Der Fokus der meisten Methoden zur Web-Entwicklung konzentriert sich vorrangig auf den Entwurf- und Implementierungsteil der Anwendung, weshalb sie nicht didaktisch, sondern eher softwaretechnisch orientiert sind. Ebenfalls ist vielen von ihnen die Trennung von Aspekten des Datenmodells, der Navigation, der Benutzungsschnittstelle und der Präsentation gemein. Trotzdem lassen sich – abhängig von Abstammung und Fokus – Methoden mit hypertext-, daten- oder objektorientierten Paradigmen voneinander unterscheiden.

5.2.2.1. Hypertext-orientierte Methoden

Im Gegensatz zu den CBTs und WBTs liegen in Hypermedia-Systemen die Inhalte nicht ausschließlich linear, sondern insbesondere vernetzt vor. Somit kann ein Top-down-Vorgehen die Entwicklung dieser Systeme nicht sehr effizient unterstützen. Es werden daher andere Vorgehensmodelle benötigt, die auf die spezifischen Entwicklungsebenen von Hypermedia-Systemen eingehen.

Nach [Sch92] sind dies (vgl. [GS96])[18]:

Generative Ebene Erstellung und Verwaltung der Text- und multimedialen Dokumente. Die Präsentation des Lehrstoffs kann Texte, Tabellen, Grafiken, Bilder, Animationen, Audio- und Videosequenzen sowie weitere Software umfassen, die Multiple-Choice, Simulationen, etc. implementieren können.

Formale Ebene Erstellung der Verknüpfungen zwischen den Texteinheiten, um aus dem linearen Text den netzwerkartigen Hypertext zu gestalten. Durch eine Aufgliederung der Lehrstoffinhalte in logische Einheiten werden die Knoten (*nodes*) des Hypertextes geschaffen, die mit Hilfe von Verweisen untereinander in Beziehungen gebracht werden[19].

Multimediale Ebene Erweiterung des Hypertexts um multimediale Objekte zu Hypermedia. Dabei werden die Objekte als selbstständige Module oder als Teile bestehender Hypertextknoten in das Netz integriert.

Diese Entwicklungsebenen wurden von Vorgehensmodellen adressiert, die insbesondere in den 1990er Jahren konzipiert wurden, also in der Zeit, in der Hypermedia-Systeme eine große Verbreitung fanden. Ein strukturierter Hypermedia-Entwurf umfasst insbesondere das logische Grundgerüst zur semantischen Vernetzung von Inhalten. Objekte einer fachlichen Domäne werden dazu auf Komponenten aufgeteilt, die wiederum mit so genannten Einheiten assoziiert werden können. Einheiten kapseln verschiedene Informationsdarstellungen[20]. Durch die freien Verweise zwischen Objekte, Komponenten und Einheiten entsteht ein *Hypergraph*, auf den dann über verschiedene Navigationsstrukturen wie Menüs oder Indizes zugegriffen werden kann. Eine konkrete Anwendung wird darauf basierend über die Instanzierung des Hypergraphen und der Erstellung einer *Browsing Semantic* erstellt. Auf diesem Weg können verschiedene Hypermedia-Anwendungen auf Basis desselben Hypergraphs erstellt werden.

Als erstes Modell für einen strukturierten Hypermedia-Entwurf bildete das *Hypertext Design Model* (HDM, vgl. [GPS93])[21] eine frühe Basis auch für datenorientierte Methoden. Im Rahmen der Entwicklung von Hypermedia-Systemen wurde HDM unter verschiedenen Aspekten wie bspw. Automatisierung weiterentwickelt, wodurch die neuen Methoden *W2000* und *HDM-Lite* entstanden.

Ein neueres Hypertext-orientiertes Vorgehen, das nicht auf HDM basiert, ist die *Web Site Design Method* (WSDM[22]), die an der Vrije Universiteit Brussel zwischen 1998 und

[18] Eine ganz ähnliche Sicht auf die Ebenen des Entwicklungsprozesses wird in [B+96] beschrieben: 1. *Entwurf des logischen Grundgerüsts* durch die Modellierung von Schemata, 2. *Entwurf der Nutzung* durch die Spezifikation von Inhalten, konkreter Zugriffsstrukturen und Navigationsunterstützung sowie 3. *Entwurf der Präsentation* durch die Bestimmung des Layouts eines Hypermedia-Systems.

[19] Hierbei können neben strukturbeschreibenden und bedeutungsabhängigen auch benutzerorientierte Beziehungen unterschieden werden, welche vom Benutzer selbst erzeugt werden können (vgl. [GS96]).

[20] Die Aufteilung auf Komponenten beschreiben also strukturelle, die Assoziation mit Einheiten perspektivische Verweise.

[21] Eine Verbesserung des Modells lag mit HDM2 vor, welches zusätzlich Navigationsmuster wie *Guided Tours*, Menüs oder Indizes umfasst.

[22] Informationen sind auf den Seiten der VU Brüssel unter `http://wise.vub.ac.be` bei Prof. De Troye und Sven Casteleyn zu finden. Letzter Zugriff: 31.07.2008.

2003 entwickelt wurde. Die Entwicklungsgrundlage bildet bei dieser Methode der Entwurf eines Aufgabenmodells, in dessen Verbindung ein Domänenmodell mit entworfen wird[23]. Aus beiden Modellen wird dann das Navigationsmodell entwickelt. Aspekte der Präsentation werden hierbei nicht behandelt.

5.2.2.2. Datenorientierte Methoden

Im Fokus datenorientierter Methoden liegt die Modellierung von datenbankgetriebenen Web-Anwendungen. Dabei werden die Konzepte des Hypergraphen mithilfe von Entity-Relationship-Modellen (ERM) abgebildet und um domänenspezifische Fachkonzepte erweitert. Beispiele für datenorientierte Methoden sind die *Web Modeling Language* (WebML[24]) und die *Relationship Management Methodology* (RMM, vgl. [ISB95]).

WebML ist ein sehr reichhaltiges Modell, das eine deutliche Präferenz auf die Modellierung der Domänenstruktur und davon abgeleitet der Navigation der Web-Anwendung hat. Eine darauf basierende Abbildung auf verschiedene Sichten erfolgt über die Adaption eines Rollenmodells.

Im Gegensatz zu WebML ist die RMM eine Weiterentwicklung des Hypertext Design Models. Da dieses ein Vorgehen nur implizit modelliert, definiert die Relationship Management Methodology – neben der Erweiterung um ein relationales Datenmodell – sechs Entwurfsphasen[25] und eine Implementierungsphase.

5.2.2.3. Objektorientierte Methoden

Ebenso wie das RMM konkretisiert das *Object Oriented Hypertext Model* (OOHDM, vgl. [RSLC95] und [SPM99]) das Hypertext Design Model im Sinne eines Vorgehensmodells. Der Fokus liegt hierbei jedoch in der Unterstützung eines objektorientierten Ansatzes. Die Ziele des Designs sind in der Hauptsache die Unabhängigkeit der einzelnen Phasen, sowie die methodeninvariante Wiederverwendung von Hypermedia-Elementen und den Entwurf ohne Berücksichtigung der späteren Implementierung. Die vier Phasen des OOHDM sind weiter gefasst als die der RMM. Sie umfassen die Domänenanalyse, das Navigationsdesign, ein abstraktes Interface-Design und die Implementierung.

Erweiterungen des OOHDM um Techniken des Semantic Webs[26] mündeten in der *Semantic Hypermedia Design Method* (SHDM), welche durch die Deklaration von Inhalten als Ressourcen und ihrer Charakterisierung durch Ontologiesprachen die OOHDM

[23] Das Domänenmodell ist dem Aufgabenmodell bei dieser Methode folglich untergeordnet.

[24] Weitere Informationen zur Web Modeling Language sind unter `http://www.webml.org` verfügbar, kommerzielle Werkzeugunterstützung ist unter `http://www.webratio.com` zu finden. Letzter Zugriff: 31.07.2008.

[25] Entwurfsphasen der RMM: (1) *Entity-Relationship-Entwurf*, Modellierung der Fachdomäne; (2) *Entity-Entwurf*, Aufbau des Hypergraphen durch Vernetzung; (3) *Navigations-Entwurf*, Definition von Zugriffspunkten und Indizes; (4) *Konvertierungsprotokoll-Entwurf*, Transformation in XML; (5) *Entwurf der Benutzungsschnittstelle* und (6) *Entwurf des Laufzeitverhaltens* zur Beschreibung der Präsentationsschicht (vgl. [ISB95]).

[26] Speziell zur Metadatenbeschreibung von Anwendungen, bspw. das *Ressource Description Framework* (RDF) und die *RDF Vocabulary Description Language* (RDFS) zur Beschreibung von Schemata, sowie die *Web Ontology Language* (OWL). Alle drei Techniken sind vom W3C spezifiziert und unter `http://www.w3.org` online verfügbar.

in puncto Vorgehen auf die Verwendung aktuellster Technologien modernisiert hat. Der objektorientierte Ansatz und die strikte Trennung von Inhalt und Layout wurden beibehalten. Die technische Unterstützung zur Entwicklung semantischer Web-Anwendungen nach der SHDM kann durch MVC-Frameworks wie HyperDE[27] erfolgen.

Dem OOHDM sehr ähnlich und auf einen durchgehenden Entwicklungsprozess abzielend ist das an der LMU München entwickelte *UML-Based Web Engineering* (UWE). Der Entwicklungsprozess lehnt sich dabei stark an den UML-basierten *Rational Unified Process* (RUP) an. Somit liefert UWE ein sehr reichhaltiges, aber auch komplexes Modell, das per se objektorientiert und auf interaktive Web-Anwendungen ausgerichtet ist. Ein Vorteil hierbei ist die explizite und deutliche Strukturierung des Präsentationsbereichs, die in der Form nicht in allen Methoden vorhanden ist.

Die *Object-Oriented Hypermedia Method* (OO-H) ist ebenfalls eine neuere Methode, welche die Vorteile von WebML, OOHDM und UWE in sich vereint. Aufgrund des von der OOHDM übernommenen Konzepts einer starken Navigationsausprägung ist sie insbesondere für die Entwicklung von Hypermedia-Anwendungen geeignet. Eine Werkzeugunterstützung ergänzt diese Methode[28]. Ein Überblick über alle hier genannten Methoden zur Entwicklung von Web-Anwendungen wird in Tabelle 5.2 gegeben.

5.2.3. Hybride Modelle

Die hybriden Modelle kombinieren inhaltliche und methodische Aspekte aus Vorgehensmodellen der didaktischen und der softwaretechnischen Disziplin. Diese Modelle eignen sich im Speziellen für die Konzeption und Implementierung von Software für Lehr- und Lernzwecke (vgl. [Ham05a], S. 51). In der Literatur finden sich Modelle zur Umsetzung klassischer hypermedialer Lernumgebungen und in Bezug auf die Systemklasse offenere Lernumgebungen.

5.2.3.1. Hybride Modelle zur Entwicklung hypermedialer Lernumgebungen

Blumstengel lehnt in [Blu98] ihr Modell zur *Entwicklung hypermedialer Systeme* an das Instruktionsdesign an und übernimmt die Evaluation als integralen Bestandteil der verschiedenen Phasen. In der ersten Phase werden dazu bereits die Evaluationsinstrumente festgelegt. Das Prozessmodell ihres Ansatzes besteht aus den vier Hauptphasen Bedarfsanalyse, Alternativentwicklung, Produktion, und Anwendung und Evaluation. Die Phasen werden mit durchgehenden Projektmanagement- und Revisionsprozessen flankiert. Der softwaretechnische Produktionsprozess besteht wiederum aus den Unterprozessen Planung, Entwurf (neben dem Screendesign auch aus einem didaktischen Feinentwurf), Implementierung, Integration und Test. Eine darüber hinausgehende Detaillierung von Vorgehensschritten und ein Rollenmodell sind nur stellenweise vorhanden, eine Methodensammlung fehlt.

[27] HyperDE ist eine Kombination aus MVC Framework (basierend auf *Ruby on Rails*) und einer Entwicklungsumgebung für Semantic Web-Anwendungen. Mehr Informationen sind unter `http://www.tecweb.inf.puc-rio.br/hyperde/wiki/WikiStart` verfügbar. Letzter Zugriff: 31.07.2008.

[28] VisualWADE Tool, verfügbar unter `http://gplsi.dlsi.ua.es/iwad/ooh_project/cawetool.htm`. Letzter Zugriff: 31.07.2008.

Nagl et al. beschreiben in [NBS+99] ein *Vorgehensmodell für die industrielle Entwicklung multimedialer Lehr- und Lernsysteme*, welches primär eine Weiterentwicklung der Elemente klassischer Softwaretechnik ist, die sich seitens der multimedialen Lehr- und Lernsysteme ergeben. Es setzt sich im Grunde aus Prozessen des Software-Managements, der -Entwicklung und -Qualitätssicherung zusammen. Das SW-Management umfasst – wie bei Blumstengel auch – Querschnittsfunktionen wie Planung und Controlling. Die Software-Entwicklung enthält die typischen Phasen wie Entwurf und Implementierung, darüber hinaus auch Wartung und Pflege, die sich einer Einführungsphase anschließen. Zur Prozesssteuerung wird ein sequenzielles Vorgehen vorgeschlagen, wenn industrielle Rahmenbedingungen wie Budgetierung und Terminierung eingehalten werden sollen; können diese vernachlässigt werden, kann auch eine evolutionäre Steuerung eingesetzt werden, die mehr Rückkopplungen impliziert. Neben den klassischen softwaretechnischen Merkmalen fließen auch inhaltliche und mediendidaktische Qualitätsmerkmale in die Software-Qualitätssicherung mit ein.

	Methode	Notation	Anforderungen	Content	Hyperlinks	Präsentation	Dynamik	Stärken
Hypertext	HDM-lite	ER + eigene	✗	✗	✓	✓	✗	Prozess zur Transformation der Modelle, automatische Generierung
	WSDM	eigene	✗	✓	✓	✗	✓	benutzerzentrierte Vorgehensweise bei Analyse
Daten	RMM	ER + eigene	✗	✓	✓	✓	✗	Hypertext-Modellierung, vordefinierter Prozess
	WebML	ER, UML	✓	✓	✓	✗	✓	ausgereifte Notation, DB-Integration
Objekte	OOHDM	UML + eigene	✓	✓	✓	✓	✗	mächtige Konzepte für kontextuelle Navigation
	SHDM	UML + eigene	✓	✓	✓	✓	✗	Semantic Web, gute Entwicklungsumgebung
	UWE	UML	✓	✓	✓	✓	✓	RUP-basierter Prozess
	OO-H	UML + eigene	✓	✓	✓	✗	✓	gute Werkzeugunterstützung

Tabelle 5.2.: Überblick über Methoden zur Entwicklung von Web-Anwendungen (in Anlehnung an [SK03], S. 72)

Ein ähnliches Modell spezifizierte Yass in [Yas00]. In Analogie zum Modell von Nagl et al. ist auch dies eine logische Erweiterung eines Phasenmodells zur Softwareentwicklung um didaktische Konzepte. Hierbei werden in der Projektdefinitionsphase Lernziele und Lerneinheiten spezifiziert und die Didaktik der Lerneinheiten aufgebaut. Ebenfalls werden die didaktischen Aspekte in der Test und Entwurfsphase explizit berücksichtigt, wo auch auf didaktische und inhaltliche Fehler, gestalterische Mängel sowie Abweichungen von den Lernzielen geprüft und korrigiert wird. Als Prozesssteuerung wird von Yass ein prototypisches oder evolutionäres Vorgehen vorgeschlagen.

Das *Vorgehensmodell zur Erstellung virtueller Bildungsinhalte* hat Klein zur Konstruktion wiederverwendbarer hypermedialer Kurse konzipiert (vgl. [KS01] und [Kle02]). Die Prozessarchitektur orientiert sich dabei am Wasserfallschema, Rückkopplungen sind zugelassen. Die einzelnen Phasen sind durch Vorgehensschritte detailliert beschrieben. Didaktische Aspekte werden in der Analysephase berücksichtigt, wo Lernziele, -eigenschaften und -inhalte in natürlicher Sprache beschrieben werden. Darüber hinaus werden in der Designphase Lerninhalte durch Module und entsprechenden Beziehungen modelliert, die Entwürfe von Lehrstoff, Lernszenarien, Benutzungsschnittstellen und Datenmodellen umfassen.

Pawlowski hat in [Paw01] mit dem generischen *Essener-Lern-Modell* (ELM) eine Prozessarchitektur spezifiziert, in der sich Design- und Entwicklungsprozesse durchgängig von der Curriculumsentwicklung bis zur Umsetzung einzelner Lerneinheiten durchziehen. Ebenso Querschnittsprozesse wie Projektmanagement. Genau wie Yass empfiehlt Pawlowski eine evolutionäre Steuerung (S. 124f.). Eine Prototypentwicklung ist ebenfalls als Teil der Implementierungsphase mit berücksichtigt (S. 123). Didaktische Aspekte umfassen Prozesse der Curriculumsentwicklung sowie die Verwendung von Lerntechnologiestandards. Darüber hinaus sind der didaktische Kontext, Aktoren, Lernziele und Methoden als Komponenten im Domänenmodell explizit vorgesehen.

5.2.3.2. Referenzmodell für Qualitätsmanagement und Qualitätssicherung

Neben den Vorgehensmodellen zur Entwicklung von klassischen Lernumgebungen existiert eine allgemein akzeptierte und harmonisierte Prozessarchitektur, die bestehende Ansätze integriert und sich nicht auf die Systemklasse hypermedialer Lernumgebungen bzw. WBTs beschränkt. Dieses *Referenzmodell für Qualitätsmanagement und Qualitätssicherung bei der Planung, Entwicklung, Durchführung und Evaluation von Bildungsprozessen und -angeboten* ist beim Deutschen Institut für Normung als DIN PAS 1032-1:2004 spezifiziert. Die PAS (*Publicly Available Specification*) wurde von einem Experten-Team aus allen Bereichen der Bildung und Weiterbildung entwickelt. Sie basiert somit auf Erfahrungen in den Anwendungs-, Forschungs- und Entwicklungsbereichen des E-Learnings. Das Referenzmodell der PAS, aus dem auch der neue Standard ISO/IEC-Norm 19796-1:2005 hervorgegangen ist, basiert in der Hauptsache aus einem generischem Prozessmodell, bestehend aus sieben Prozesskategorien mit insgesamt 38 Prozessen (Abb. 5.5), und einem umfangreichen generischen Beschreibungsmodell. Das Beschreibungsmodell impliziert u. a. ein Rollenmodell (die Zuordnung von Aktoren zu Prozessen), eine Methodensammlung und einen Qualitätskriterienkatalog.

Da keine logischen oder zeitlichen Beziehungen zwischen Prozessen definiert sind und es für einzelne Prozesse ebenfalls keine eindeutige Zuweisung von Methoden gibt, muss bei Anwendung des generischen Modells zunächst eine individuelle Auswahl, Anpassung und ggf. Erweiterung der Prozesse vorgenommen werden.

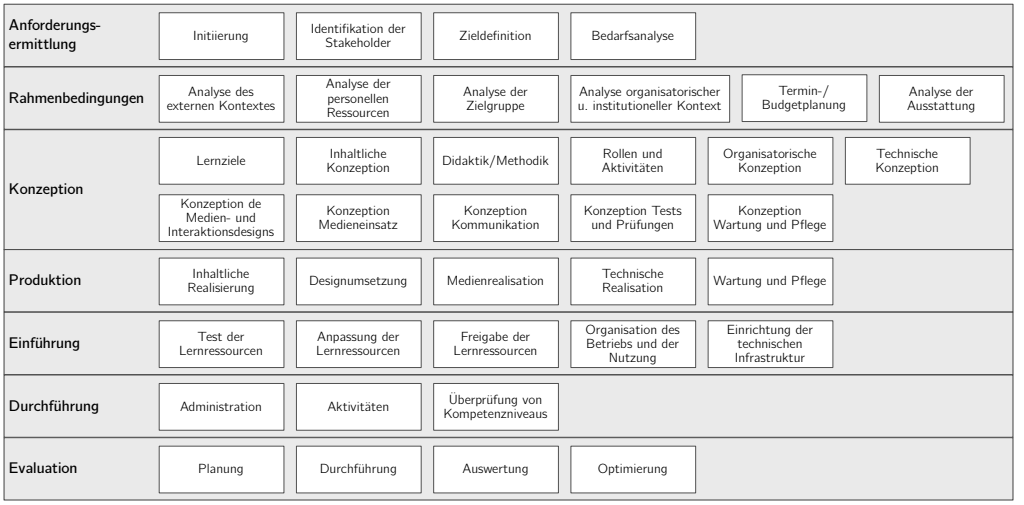

Abbildung 5.5.: Die Prozessarchitektur der DIN PAS 1032-1:2004 (vgl. [DIN04a], S. 10ff.)

Im Gegensatz zum *Systems Approach*, auf dem Modelle des Instruktionsdesigns basieren, wird bei der PAS von der Annahme ausgegangen, dass nicht immer eindeutige Kriterien zur Evaluierung einer Lernressource oder eines Lernszenarios definiert werden können. Daher werden verschiedene Qualitätsanforderungen formuliert, denen die Gestaltung und Durchführung der Entwicklungs- und Lernprozesse genügen sollen. Neben den didaktischen Aspekten müssen allerdings auch die Kriterien ISO 9241 für Softwareprodukte erfüllt sein. Somit ist die PAS ein transparenter Rahmen zur Planung der Entwicklung computergestützter Lernwerkzeuge, der neben einer Referenzprozessarchitektur einen Fokus auf die Qualitätsaspekte bei Prozessen und Produkten bietet. Zur Prozesssteuerung selbst wird in dem Referenzmodell keine Vorgabe gemacht. Sie muss im Rahmen der Instanzierung dieses Modells definiert werden.

5.2.4. Zusammenfassung und Bewertung

In diesem Unterkapitel wurden didaktisch orientierte, technisch orientierte und gemischt orientierte Vorgehensweisen zur Durchführung von Entwicklungsprojekten im E-Learning-Kontext vorgestellt. Es soll nun eine für den Anwendungskontext der Arbeit geeignete Prozessarchitektur identifiziert werden, die dann in einem nächsten Schritt auf die besonderen Anforderungen normsprachlicher Spezifizierung und Produktfamilien-orientierte Softwareentwicklung angepasst wird.

Die didaktisch orientierten Modelle des Instruktionsdesigns bieten durch ihre deterministische Prozessstruktur und ihrer Evaluationskomponenten eine gute Planungssicherheit für Methoden des Projektmanagements. Wichtige Aussagen über den Entwicklungsstand eines Lernsystems können dadurch schnell getroffen werden. Zusätzlich erlaubt die Rückkopplung von Maßnahmen zur Qualitätssicherung zu vorherigen Phasen und Aktivitäten ein iteratives Vorgehen. Die Regelsysteme komplexerer ID-Modelle schränken einen hochschulweiten Einsatz jedoch stark ein. Zum einen steht die zur operativen Definition von Lehrzielen notwendige aufwändige Zielgruppenanalyse oft nicht im Verhältnis eines didaktisch hochwertigen Lernangebots (vgl. [Iss97], S. 215). Genauso scheuen viele Anwender die umfangreiche formative Evaluation (vgl. [Ker01], S. 326). Zum anderen sind ID-Modelle für den Bereich der Printmedien konzipiert worden. Sie vernachlässigen daher in der Prozessarchitektur die Entwicklung von Medien bzw. Softwaresystemen. Darüber hinaus – und das ist der entscheidende Punkt – sind ID-Modelle am Entwurfsobjekt gebunden. Sie sind per se nicht generisch, sondern immer nur situativ gültig respektive nur für bestimmte Szenarios einsetzbar.

Die vorgestellten technisch orientierten Modelle zur Entwicklung von Lernsystemen beschreiben ein Vorgehen unterschiedlich detailliert. Aspekte wie Anforderungserhebung, Einführung und Evaluation werden teilweise vernachlässigt und sich nur auf die Softwareentwicklung fokussiert. Für den praktischen Einsatz macht dies eine vorherige Einbettung in ein übergeordnetes Schema notwendig. Darüber hinaus konzentrieren sich alle Modelle auf die Systemklasse der hypermedialen Lernumgebungen respektive Web based Trainings, sind also ebenfalls am Entwurfsobjekt gebunden und nicht generisch.

Hybride Modelle berücksichtigen sowohl technische als auch didaktische Aspekte. Dabei sind manche eher an das didaktische Design angelehnt (bspw. Blumstengel), die Mehrheit der Modelle basiert aber auf Softwareentwicklungsmodellen, die um didaktische Aspekte angereichert wurden (wie bspw. Nagl et al. und Yass). Bis auf das DIN PAS Referenzmodell unterstützen sie die Entwicklung klassischer Lernumgebungen, und lassen sich nicht ohne weiteres zur Entwicklung lernerzentrierterer Szenarien heranziehen.

Das DIN PAS Referenzmodell berücksichtigt als hybrides Modell ebenfalls technische und didaktische Aspekte. Im Vergleich zu den anderen vorgestellten Modellen ist es relativ jung und basiert als DIN-Spezifikation auf den kombinierten Erfahrungen von Experten in der Anwendung, Entwicklung und Forschung von E-Learning-Umgebungen. Mit einem detaillierten Prozessmodell, einer Methodensammlung und einem Qualitäts-Kriterienkatalog ist dieses Modell am umfangreichsten. Durch seine Referenzfunktion ist es per se generisch. Für bestimmte Szenarios oder E-Learning-Systemklassen lassen sich Modellinstanzen ableiten und anpassen.

Für den Kontext dieser Arbeit bietet sich daher eine Instanzierung und Anpassung der PAS 1032-1:2004 an, die im Folgenden vorgenommen werden soll.

5.3. Konzeption der Methode auf Basis der DIN-PAS 1032-1:2004

In diesem Unterkapitel soll das Referenzmodell der PAS, also die Prozessarchitektur und das Beschreibungsmodell, auf den spezifischen Kontext dieser Arbeit angepasst und erweitert werden. Dazu wird nach dem von Gutzwiller in [Gut94] vorgeschlagenen Verfahren zur Methodenbeschreibung vorgegangen.

Zu Beginn werden daher die Handlungsfelder und Gestaltungsobjekte der Methode im Rahmen eines Metaobjektmodells vorgestellt. Diese geben einen Überblick, in welchen Beziehungen die resultierenden Ergebnisse zueinander stehen.

Das Vorgehensmodell umfasst die separate Entwicklung der Anwendung und der Plattform. Da die PAS allerdings nur auf die einmalige Entwicklung einzelner E-Learning-Komponenten ausgerichtet ist und keine Meta-Prozesse wie die gemeinsame Entwicklung hochschulweiter E-Learning-Konzepte oder den Aufbau einer Standardarchitektur umfasst, werden im zweiten Abschnitt (5.3.3) diesbzgl. notwendige Prozesse identifiziert und in das Modell aufgenommen.

Die erweiterte Prozessarchitektur wird im Anhang A.5 ausführlich beschrieben. Dieses Beschreibungsmodell impliziert die Zuordnung von Rollen, Ergebnissen und Methoden/-Techniken zu Prozessen. Daher wird in diesem Kapitel auf eine detailliertere Darstellung des Ergebnisses verzichtet.

Zuletzt wird jeweils ein Konzept zur Prozesssteuerung sowohl für die Plattformentwicklung als auch für die (5.3.4) auf Grundlage von Erfahrungen vorgeschlagen.

5.3.1. Das Metaobjektmodell

Ein Metaobjektmodell ist ein konzeptionelles Datenmodell, das einen schnellen Überblick über die beschreibenden und gestaltenden Bereiche der Methode liefert. Das in Abb. 5.6 dargestellte Modell basiert dabei auf den in den Kapiteln 4 und 5 vorgestellten Konzepten für die normsprachliche Spezifizierung und für die Entwicklung der Architektur sowie ihren Beziehungen zueinander. Es ist in UML notiert. Aus Gründen der Übersichtlichkeit wurde auf die Angabe der Kardinalitäten verzichtet, da sie für die gewählte Betrachtungsebene keine wesentlichen Aussagen liefern.

Im Metaobjektmodell repräsentieren verschiedene Gestaltungsobjekte die drei Handlungsfelder der Methode:

Normsprachliche Spezifikation Basierend auf einem pädagogisch-didaktischen Modell, das in der Methode analysiert wird und daher nicht selbst zum konzeptionellen Teil gezählt werden kann, werden die statischen, funktionalen und dynamischen Gestaltungsaspekte einer Lernumgebung spezifiziert. Hierzu werden die normsprachliche Konstrukte der in Kapitel 4 erarbeiteten Grammatik und Terminologie verwendet (in der Abbildung invers gekennzeichnet).

Entwurf Unter Zuhilfenahme der normsprachlichen Spezifikation erfolgt in einem zweiten Schritt der technologiespezifische Entwurf der Lernumgebung. Dieser basiert – wie in Kapitel 5 beschrieben – grundlegend auf der Produktstandardarchitektur,

die wiederverwendbare Kooperations- und E-Learning-Dienste über Komponenten bereitstellt. Diese Architektur wird naturgemäß durch das Umsystem eines Entwicklungsprojektes beeinflusst. Zum einen durch hochschulweite IT-, Medien- und E-Learning-Strategien, zum anderen durch Referenz- und Rahmenarchitekturen wie in Abschnitt 2.2.4.3 vorgestellt respektive in Kapitel 5 erarbeitet. Im Entwurf wird eine passende Auswahl wiederverwendbarer Komponenten getroffen, deren Variationspunkte entsprechend den Anforderungen konfiguriert und um anwendungsspezifische Komponenten ergänzt. In Gesamtheit bildet dies die Produktarchitektur. Darüber hinaus umfasst die Methode im Rahmen einer proaktiven Wiederverwendung den separaten Entwurf der Produktstandardarchitektur.

Entwicklung In Analogie zum Entwurf erfolgt die Entwicklung von Plattform und Anwendung separat. Ein Teil der wiederverwendbaren Dienste wird dabei über ein Groupware-Framework abgedeckt. Zur Entwicklung einer Anwendung werden anwendungsspezifische Dienste implementiert und integriert. Die Entwicklung beinhaltet daher auch Integrationstests.

Die Methode selbst umfasst darüber hinaus Querschnittsfunktionen wie Projekt-, Risiko- und Qualitätsmanagement, die zugunsten einer besseren Übersichtlichkeit nicht als Handlungsfelder im Metaobjektmodell mit aufgenommen wurden.

5.3.2. Vorgehen zur Anwendungsentwicklung

Wie bereits in der Vorstellung der PAS erwähnt, muss das generische Referenzmodell zur Verwendung auf einen bestimmten Anwendungskontext zugeschnitten werden. Dazu wird im Folgenden zunächst die Anwendungssituation konkretisiert, die die in dieser Arbeit vorgestellte normprachliche Spezifizierung und Produktlinien-orientierte Umsetzung von E-Learning-Komponenten im Kontext universitärer Informationsarchitekturen umfasst. Davon werden auch die Rollen abgeleitet, die im Beschreibungsmodell im Anhang verwendet werden. In einem zweiten Schritt werden diejenigen Prozesse der PAS identifiziert, die für diese Anwendungssituation nicht relevant sind, oder die hierfür abgeändert oder anderen Prozessstrukturen untergeordnet werden. Die verbleibende Prozessauswahl wird in einem dritten Schritt dann spezifiziert und um neue Prozesse ergänzt.

Zur besseren Unterscheidung zwischen den Prozessen der PAS und denen der zu konstruierenden Methode wird den Nummern der PAS-Prozessen ein P vorangestellt.

5.3.2.1. Spezifizierung der Anwendungssituation

Das zu beschreibende Vorgehen ist primär geprägt durch die Merkmale des zweigeteilten normsprachlichen Fachentwurfs, durch die separate Betrachtung von Plattform und Anwendung im Softwareentwicklungsprozess und letztendlich auch durch die Komponentenorientierung, die bei Entwurf und Implementierung besonders berücksichtigt werden muss. Über allem steht jedoch die Anforderung, dass die zu entwickelnde E-Learning-Komponente zu einem Bestandteil der hochschulweiten Infrastruktur und daher auch – losgelöst von der fachlichen bzw. inhaltlichen Ausgestaltung – sich am hochschulweiten E-Learning-Verständnis orientieren sollte.

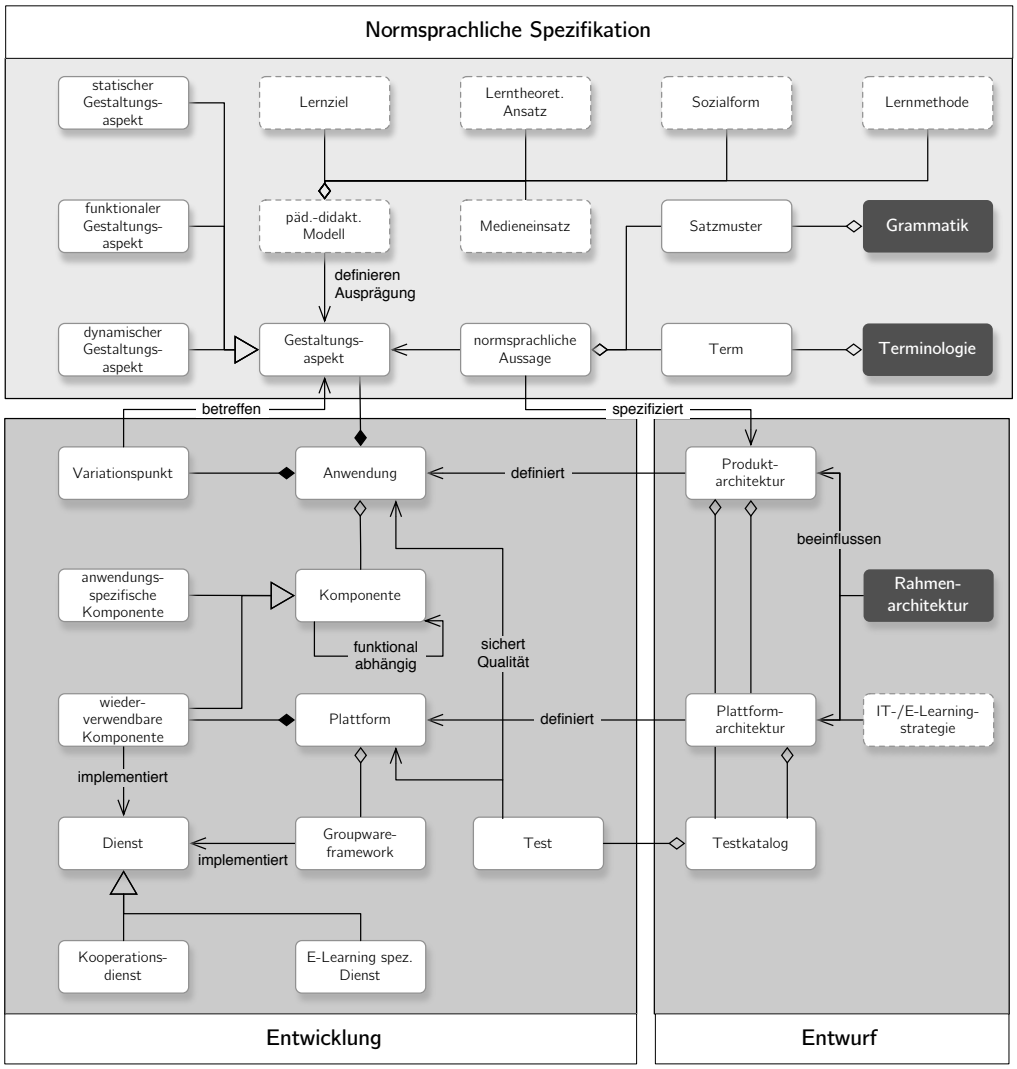

Abbildung 5.6.: Das Metaobjektmodell zum Überblick über die beschreibenden und gestaltenden Bereiche der Methode

Aus diesem Grund erscheint es sinnvoll, die Anwendungsentwicklungen zusammen mit einem zentralen technischen Dienstleister wie ein Medienzentrum, E-Learning-Center, Rechenzentrum o. ä. durchzuführen, das sowohl mit der E-Learning- als auch mit der IT-Strategie der Hochschule vertraut ist und darüber hinaus die Entwicklung der Produktstandardplattform verantwortet. Aus softwarearchitektonischer Sicht steht daher der Blick auf die gesamtuniversitäre IT-Infrastruktur im Mittelpunkt, von dem ausgehend neue Entwicklungsprojekte, Schnittstellenintegration und der Aufbau von Infrastrukturen und Diensten durchgeführt werden können.

Ebenfalls muss die Domäne des E-Learnings gemeinsam mit den Anwendern aus unterschiedlichen Fachbereichen konzeptuell erschlossen werden. Das impliziert zum einen die Bildung von Begriffen und Beschreibung von Konzepten[29], zum anderen auch die Erarbeitung gemeinsamer Qualitätskriterien, an die sich die Entwicklung und Durchführung von E-Learning-Projekten richten können. Dieses ist ein stetiger Prozess, weshalb die Einrichtung eines festen Arbeitskreises sinnvoll sein kann, mit E-Learning-Experten und Softwareentwicklern auf der einen und Fachbereichsvertretern und Didaktiker auf der anderen Seite. Eine hochschulweite IT-/E-Learning-Strategie, welche die o. g. Aspekte berücksichtigt, kann hierzu den mittel- und langfristigen Handlungsrahmen definieren.

In einer solchen Konstellation, die bereits an einigen Hochschulen aufgebaut wurde bzw. sich im Aufbau befindet, können gemeinsam mit Fachbereichen und einzelnen Lehrstühlen Projekte durchgeführt und neue E-Learning-Komponenten entwickelt und in die Infrastruktur integriert werden.

5.3.2.2. Anpassung und Erweiterung des Referenzmodells

Die spezifizierte Anwendungssituation beschreibt die Entwicklung von E-Learning-Werkzeugen auf Basis einer Standardarchitektur, die für den hochschulweiten Betrieb bestimmt sind. Die Ausführungen zu Vorgehensmodellen haben bereits deutlich gemacht, dass der Entwurf und die Entwicklung eine hohe Komplexität aufweisen, die im Rahmen eines Projektes gehandhabt werden muss. Insbesondere wenn es sich um eine hochschulweite Lösung mit hohen Nutzerzahlen oder eine organisationskritische Anwendung handelt, wie beispielsweise eine Kursanmeldung oder ein Kommunikationskanal, ist ein solches Projekt oft mit Risiken behaftet und es muss eine bestimmte Qualität sichergestellt werden. Neben dem Risikomanagement und der Qualitätssicherung zählt auch die Organisation des Projektes selbst zu den Querschnittsfunktionen, die umso wichtiger wird, je größer das Projekt ist.

Projektvorbereitung Das Referenzmodell berücksichtigt die Qualitätssicherung zwar implizit im Beschreibungsmodell[30], im Prozessmodell der PAS ist die Qualitätssicherung jedoch nicht als Prozess explizit mit aufgeführt. Gleiches gilt für das Risikoma-

[29] Hierzu bietet die in Kapitel 4 beschriebene normsprachliche Rekonstruktion auf Basis einer Domänenterminologie eine geeignete Grundlage.

[30] Beispielsweise müssen die Kriterien von ISO 9241 für Softwareprodukte erfüllt sein. Darüber hinaus gibt es eine Reihe von lern- und medienpsychologischen Qualitätskriterien in sieben Kriterienbereichen, die auf die spezielle Zielsetzung des Lernen bezogen sind.

nagement und die Projektorganisation[31]. Aus diesem Grund wird eine erste Phase „Projektvorbereitung“ definiert, in der neben den zusätzlich aufgenommenen Management-Prozesse 2.1.2, 2.1.8, 2.1.10, 2.1.11 auch Prozesse der PAS-Kategorie „Rahmenbedingungen“ zugeordnet werden. Dabei geht der Prozess „P2.2 Analyse der personellen Ressourcen“ zum Teil in „2.1.2 Festlegung von Rollen und Verantwortlichkeiten“ und in „2.1.9 Projektplanung“ mit ein und ist im Modell in ursprünglicher Form nicht mehr vorhanden. Ebenfalls ist die „P2.4 Analyse des organisatorischen und institutionellen Kontextes“ zusammen mit „P2.6 Analyse der Ausstattung“ in den Prozess „2.1.7 Rahmenbedingungen ermitteln“ eingegangen. Diese beiden Prozesse sind nicht mehr explizit genannt, die ihnen zugeordneten Tätigkeiten sind aber im Beschreibungsmodell unter 2.1.7 vorhanden.

Analyse Aufgrund der gewollten didaktischen Neutralität des Modells und der im Gegenzug dazu starken technischen Bindung an die komponenten- und Produktlinien-orientierte Entwicklung verlagert sich der Schwerpunkt im neuen Vorgehensmodell im Entwurf und der Umsetzung auf die softwaretechnischen Aspekte. Um die didaktischen Aspekte der Konzeption trotzdem nicht zu vernachlässigen, wird die ursprüngliche mediendidaktische Konzeption (P3.3, P3.7, P3.8, P3.9,) in die Analysephase mit aufgenommen (2.2.1, 2.2.2, 2.2.3) und geht damit also in die detaillierte Anforderungserhebung mit ein. Die Analyse der Zielgruppe erfolgt bereits in der Ermittlung der Rahmenbedingungen (2.1.7). Der Prozess „Konzeption der Tests und Prüfungen“ (P3.10) wird im neuen Modell aufgrund der geänderten Schwerpunktsetzung jedoch nicht mit berücksichtigt.

Darüber hinaus wird im neuen Modell ein weiterer Prozess mit aufgenommen: Zusammen mit den Stakeholdern wird die Statik, die Funktionalität und die Dynamik der zu unterstützenden Lern- und Arbeitsszenarien normsprachlich modelliert (2.2.4). Dazu sind dem Anwendungsfall entsprechend Gruppen-/Nutzerstrukturen, Inhaltestrukturen und Prozessstrukturen abzubilden. Funktionalität und Dynamik definieren, welche Freiheitsgrade auf den drei Dimensionen Teams, Inhalte und Prozesse unterstützt werden sollen.

Konzeption Die Konzeptionsphase des neuen Modells beinhaltet auf der normsprachlichen Spezifizierung aufbauend zum einen den Entwurf der Produktarchitektur, zum anderen die Erstellung eines organisatorischen Konzepts zur Systemeinführung (2.3.4), Betrieb und Support (2.3.5) sowie zur Wartung und Pflege (2.3.6). Der PAS-Prozess „Organisatorische Konzeption“ (P3.5) wurde dazu verfeinert und ist in seiner ursprünglichen Form nicht mehr im Modell enthalten. Dies trägt dem erhöhten Abstimmungsbedarf Rechnung, der in der Regel bei hochschulweit betriebenen E-Learning-Werkzeugen notwendig ist.

[31] Lediglich die „Termin-/Budgetplanung“ wird hierzu angeführt. Auf die Strukturierung des Projektes sowie auf die Festlegung von Rollen und Verantwortungen wird jedoch nicht näher eingegangen.

Der Entwurf der Produktarchitektur orientiert sich am komponenten- und Produktlinien-orientierten Vorgehen. Daher werden drei neue Prozesse eingeführt:

2.3.1 Herleitung der Produktarchitektur Instanzierung der Produktstandardarchitektur inkl. Beschreibung der Konfiguration und der produktspezifischen Anpassungen und Erweiterungen

2.3.2 Identifizierung relevanter Komponenten Abgleich der erhobenen Anforderungen mit den Feature-Modellen und Konfigurationsmöglichkeiten der vorhandenen Komponenten

2.3.3 Entwurf produktspezifischer Komponenten Entwurf produktspezifischer Funktionen als Komponenten auf Basis der erhobenen Anforderungen.

Diese drei neuen Prozesse verfeinern den PAS-Prozess „Technische Konzeption“ (P3.6), der daher in seiner generischen Form nicht übernommen wird.

Implementierung, Integration, Test In Analogie zur Konzeptionsphase wird auch hier der generische PAS-Prozess „Technische Umsetzung“ (P4.4) durch neue Prozesse „Umsetzung der Produktarchitektur“ (2.4.1), „Instanzierung und Modifikation von Komponenten“ (2.4.2) und „Implementierung produktspezifischer Komponenten“ (2.4.3) konkretisiert. Der neue Prozess „Umsetzung des Medien- und Interaktionsdesigns“ (2.4.4) subsumiert die PAS-Prozesse „Designumsetzung“ (P4.2) und „Medienrealisation“ (P4.3). Der Grund für diese höhere Abstraktion ist der, dass die Content-Entwicklung kein expliziter Bestandteil des Modells sein soll. Das Modell soll nicht nur die Erstellung klassischer Hypermedia-Umgebungen und Web based Trainings unterstützen, sondern auch konstruktivistischere, lernerzentrierte Werkzeuge. Eine „Inhaltliche Realisation“ (P4.1) ist daher nicht per se relevant, weshalb der PAS-Prozess im neuen Modell auch nicht weiter berücksichtigt wird. In einem konkreten Projekt kann dieser Prozess bei Bedarf problemlos mit eingebunden werden.

Die Implementierungsphase schließt mit einer aufeinander abgestimmten Reihe von einzelnen Tests, um alle voneinander abhängigen Komponenten im Zusammenspiel miteinander zu testen („Integrationstest“, 2.4.5). Hierbei sollen (a) die Benutzungsschnittstelle, (b) die Komponentenschnittstellen der Produktstandardarchitektur im Zusammenspiel mit neu entwickelten Komponenten und (c) die Kommunikationsschnittstelle mit dem Basissystem getestet werden. In Summe mit zwei weiteren Prozessen „Systemtest“ (2.5.2) und „Abnahme oder Übernahmetest“ (2.5.5) aus der Systemeinführungs- und Abnahmephase konkretisiert der Integrationstest den PAS-Prozess „Test der Lernressourcen“ (P5.1). Auch dies trägt dem erhöhten Bedarf eines reibungslosen Ablaufs der Einführung und des Betriebs einer hochschulweiten, evtl. sogar organisationskritischen Anwendung Rechnung.

Systemeinführung und Abnahme Die „Einrichtung der technischen Infrastruktur“ (2.5.1, angelehnt an den PAS-Prozess P5.5) wurde im Beschreibungsmodell insbesondere um drei Aspekte konkretisiert:

- ➢ Sicherstellung von Infrastrukturkomponenten, die den Anforderungsspezifika entsprechen
- ➢ Maßnahmen zur Ausfallsicherung, sicherem Zugriff und sicheren Transaktionen
- ➢ Bereitstellung von Dokumentationen und Anleitungen für den Betrieb, den Zugriff und die Transaktionssicherheit.

Danach erfolgt der bereits erwähnte Systemtest, und nach Anpassung und Bereitstellung schließlich der Abnahme- oder Übernahmetest. Die Phase wird durch die Sicherstellung des organisatorischen Betriebs (2.5.6) in Anlehnung an die in der Konzeptionsphase entwickelte Aufbau- und Ablauforganisation (2.3.5) abgeschlossen. Abhängig vom „Konzept der Systemeinführung“ (2.3.4) können nun die im Projekt entwickelten Komponenten erst einmal einer begrenzten Nutzergruppe oder sogar schon hochschulweit freigegeben werden.

Betrieb Für die Betriebsphase wurde aus der PAS alleinig der Prozess „Administration“ (P6.1) übernommen. Die im Referenzmodell enthaltenen Lern-, Unterstützungs- und Transferaktivitäten (P6.2) sowie prüfungsrelevante Tätigkeiten (P6.3) sind für den spezifizierten Anwendungskontext nicht relevant. Statt dessen wurden zwei neue Prozesse „Support“ (2.6.2) und „Wartung und Weiterentwicklung“ (2.6.3, angelehnt an den PAS-Prozess P4.5) definiert.

Evaluation In der Evaluationsphase soll eine systematische Untersuchung der Verwendbarkeit bzw. Güte der entwickelten E-Learning-Komponente durchgeführt werden. Dazu kann der Evaluations-Regelkreis der PAS – bestehend aus Planung (P7.1), Durchführung (P7.2), Auswertung der Evaluation (P7.3) und die darauf aufbauende Verbesserung von Produkten und Prozessen (P7.4) – ohne größere Modifikation übernommen werden.

5.3.3. Vorgehen zur Entwicklung der Standardplattform

Die gemeinsame Erschließung von Konzepten mit E-Learning-Experten und Softwareentwicklern auf der einen und Fachbereichsvertretern und Didaktikern auf der anderen Seite ist essenziell für die Bildung eines gemeinsamen Fachvokabulars. Nur somit können fachbereichsübergreifende Anforderungen aufgenommen, softwaretechnisch umgesetzt und als Lösung hochschulweit angeboten werden. Neben der reinen Anwendungsentwicklung soll das zu konstruierende Modell daher auch die hochschulweite Domänenentwicklung des E-Learnings unterstützen, sowohl aus konzeptioneller als auch aus technischer Sicht. Letzteres meint die Entwicklung der Standardplattform.

Die von dem PAS-Referenzmodell abgeleitete Prozessarchitektur, das bislang nur die Durchführung eines Entwicklungsprojektes umfasst, wird daher um drei Prozesskategorien erweitert: „Analyse der E-Learning-Domäne“ (1.1), „Entwurf der Standardplattform“ (1.2) und „Entwicklung der Standardplattform“ (1.3). Da diese Prozesse nicht in der PAS enthalten sind, sollen sie an dieser Stelle kurz erklärt werden. Darüber hinaus werden auch sie im Beschreibungsmodell im Anhang detailliert dargestellt.

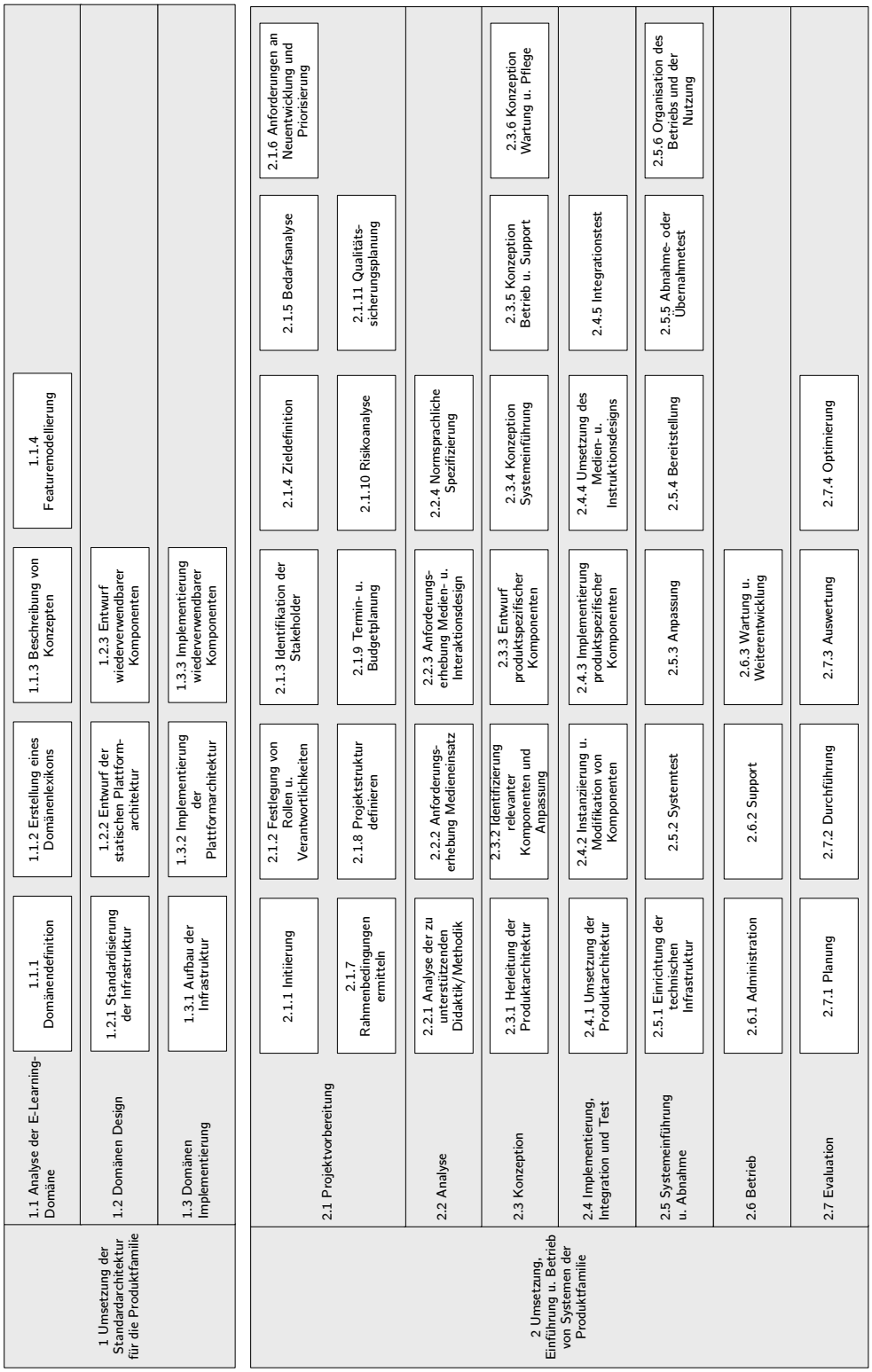

Abbildung 5.7.: Prozessarchitektur der konzipierten Methode

5.3.3.1. Analyse der E-Learning-Domäne

Die dieser Kategorie zugeordneten Prozesse haben in ihrer Kombination zweierlei Funktion: Zum einen die mittel- und langfristige Bildung von Konzepten, deren Intention allen Stakeholdern an der Universität bekannt ist. Zum anderen die Definition des Produktraums, der durch Eigenentwicklungen auf Basis der Standardplattform umgesetzt werden kann. Hierzu werden in der Prozessarchitektur vier neue Prozesse definiert:

1.1.1 Domänendefinition Beschreibung des Umfangs der zu betrachtenden E-Learning-Domäne und Charakterisierung des Produktraums von zu betrachtenden E-Learning-Werkzeugen[32]. Hierbei kann eine hochschulweite E-Learning-Strategie einen Rahmen vorgeben.

1.1.2 Erstellung eines Lexikons Beschreibung der E-Learning-Begriffe, die hochschulweit zur Strategie-, Konzept- und Anwendungsentwicklung verwendet werden sollen. Ziel dabei ist die Sicherstellung einer gemeinsamen Kommunikationsbasis. Hierzu kann auch ein Normsprachenlexikon verwendet werden, in dem neue Fachbegriffe durch eine bereits definierte Terminologie rekonstruiert werden (vgl. hierzu Abschnitt 3.3.1).

1.1.3 Beschreibung von Konzepten Zur Sicherstellung eines gemeinsamen Verständnisses der Statik, Funktionalität und Dynamik von E-Learning-Szenarien muss deren Beschreibung mit Hilfe einer angemessenen Modellierung, grafisch oder normsprachlich erfolgen.

1.1.4 Featuremodellierung Auf Basis der Modelle der E-Learning-Domäne kann der betrachtete Produktraum bzgl. Featurevariationen, -kombinationen und -konflikten analysiert und dokumentiert werden. Dies macht Abhängigkeiten zwischen Funktionen transparent und liefert eine geeignete Entscheidungsbasis für die Systemabgrenzung von Produktplattform und darauf basierender Anwendung. Hierzu können Hilfsmittel wie das in Abb. 4.27 dargestellte Featuremodell eingesetzt werden.

Wie bereits einleitend erwähnt, wird dieser Teil des Vorgehens am Besten im hochschulweiten Rahmen eines Arbeitskreises durchgeführt, der neben E-Learning-Experten, Software-Entwicklern und Didaktikern aus Vertretern der verschiedenen Fachbereiche besteht.

5.3.3.2. Entwurf der Standardplattform

Auf Basis der erarbeiteten gemeinsamen Modelle sowie der Systemabgrenzung von Standardplattform und darauf basierenden Anwendungen kann eine Standardisierung der Infrastruktur vorgenommen und die Produktstandard-Architektur und ihre Komponenten entworfen werden. An dieser Phase sind vorrangig SW-Architekten, -Entwickler und Systemadministratoren beteiligt. Sie gliedert sich wie folgt:

[32] Beispielsweise ob sich die universitätsinterne Softwareentwicklung vorrangig auf die Unterstützung materialorientierte, kooperative Szenarien beschränkt, oder auch fachspezifische Simulationen und virtuelle Labore umfasst.

1.2.1 Standardisierung der Infrastruktur Bezüglich der Entwicklungsumgebung müssen zwischen Architekten, Entwicklern und Administratoren Vereinbarungen über die Entwicklungs- und Laufzeitumgebung getroffen werden. Dazu gehört auch eine Auswahl von SWE-Pattern und Frameworks sowie die Erstellung einer Architektur-/ Technologiestrategie[33] und Programmierrichtlinien. Hierzu kann die in dieser Arbeit konzipierte Rahmenarchitektur herangezogen werden.

1.2.2 Entwurf der Produktstandardarchitektur Zur Vorbereitung arbeitsteiligen Entwickelns müssen die Kernfunktionen des funktionsübergreifenden Rahmenwerks entworfen werden. Darüber hinaus müssen existierende Lösungen eingebunden und funktionale Erweiterungen in Form zusätzlicher Komponenten leicht integriert werden können. Dazu wird ein PlugIn-Mechanismus benötigt, der ebenfalls zu entwerfen ist.

1.2.3 Entwurf wiederverwendbarer Komponenten Sind durch die standardisierte Infrastruktur noch nicht alle allgemeinen Dienste und Kooperationsdienste abgedeckt, müssen die fehlenden Dienste entworfen werden. Weiterhin müssen auf Grundlage allgemeiner Dienste und Kooperationsdienste die E-Learning-spezifischen Dienste konzipiert werden.

Ergebnis dieser Phase sind die Entwurfsdokumente für die Standardarchitektur, auf denen in der nächsten Phase die Umsetzung der Standardplattform basiert.

5.3.3.3. Entwicklung der Standardplattform

Ziel dieser Phase ist die Bereitstellung einer standardisierten Infrastruktur und darauf aufbauender Produktstandardarchitektur. Die Phase gliedert sich in den Aufbau der Infrastruktur, in die Implementierung der Standardarchitektur und wiederverwendbarer Komponenten.

1.3.1 Aufbau der Infrastruktur Installation und Integration verschiedenster Technologien wie Infrastrukturkomponenten, IDEs, Frameworks, Schnittstellen und Protokolle, um auf dieser Basis eine lauffähige Entwicklungs- und Laufzeitumgebung für die Plattform und ihre Komponenten bereitzustellen.

1.3.2 Implementierung der Standardarchitektur Umsetzung der Kernfunktionen der entworfenen Standardplattform sowie die Umsetzung eines PlugIn-Mechanismus zur Integration von erweiternden Komponenten.

1.3.3 Implementierung wiederverwendbarer Dienste Implementierung allgemeiner und E-Learning-spezifischer Funktionen, die in den verschiedenen, darauf basierenden Anwendungen wieder verwendet werden können.

[33] Eine Architektur-/Technologiestrategie sollte in diesem Fall folgende Spezifikationen umfassen: Basisinfrastruktur für Schichten und Vermittlungsschichten, genormte Schnittstellen zu den genutzten externen Systemen, OO-Repository und Groupware-Framework, Programmiersprache(n) und Application Server, Funktionsbibliotheken.

Das Ergebnis dieser Phase ist eine funktionierende Entwicklungs- und Laufzeitinfrastruktur, sowie eine funktionierende Standardplattform inklusive allgemeiner und domänenspezifischer Dienste.

5.3.4. Empfehlung einer Ablaufreihenfolge

Im letzten Abschnitt wurde die Prozessarchitektur durch Auswahl, Anpassung und Erweiterung der Prozesse aus der DIN PAS 1032-1:2004 aufgebaut (vgl. Abb. 5.7). Darüber hinaus wurden inhaltlich und logisch zusammenhängende Prozesse bereits zu Phasen zugeordnet, zu deren Abschluss die im Beschreibungsmodell genannten Ergebnisse (Meilensteine) vorliegen müssen. Dabei bilden die Plattform- und Anwendungsentwicklung zwei separate Iterationszyklen (vgl. hierzu auch S. 170ff.). Wie in Abbildung 5.8 dargestellt ist, bauen die Phasen logisch aufeinander auf, wobei die Ergebnisse der Plattformentwicklung in korrespondierende Phasen der Anwendungsentwicklung eingehen[34]. Um die Struktur und den Ablauf eines realen Projektes anhand dieser statischen Prozessarchitektur planen zu können, soll im Folgenden ein Prinzip der Ablaufreihenfolge zur Prozesssteuerung empfohlen werden.

Abschnitt 5.1.4 liefert dazu bereits eine Entscheidungsgrundlage für die Softwareentwicklung im universitären Kontext, die ein iterativ-inkrementelles Vorgehen nahe legt, allerdings auch die Nachteile aufzeigt.

In der mehrjährigen Plattform- und Anwendungsentwicklung an der Universität Paderborn[35] haben sich ein prototypisches Vorgehen für die Plattformentwicklung und ein iterativ-inkrementelles Vorgehen für die Anwendungsentwicklung bewährt.

Entwurf und Entwicklung von Komponenten der Plattform werden proaktiv durch Studierende im Rahmen von Projektseminaren und Abschlussarbeiten betrieben und sind in der Regel nicht zeitkritisch. Durch experimentelle Prototypen können schwierige Entwurfsentscheidungen überprüft werden. Daneben dienen explorative Prototypen der besseren Darstellung neuer Plattformfunktionen und liefern Anregungen, Anforderungen und Verbesserungsvorschläge seitens der Anwender. Aus dem gleichen Grund sind explorative Prototypen auch für die Forschung im Bereich CSCW/L von Interesse. An fertiggestellten Komponenten werden Integrations- und Funktionstests durch erfahrene wissenschaftliche Mitarbeiter vollzogen, bevor sie in die Plattform integriert werden.

Die anwendungsspezifische Konfiguration und Komponentenentwicklung erfolgen hingegen iterativ-inkrementell, da die Fertigstellung der wichtigsten Funktionen zumeist zeitkritisch sind[36]. Dazu ist in der Projektvorbereitungsphase der Prozess „Anforderungen an Neuentwicklung (grob) und Priorisierung“ (2.1.6) vorgesehen, in welchem wichtige Anforderungen dokumentiert und priorisiert werden sollen, damit sie als Basis für die Planung der Inkremente respektive Iterationen dienen können. Die Iteration orientiert sich dabei an den jeweils definierten Inkrementen und umfasst die Phasen Analyse (2.2), Konzeption (2.3) und Anpassung, Integration, Test (2.4).

[34] Der in Abb. 4.26 dargestellte Rückfluss von Ergebnissen der Anwendungsentwicklung in die Plattformentwicklung wurde bei dieser Abbildung zur besseren Übersichtlichkeit ausgelassen, ist aber ebenso vorhanden.

[35] Hierauf wird im folgenden Kapitel näher eingegangen.

[36] In der Regel müssen die wichtigsten Funktionen vor dem Beginn eines neuen Semesters fertiggestellt sein.

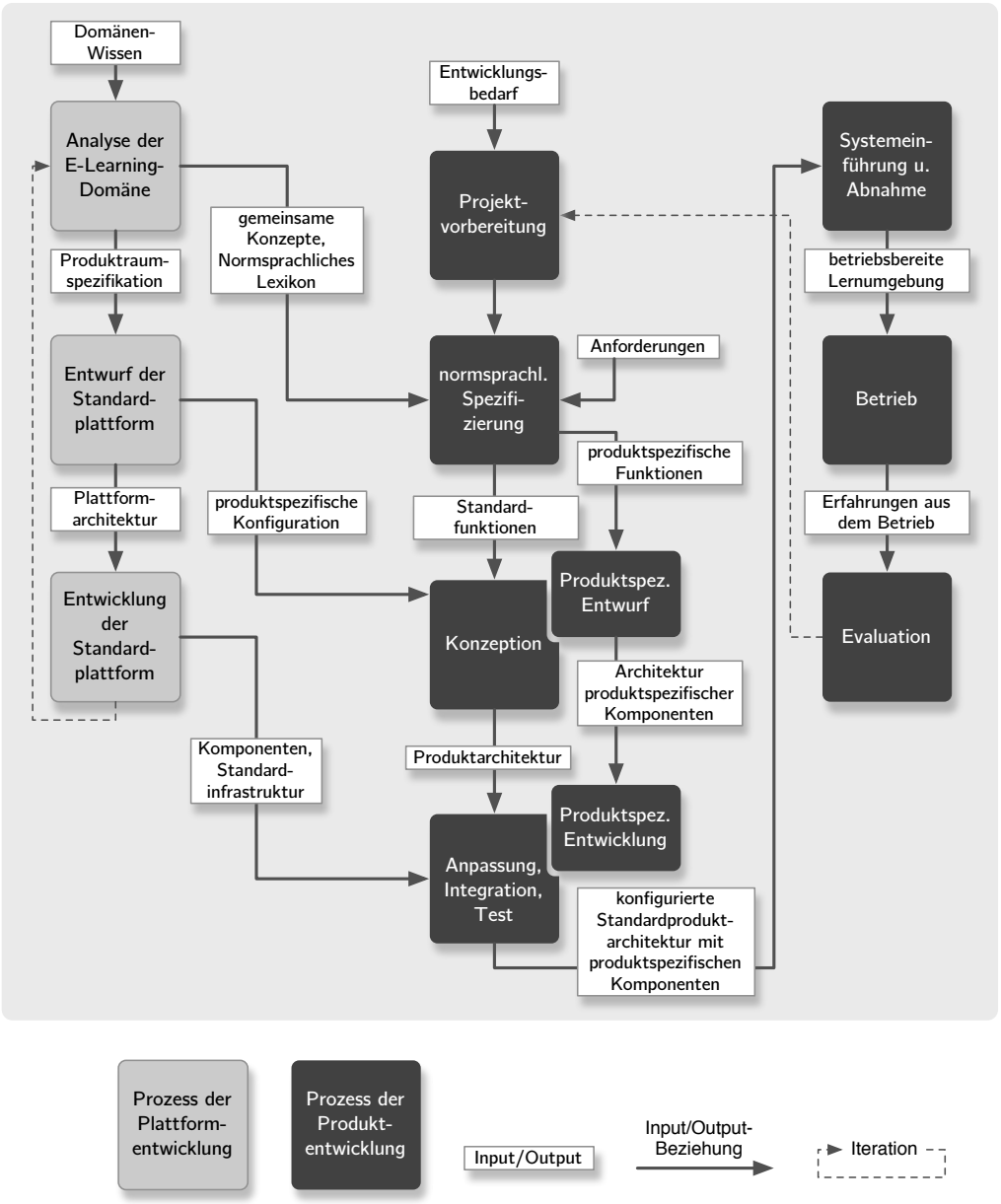

Abbildung 5.8.: Beziehungen zwischen den Phasen der konzipierten Methode

5.4. Zusammenfassung

In diesem Kapitel wurde eine Methode zum komponentenorientierten Entwurf und Entwicklung von Lernumgebungen konzipiert, welche die normsprachliche Spezifikation beinhaltet und per se auf die Umsetzung mehrerer Systeme ausgerichtet ist. Dazu wurden zunächst die theoretischen Grundlagen von Vorgehensmodellen untersucht und Gestaltungsobjekte für die Methodenkonzeption identifiziert. Eine daran anschließende Betrachtung bereits existierender Vorgehensmodelle zur Umsetzung von Lernumgebungen hat das Referenzvorgehensmodell DIN PAS 1032-1:2004 als gute Ausgangsbasis ausgewiesen. Darauf aufbauend wurden relevante Prozesse ausgewählt, angepasst und um darüber hinaus benötigte Prozesse ergänzt. Das Resultat ist eine Methode, bestehend aus

- einer Prozessarchitektur, welche in Summe 49 Prozesse umfasst, die auf 10 aufeinander aufbauende Phasen verteilt sind,
- einem Beschreibungsmodell (Anhang A.5), das für jeden Prozess Beschreibung, Aspekte, Ziele, Ergebnisse und Qualitätskriterien beinhaltet,
- ein Rollenmodell[37], das die Beteiligung einer Rolle an Prozessen definiert und somit Aufschluss über Verantwortung und benötigte Kompetenz in konkreten Projekten geben kann,
- eine Techniksammlung[38] für jeden Prozess und
- ein Metaobjektmodell, das die Ergebnisse der Methode in Beziehung zueinander setzt.

[37] Implizit im Beschreibungsmodell vorhanden.
[38] Analog zum Rollenmodell ebenfalls als Teil des Beschreibungsmodells.

6. koaLA – Integrierte Lern- und Arbeitswelten für die Universität 2.0

Im Rahmen des BMBF-Förderprogramms „Neue Medien in der Bildung“ wurde an der Universität Paderborn das Projekt „Locomotion“[1] durchgeführt. Dieses Projekt bildete den Umsetzungsrahmen für die im Kontext dieser Arbeit entstandene hochschulweite Lern- und Arbeitsplattform koaLA[2]. Die Plattform wurde als Anwendung auf Basis einer Produktstandardarchitektur für web-basierte CSCW/L-Systeme umgesetzt und die spezifizierten Nutzungsszenarien ausnahmslos mithilfe von Konzepten der Wissensraummetapher rekonstruiert. Semantisch zusammenhängende Einheiten des Gesamtsystems sowie Nutzungsszenarien wurden als Komponenten gekapselt und können somit in verschiedenen Einsatzkontexten wieder verwendet werden.

Die koaLA-Plattform soll zur Abrundung der Arbeit im Folgenden vorgestellt werden.

6.1. Die Unterstützung formaler und informeller Lernkontexte

Werkzeuge wie Wikis, Weblogs und Podcasts stellen in Verbindung mit sozialen Netzwerken den Benutzer als Produzent von Inhalten respektive seine kooperativen Tätigkeiten deutlich in den Mittelpunkt. Ein Einfluss, der nicht nur beim Arbeiten, sondern auch beim Lernen weg von vollständig vorgegebenen Strukturen und geschlossenen Systemen hin zu wissens- und individuumzentrierten, offenen Umgebungen führt. An Universitäten können diese Werkzeuge für eine stärker konstruktivistisch geprägte Lehre und für Lehrveranstaltungsformate wie Projektseminare, Jour-Fixe-Konzepte und Planspiele eingesetzt werden, welche die Studierenden aktiv in die Lehre mit einbeziehen (vgl. [HKSE03], [RHS06], [Ale06]). Die großen Potenziale dieser Werkzeuge für die Lehre wurden bereits in Abschnitt 2.3.4 herausgearbeitet.

Daneben besteht aber immer noch Bedarf, sowohl die Elemente der Prüfungsordnungen wie Module, Veranstaltungen, Seminare etc., die den organisatorischen Rahmen darstellen, als auch klassische Lehrformen wie beispielsweise der Frontalunterricht in Massenveranstaltungen mithilfe von Computern zu unterstützen. Hierzu sind Konzepte zu implementieren, in denen Prozessstrukturen und die Organisation von Gruppen und

[1] *Low-cost Multimedia Organisation & Production*, ein prozessorientiertes Vorhaben der Universität Paderborn zur alltagstauglichen Erschließung und nachhaltigen Verankerung digitaler Medien in allen Bereichen ihrer Lehr- und Lernpraxis (Förderkennzeichen: 01 PI 05013). Weitere Informationen zum Locomotion-Projekt unter `http://locomotion.uni-paderborn.de/`. Letzter Zugriff: 31.07.2008.

[2] Die Lern- und Arbeitsplattform koaLA („ko-aktives Lernen und Arbeiten“) ist unter der Adresse `http://koala.uni-paderborn.de/` online verfügbar. Letzter Zugriff: 31.07.2008.

Inhalten stark vorgegeben sind. Diese beiden Extreme, die selbst gewählten Gruppenstrukturen wie Lerngruppen und Arbeitsgemeinschaften sowie Web 2.0-Werkzeuge auf der einen und die durchstrukturierten und fest vorgegebenen Organisationskonzepte des Lehrveranstaltungsmanagements auf der anderen Seite spannen den Spezifikationsraum der zu unterstützenden kooperativen Lern- und Arbeitsszenarien auf. Soll der Lernende im Zentrum einer Lern- und Arbeitsumgebung stehen, muss dieser darüber hinaus seine eigenen Prozesse der Wissensschaffung selbst koordinieren und leicht zwischen verschiedenen individuellen, kooperativen sowie strukturierten und selbstorganisierten Kontexten wechseln können (vgl. Abb. 6.1).

Abbildung 6.1.: Einfügen aus der Zwischenablage in den eigenen Arbeitsraum

Zur Umsetzung der koaLA-Plattform wurden dazu Komponenten einer Architektur miteinander kombiniert, die verschiedene Variationspunkte als Konfigurationsmöglichkeiten anbieten, insbesondere in der Ausgestaltung virtueller Wissensräume und Organisationsformen (vgl. hierzu auch [RH05]).

6.1.1. Konfigurierbare Wissensräume

Materialzentrierte Lernszenarien fokussieren vor allem das (kooperative) Arbeiten mit Informationen in Form von Lernobjekten und Dokumenten. Die didaktische Spannweite ist dabei groß. Sie reicht von der einfachen Rezeption vom Dozenten ins Netz gestellter Informationen bis hin zur selbstorganisierten Projektgruppe. Um auf dieser Skala verschiedene Szenarien unterstützen zu können, wurden sechs grundlegende Variationspunkte von virtuellen Wissensräumen identifiziert:

Zugriffsvorbedingung Die Zugriffsvorbedingung koppelt die Zugriffsrechte an Materialien im Raum an ein Systemereignis. Beispiel: Studierende müssen zuerst ein eigenes

Dokument abgeben, bevor sie auf die Dokumente der ihrer Kommilitonen Zugriff erhalten.

Zeitpunkt Die Konfiguration eines Zeitpunktes umfasst neben dem Zeitpunkt selbst die Konfiguration des Raums vor und nach dem Zeitpunkt. Beispiel: Innerhalb einer Abgabefrist können Dokumente im Raum abgelegt werden, danach wird das dafür notwendige Einfügerecht entzogen.

Kommentierbarkeit Materialien im Raum können objektbezogen von Benutzern kommentiert werden. Dies kann (a) personifiziert oder (b) anonym erlaubt oder (c) ganz untersagt sein. Beispiel: Kommentare können anonym abgegeben werden. Hemmnisse können in Lernszenarien auftreten, in denen Studierende ihre Arbeiten gegenseitig zu bewerten haben. Hierbei kann eine anonyme Kommentarfunktion den Teilnehmern helfen, Kritik zu äußern. Dahingegen können personifizierte Kommentare in didaktischen Szenarien wie der Projektmethode sinnvoll sein.

Darstellung Materialien von Studierenden können anonym oder mit Nennung des Namens dargestellt werden. Beispiel: Der Autor eines Dokuments ist für andere Kursteilnehmer nicht erkennbar, wohl aber für den Dozenten. In Analogie zur anonymen Kommentierbarkeit kann auch hier Hemmnissen entgegengewirkt werden, eigene Lösungen zur allgemeinen Diskussion zu stellen.

Rechte für Eigen- und Fremdmaterial Die Zugriffsrechte auf eigene Dokumente und auf die anderer Kursteilnehmer können separat verwaltet werden. Die Interaktion des Benutzers mit den Materialien wird mit den Rechten für Lesen, Schreiben und Löschen der Dokumente reguliert. Beispiel: Nach Ablauf einer Abgabefrist wird das Löschrecht an der bereits abgegebenen eigenen Lösung entzogen. Leserechte auf Dokumente der Kommilitonen sind sowohl vor als auch nach dem konfigurierten Zeitpunkt nicht gesetzt.

Eine Konfiguration von Variationspunkten beschreibt den internen Zustand einer Komponente. Externe Abhängigkeiten zwischen Komponenten sind auf die abstrakte und einheitliche Semantik der Basisklassen beschränkt, so dass eine lose Kombination von Szenarien ermöglicht wird. Abbildung 6.2 zeigt exemplarisch einen in einem Kursszenario eingebetteten virtuellen Raum, der durch seine Konfiguration verschiedene Ausprägungen in den Variationspunkten annehmen kann. Auf diese Art und Weise unterstützt koaLA in Kursräumen neben den vom Dozenten verwalteten Lektionen auch Projekträume, Möglichkeiten der Übungszettelabgabe, Seminarräume sowie speziellere Werkzeuge wie Wikis, Weblogs, Foren und Pyramidendiskussionen.

Durch die Möglichkeit zur flexiblen Kombination der Szenarien konnten alle Werkzeuge, die in den virtuellen Kursräumen zur Verfügung stehen, auch in Gruppenarbeitsräumen integriert werden. Jede Lerngruppe in koaLA kann somit bspw. ihr eigenes Wiki, Weblog oder Forum betreiben und kann darüber hinaus noch definieren, ob es zur Außendarstellung der Gruppe beitragen, also auch für Nicht-Gruppenmitglieder lesbar sein soll, oder nur für den internen Gebrauch bestimmt ist.

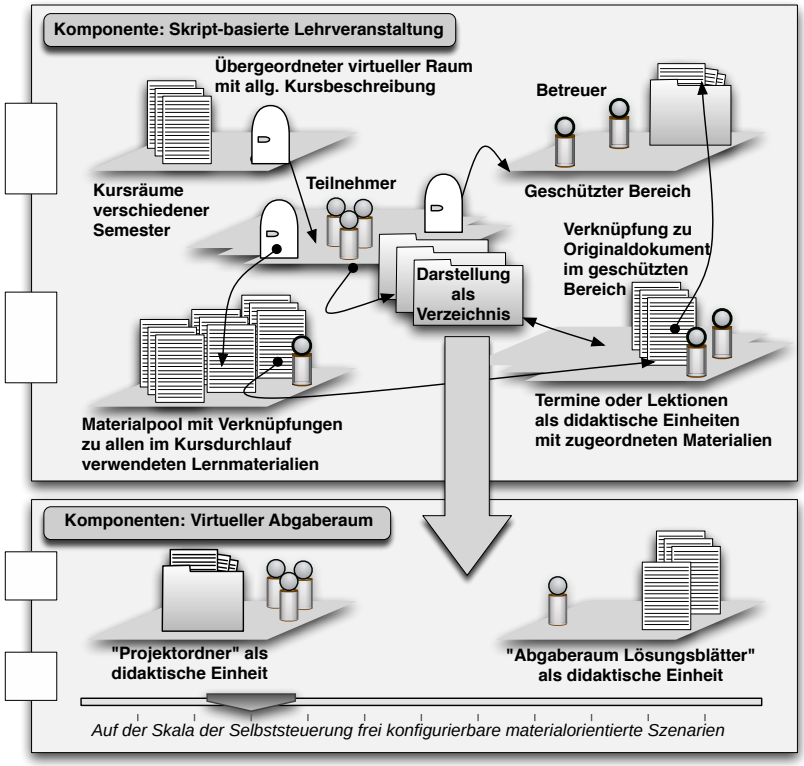

Abbildung 6.2.: Kombination konfigurierbarer Komponenten zur flexiblen Unterstützung von Lernszenarien (aus: [RH05])

6.1.2. Flexible Gruppenstrukturen

Individuelle Arbeitsräume als auch kooperative Arbeitsräume wie Kursräume werden in koaLA an Benutzergruppen gebunden. Somit stellt die Gruppe ein Konzept dar, mit dem Kommunikationsobjekte, Materialien, soziale Strukturen und zusätzliche Funktionen organisiert werden können (vgl. Abb. 6.3). Dabei sind Kursräume eine besondere Ausprägung einer Gruppe, ebenso wie private Gruppen. In Analogie zu den kombinierbaren, raumbasierten Lernszenarien können verschiedene Gruppen über eine Hierarchie miteinander in Beziehung gesetzt werden (vgl. Abb. 6.4). Hierdurch war es möglich, zum einen eine klare Trennung zwischen selbstorganisierten Gruppen und organisatorischen Konzepten wie Semestern, Modulen, Veranstaltungen, Übungsgruppen, Fakultäten und Studiengängen zu definieren, zum anderen aber auch den Wechsel zwischen Gruppen – also zwischen informellen und formalen Lernkontexten – für den Benutzer bestmöglich zu unterstützen.

Abbildung 6.3.: Jede Gruppe verfügt über Weblogs, Foren, Wikis und verschiedenartig organisierte Dateiräume (hier: Lektionen).

So erlaubt es koaLA jedem Benutzer eigene Gruppen anzulegen. Dabei werden anhand der Sichtbarkeit anderen Nutzern gegenüber drei Ausprägungen unterschieden: Öffentliche Gruppen sind für jeden Benutzer sowohl im Gruppenverzeichnis als auch in den Profilen der Teilnehmer sichtbar. Der Zugang zu einer *öffentlichen Gruppe* kann dabei frei oder durch ein Passwort geschützt sein. *Öffentliche Gruppen mit Einladung*

Abbildung 6.4.: Moderierte Übungsgruppen im Rahmen einer Veranstaltung

sind ebenfalls für jeden Nutzer sichtbar, neue Mitglieder können allerdings ausschließlich durch eine Einladung eines Gruppenadministrators[3] aufgenommen werden. Eine *private Gruppe* ist nicht öffentlich sichtbar. Ihre Mitgliedschaft erfolgt exklusiv über die Einladung eines Gruppenadministrators. Unabhängig vom Gruppenformat haben ihre Administratoren erweiterte Möglichkeiten der Rechtesteuerung innerhalb der Gruppe. So können beispielsweise in der Materialverwaltung Bereiche eingerichtet werden, die für anderen Nutzer nicht bzw. explizit schreibbar sind.

Kurse werden in koaLA manuell durch einen Semesterbetreuer[4] oder semi-automatisch durch den Abgleich mit dem Prüfungsverwaltungssystem angelegt (vgl. S. 218). Wie oben bereits erwähnt, sind Kurse nur eine besondere Form von Gruppen, die an einer bestimmten Stelle in der Gruppenhierarchie ausgewiesen werden. Der Mechanismus der Rechtesteuerung verhält sich daher analog zu Gruppen. Durch diese Flexibilität können in der Praxis unterschiedlichste Veranstaltungsformate abgebildet werden: Große Veranstaltungen mit hunderten von Teilnehmenden erfordern oft bedingt durch das didaktische Vorgehen andere Rechtekonfigurationen als kleine Projektseminare mit 10-20 Teilnehmenden.

Jede Gruppe stellt einen Mail-Verteiler dar, der über das interne Nachrichtensystem aber auch mit E-Mail-Clients bedient werden kann.

[3] Zum Gruppenadministrator wird automatisch der Nutzer, der die Gruppe anlegt. Er kann weitere Administratoren für die Gruppe benennen und – sofern mindestens ein weiterer benannt ist – selbst von der Administratorenrolle wieder Abstand nehmen.

[4] Der Semesterbetreuer ist in der Lage, neue Kurse einzurichten und fehlerhaft eingerichtete zu ändern bzw. zu löschen. An der Universität Paderborn übernimmt diese Rolle der Benutzersupport.

6.1.3. Soziale Netzwerke

Jeder Benutzer hat in koaLA eine Profilseite, die er mit Informationen über sich selbst füllen kann[5] (vgl. Abb. 6.5). Neben den Kontaktinformationen wie E-Mail-Adresse, Telefonnummer und *Instant Messaging*-Daten können weitere persönliche Daten zum Studium (Schwerpunkt und Fachbereich) und zu persönlichen Interessen angegeben werden.

Abbildung 6.5.: Profil einer koaLA-Benutzerin

Die Teilnehmenden einer Gruppe (und somit auch Kursteilnehmer) haben über die Funktion *Teilnehmer* Zugriff auf eine Teilnehmerliste. Die Liste sowie die mit Nutzerprofilen verknüpften, für andere sichtbare Aktionen (Forenbeiträge, Kommentare, Materialien, etc.) bilden die Basis für die Wahrnehmung der anderen Teilnehmer und damit für die virtuelle Zusammenarbeit. Der Netzwerkgedanke findet sich in den auf einer Profilseite angezeigten öffentlichen Gruppen und persönlichen Kontakten, an denen der Benutzer teilnimmt bzw. die er bestätigt hat. Über die Funktion „als Kontakt hinzufügen“ können Benutzer untereinander ihr Kontaktnetzwerk ausbauen.

[5]Die Angabe der Daten ist zwar nicht verpflichtend, die Praxis hat aber gezeigt, dass diese Möglichkeit der Selbstdarstellung von etwa der Hälfte der Benutzer freiwillig genutzt wird.

6.2. Die Systemarchitektur

Die ko-aktive Lern- und Arbeitsplattform koaLA wurde als Anwendungsschicht auf dem CSCL-Server open-sTeam umgesetzt, der grundlegende Funktionen zur Kommunikation und Inhalte-Verwaltung über Programmierschnittstellen bereitstellt. Im Kern implementiert der am Heinz Nixdorf Institut Paderborn entwickelte Server das in Abschnitt 3.2.3 vorgestellte Objektmodell virtueller Wissensräume. Der Gedanke von „angepassten Sichten“ auf virtuelle Wissensräume wird selbst bei der Integration standardisierter Internet-Protokolle konsequent fortgesetzt: Gängige Kommunikations- und Infrastrukturprotokolle werden im Server dabei mit den semantischen Objektstrukturen zusammengeführt, wobei verschiedene Adapter raumbasierte Nachrichten oder Aktionen in protokollgerechte Informationen übersetzen (und vice versa). Ein Ereignissystem sorgt für eine Verteilung von Nachrichten an angebundene Clients.

Semantische Objektstrukturen werden über die sTeam-Basisobjekte (vgl. Abb. 3.8) aufgebaut und persistiert. Ein Objekt ist mit einem festem Metadatensatz ausgezeichnet[6], weitere Daten können jederzeit dynamisch über `set`-Methoden attribuiert werden. Das Auslesen der Daten erfolgt analog dazu über `get`-Methoden. Durch diese und wenige weitere Fabrik- und Objektmethoden wie `create` und `delete` ist das Datenmanagement für auf sTeam aufbauende Anwendungen vollständig gekapselt.

Die darauf aufbauenden koaLA-Komponenten sind in der serverseitigen Skriptsprache PHP (Hypertext Preprocessor) entwickelt. Hierzu bietet sTeam eine vollständige Programmierschnittstelle (API). Alle in sTeam verwendeten Basisobjekte und der Großteil an Funktionen und Methoden sind in Form von Proxyobjekten über eine PHP-Klassenbibliothek verfügbar. Hierüber können die fachlichen Objekte der jeweils normsprachlich spezifizierten Nutzungsszenarien komponiert werden, wodurch in der Regel eine äußerst effiziente und kostengünstige Umsetzung der jeweiligen Szenarien gelingt (vgl. [HR05]). Komponenten werden durch ein zentrales Objekt identifiziert und können durch verschiedene fachliche Methoden in der Schnittstelle ausgestaltet werden. Bei der Ausführung des PHP-Codes erfolgt die Kommunikation mit dem sTeam-Server dann über den *Client Object Access Layer* (COAL). Der kooperative Zugriff erfolgt daher zentral über gemeinsame Objekte.

In der Produktarchitektur von koaLA werden mithilfe dieser Komponenten dynamische Lern- und Arbeitskontexte ausgebaut, die zwar oftmals die gleichen fachlichen Funktionen benötigen, sich aber letztendlich durch den Freiheitsgrad der Selbstorganisation voneinander unterscheiden. Die Integration von zentralen Basis- und Komplexdiensten der universitären Informationsarchitektur über offene Service-Schnittstellen erfolgt abhängig ihres Wiederverwendungsgrades zu einem Teil über Serverkomponenten, zum anderen in der Anwendungsschicht (siehe 6.3).

Die Präsentationsschicht der koaLA-Plattform wurde mithilfe von AJAX-Funktionen derart ausgestaltet, dass eine einheitliche und intuitive Bedienung in Verbindung mit einer ansprechenden Oberfläche unterstützt wird[7]. Somit sollen Nutzungsbarrieren ge-

[6] Bspw. Objektname, Beschreibung, Erzeuger, Erstellungs- und Modifizierungsdaten und Objekttyp.

[7] Die Mehrheit der befragten Studierenden (89 % bei n=390) empfindet koaLA als selbsterklärend und nutzt das offizielle Beratungsangebot zur Plattform nur wenig (vgl. [G+08], S. 17ff.).

senkt und die Akzeptanz der Nutzer erhöht werden. Beispielsweise kann der Betreuer einer Lehrveranstaltung die Reihenfolge seiner Lektionen durch das Verschieben (Drag & Drop) einzelner ändern, ohne dass die Seite gespeichert und neu geladen werden muss.

Abbildung 6.6 zeigt eine Gesamtübersicht der Architektur von koaLA.

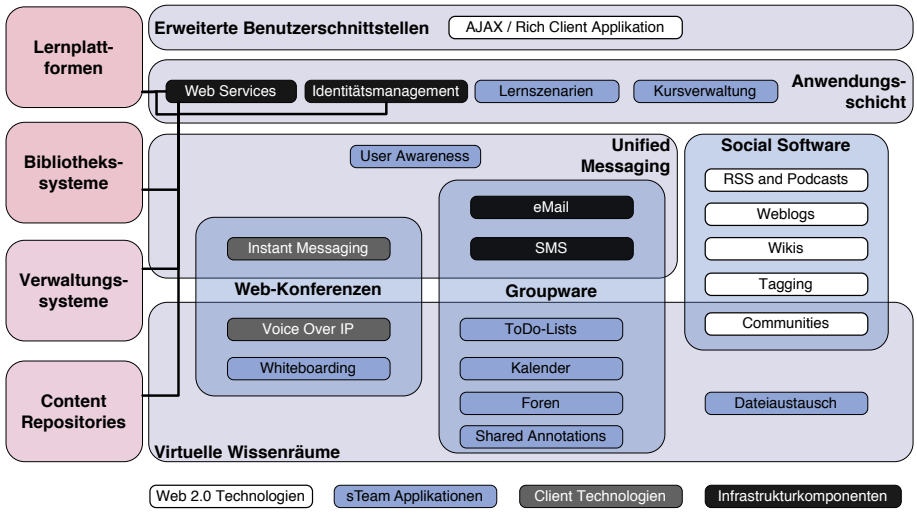

Abbildung 6.6.: Die Web 2.0-Architektur der ko-aktiven Lern- und Arbeitsumgebung koaLA

6.3. Integration in die hochschulweite IT-Infrastruktur

In Kapitel 2 wurde bereits auf die oftmals geringen Anknüpfungspunkte von Lernumgebungen zu den Organisationskontexten der Lernenden hingewiesen (S. 26ff.). Beispiele sind Systeme zur Studienorganisation und Verwaltung, also z. B. Systeme zur Anmeldung und Durchführung von Veranstaltungen, elektronische Seminarapparate oder die (digitale) Bibliothek, die neben den Lernumgebungen die heterogene IT-Landschaft an einer Universität prägen.

In koaLA wurden die in der Rahmenarchitektur beschriebenen serviceorientierten Ansätze genutzt, um für wichtige Prozesse Medienbrüche aufzuheben und Funktionen oder Informationsobjekte anderer Systeme zu integrieren. In einem ersten Schritt wurde ein einheitlicher Zugang zu den beteiligten Systemklassen über den universitätsweiten Authentifizierungsdienst hergestellt. Hochschulangehörige brauchen also keinen separaten neuen Zugang anlegen, um koaLA nutzen zu können. Darauf aufbauend wurde der im Locomotion-Projekt ebenfalls eingeführte elektronische Seminarapparat der hiesigen Bibliothek angebunden. Damit sind die Informationen zu Büchern und digitalen Objekte des elektronischen Seminarapparates direkt innerhalb der Kursräume von allen Teilneh-

mern des Kurses im Zugriff (vgl. Abb. 6.7). Über diese Funktion kann beispielsweise die Verfügbarkeit der in der Bibliothek stehenden Exemplare überprüft und digitale Ausgaben sofort herunter geladen werden, ohne das System wechseln zu müssen.

Abbildung 6.7.: In Kursräume eingebettete Seminarapparate der Bibliothek

Als drittes System wurde das Paderborner HIS-LSF-Portal[8] angebunden, um eine doppelte Datenerfassung bei dem Anlegen von Kursen zu vermeiden und angemeldete Teilnehmer zu Veranstaltungen mit den Daten in der Lern- und Arbeitsplattform zu synchronisieren (vgl. [GR07]).

Darüber hinaus existieren verschiedene JSR168-kompatible Portlets, die in entsprechenden Portalsystemen eingebunden werden können und verschiedene Sichten auf Informationen in koaLA bieten[9]. Jedes Dateiobjekt, jeder Forenbeitrag oder Diskussionsstrang sowie Weblogs oder Wikis können als RSS-Feed abonniert werden. Damit können Aktivitäten im System auch dann mitverfolgt werden, ohne dass ein Benutzer am System direkt angemeldet sein muss. RSS-Feeds können von lokalen Lesewerkzeugen (*RSS Reader*), speziellen Web-Anwendungen oder Portalen abgerufen und entfernt verwaltet werden. In koaLA selbst werden alle vom Benutzer abonnierten Feeds aggregiert auf einer entsprechenden Seite angezeigt. Direkt nach der Systemanmeldung erscheinen die aktuellsten zehn Neuigkeiten außerdem auf der persönlichen koaLA-Startseite.

[8] Hochschul-Informations-System Lehre-Studium-Forschung. Weitere Informationen auf den Webseiten der HIS GmbH unter `http://www.his.de/abt1/ab10`. Letzter Zugriff: 31.07.2008.

[9] Zum Beispiel Übersichten über die im aktuellen Semester belegten Kurse und aktuelle Termine.

Als externes Nachweissystem für E-Learning-Inhalte wurde der europäische ARIADNE-Knowledge-Pool für den Testbetrieb an koaLA angebunden. Wie in Abschnitt 4.2.6 beschrieben, können über Web-Services Lerninhalte angefragt und ausgelesen werden. Die Ergebnisse können in der sTeam-Persistenzschicht abgelegt und analog zu den digitalen Informationsobjekten aus dem elektronischen Seminarapparat in allen unterstützten Lernszenarien weiter verwendet werden. Darüber hinaus können eigene Inhalte über den ARIADNE-Pool veröffentlich und damit beispielsweise anderen Hochschulen zur Verfügung gestellt werden.

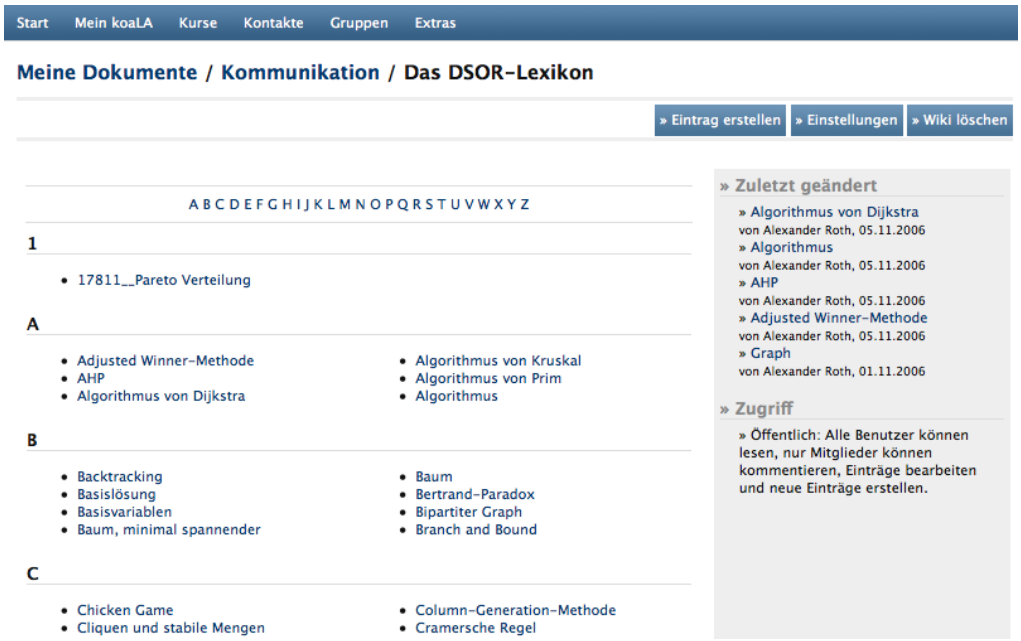

Abbildung 6.8.: Ein via Wiki-Technologie realisiertes Fachglossar für Operations Research/Management Science

Eine Anbindung anderer Lernplattformen an koaLA wurde im Rahmen des Locomotion-Projekts nicht vorgenommen. Diese Möglichkeit wurde jedoch bereits in einem vorherigen BMBF-Projekt „Virtuelles Studienfach Operations Research/ Management Science“ (kurz: VORMS[10]) vom Autor der vorliegenden Arbeit auf der gleichen Produktstandardarchitektur bereits implementiert (vgl. [RHS05] und [RS05]).

[10]Informationen über das virtulle Studienfach VORMS findet sich im Internet unter `http://www.vorms.org`. Letzter Zugriff: 31.07.2008.

Abbildung 6.9.: Weblogs werden vorrangig zur Ankündigung und Koordination genutzt.

6.4. Die hochschulweite Einführung von koaLA

Die hochschulweite Einführung der koaLA-Umgebung erfolgt im Rahmen des Locomotion-Projektes über drei Stufen: Testbetrieb, Pilotphase und hochschulweite Einführung.

Zunächst wurde koaLA zum Start des Wintersemester 2006/2007 für interessierte Studierende und Dozierende im Rahmen eines *Testbetriebs* eingeführt und in 20 Veranstaltungen als Lernmanagementsystem, Kommunikationsplattform und Gruppenarbeitsplatz genutzt. Die Dozierenden migrierten zumeist von eigenverantwortlich betriebenen oder eigens entwickelten Plattformen zu diesem zentralen Angebot. Da die Zahl der Veranstaltungen ausreichend für einen Testbetrieb waren, wurde die neue Plattform innerhalb der Universität zunächst nicht aktiv beworben. Trotzdem wurde festgestellt, dass nach nur zwei Wochen ca. 200 der zu diesem Zeitraum bereits angemeldeten 2000 Studierenden keine Kurse belegt hatten. Sie nutzten jedoch die Social Networking-Funktionen und das Angebot, sich in eigenen Arbeitsgruppen selbst organisieren zu können.

Die Bandbreite der Nutzung im Rahmen der Lehre reichte von kleinen Projektseminaren mit ca. 20 Teilnehmern bis zu Massenveranstaltungen mit über 800 Teilnehmern. Hier wurden unterschiedliche didaktische Szenarien realisiert. Die kleineren Seminare stellten kooperative Funktionen wie eine gemeinsame Materialsammlung und Diskussionen an

Dokumenten bereit, wobei die großen Veranstaltungen eher auf die reine Materialbereitstellung (Download) fokussiert waren und Foren eher zur Klärung organisatorischer Fragen einsetzen.

Einige wenige Dozierende nutzten in der Testphase bereits Weblogs um Informationen zur Veranstaltung zu veröffentlichen und zu diskutieren. Diese Funktion wurde sowohl von Dozierenden als auch von Studierenden als geeignete Darstellungsform für organisatorische Informationen, Hilfestellungen bei Übungsaufgaben und Motivation beschrieben.

Im Pilotbetrieb, der im Sommersemester 2007 lief, wurde die Nutzung auf 70 Veranstaltungen und über 5000 Nutzer ausgebaut. Als Referenzstudiengänge waren dabei insbesondere der „Zwei-Fach-Bachelor“ in den Kulturwissenschaften und die Informatik angesprochen. Punktuell hatte die Plattform aber bereits auch in anderen Bereichen Nutzer gewonnen. Neben Betrieb und Weiterentwicklung der Systeme wurden in der Pilotphase Schulungs- und Beratungsangebote initiiert. Für die Studierenden sind diese im etablierten Notebook-Café und bei der Schulungsinitiative doIT angesiedelt. Für die Lehrenden und Verwaltungsmitarbeiter wurde ein Schulungskonzept erarbeitet, das zusammen mit der Hochschuldidaktik umgesetzt wird.

Die Ausweitung auf den hochschulweiten Einsatz fand im Wintersemester 07/08 statt. Zum Zeitpunkt der Fertigstellung dieser Thesis im Sommersemester 2008 wurden über 400 Veranstaltungen mit koaLA unterstützt und der 11.000 Student auf der Plattform begrüßt.

6.5. Erfahrungen aus der Pilotbetriebsphase

Im Rahmen des Locomotion-Projektes wurde die Plattform mehrmals evaluiert (bspw. [G+08] und [Mar08]). Dabei wurde koaLA als niedrigschwelliges Angebot bestätigt, das leicht zu bedienen ist und wenig bis keine Hilfestellung erfordert[11]. Die meisten Dozierenden[12] denken, dass koaLA leicht zu benutzen ist (74 %) und dass die meisten Benutzer den Umgang mit koaLA sehr schnell erlernen können (85 %). Nur ein geringer Prozentsatz gab an, dass vor der Nutzung von koaLA viele Dinge erlernt werden mussten (4 %)[13].

Die Mehrzahl der Dozierenden, die koaLA einsetzen, nutzen die Plattform mehrmals pro Woche (54 %) oder öfter (17 %). Oft oder sehr oft benutzte Komponenten sind dabei Kursräume (70 %), internes E-Mailsystem[14] (39 %), Foren (28 %) und Gruppenarbeitsräume (25 %).

Das soziale Netzwerk (16 %), Weblogs (12 %) und Wikis (6 %) wurden weniger häufig benutzt. Einige Dozierende gaben an, dass sie den Unterschied zwischen Foren, Weblogs und Wikis nicht kennen. Die Möglichkeit, unterschiedliche Grade der Selbstorganisation für Arbeitsräume und Kommunikationswerkzeuge über einen Rechtedialog konfigurieren zu können, finden die meisten Dozierenden jedoch nützlich.

[11] Zu den Hilfestellungen zählt neben dem Supportangebot (telefonisch, E-Mail-basiert oder persönlich) auch eine Online-Hilfe der Plattform, die aus verschiedenen moderierten koaLA-Foren bestand und über eine gesonderte Funktion auf jeder Maske verfügbar war.

[12] An der hier wiedergegebenen Umfrage haben 67 Dozierende teilgenommen (vgl. [Mar08]).

[13] Die Dozierenden schätzten sich dabei bzgl. ihrer Computer- und Internetkenntnisse als Fortgeschrittene (40 %) bzw. Experten (40 %) ein. 20 % gaben nur grundlegende Kenntnisse an.

[14] Die Weiterleitung von internen Nachrichten an eine persönliche E-Mail-Adresse wird unterstützt.

Zusammenfassend lässt sich feststellen, dass das System durch die Bereitstellung als zentrales Angebot und die Integration in die IT-Infrastruktur in relativ kurzer Zeit eine weite Verbreitung an der Hochschule gefunden hat. Die (konfigurierbaren) Grundfunktionen von koaLA genügen, um die meisten Basisszenarien in der Lehre zu unterstützen.

Um eine bestimmte Didaktik oder Organisation zu unterstützen, wurden Funktionen der Plattform in Zusammenarbeit mit einzelnen Lehrstühlen und Fachbereichen durch zusätzliche Komponenten bereits erweitert.

Darüber hinaus ist ein Schulungsbedarf zu konstatieren, da Web 2.0-Werkzeuge wie Wikis, Weblogs und soziale Netzwerke bislang nur von wenigen Dozierenden für die Lehre adaptiert wurden. Hier ist es notwendig, die Unterschiede zwischen den neuen Konzepten zu vermitteln und den organisatorischen und didaktischen Nutzen durch entsprechende Einsatzszenarien zu veranschaulichen.

7. Schlussbetrachtung

Zum Abschluss des Hauptteils der Arbeit sollen alle Ergebnisse noch einmal kurz zusammengefasst werden. Daran anschließend folgt eine kritische Würdigung, die insbesondere auf den Nutzen, die Risiken und Erfolgsfaktoren des hier vorgestellten Ansatzes eingeht. Letztendlich ergeben sich durch die vorliegende Arbeit eine Reihe von Perspektiven, die sich weiter erforschen lassen.

7.1. Zusammenfassung

Eine organisationsweite Unterstützung von E-Learning an Präsenzhochschulen wird technologisch und organisatorisch häufig durch die große Heterogenität vorhandener Veranstaltungsformen, Didaktiken und multimedialen Systemen erschwert. Bisherige Ansätze zur Kopplung verschiedener Lernwerkzeuge fokussieren den Funktionsumfang des daraus entstehenden Systemverbundes, nicht die Wiederverwendbarkeit bzw. Integrationsfähigkeit. Somit ist die Entwicklung und Integration neuer Lernwerkzeuge nur mit relativ großem Aufwand möglich.

Diese Arbeit hat einen ganzheitlichen Ansatz aufgezeigt, mit dem kooperative Lern- und Arbeitsumgebungen für den universitären Einsatz entwickelt werden können. Der vorgestellte Ansatz basiert auf drei Elementen:

- Der vorgestellte *normsprachliche Spezifikationsrahmen* umfasst eine Terminologie aus 101 Ding-Prädikatoren und 39 Geschehnis-Prädikatoren, sowie eine auf sechs Satzmustern beschränkte Grammatik. Es wurde gezeigt, wie Anforderungen an statische Konzepte, Funktionalität und an Dynamik von Lernszenarien und -umgebungen mithilfe der Normsprache spezifiziert werden können.

- Das dargestellte *Konzept einer komponentenorientierten E-Learning-Architektur* spezifiziert eine Plattform, die für verschiedenste (verteilte) Anwendungen in der Domäne des universitären E-Learnings eine geeignete Basis darstellt. Es wurde gezeigt, wie man darauf aufbauend eine Produktarchitektur konzipiert und eine Integration in die universitäre Informationsarchitektur schafft.

- Die konstruierte *Methode* beschreibt ein zweigeteiltes Vorgehen zur Spezifikation und Entwicklung von Plattform und Anwendung. Sie umfasst ein Vorgehensmodell mit 49 Prozessen, ein ausführliches Beschreibungsmodell, ein Rollenmodell und eine Techniksammlung.

Zuletzt wurde die Realisierbarkeit des Ansatzes mit der an der Universität Paderborn hochschulweit eingeführten Lern- und Arbeitsplattform koaLA aufgezeigt.

7.2. Kritische Würdigung

Die Einführung zentraler Strategien an öffentlichen Hochschulen ist durch ihre spezifische Organisationsform – viele teilautonome Lehreinheiten mit dezentral verteilter Macht – schwierig. Hierunter fällt auch der in dieser Arbeit vorgestellte Ansatz zum Aufbau einer E-Learning-Infrastruktur, die Teil eines integrierten Informationsmanagements einer Hochschule ist.

Im Folgenden werden daher in Anlehnung an Roth und Hoppe der Nutzen, die Risiken und Erfolgsfaktoren des in dieser Arbeit vorgestellten Ansatzes kritisch dargestellt (vgl. [RH06] und [RH07]).

7.2.1. Nutzen

Aus pädagogisch-didaktischer Sicht muss der E-Learning-Einsatz zur Optimierung individueller Lernprozesse beitragen (vgl. [Bau03b]). Mit dem in dieser Arbeit vorgestellten Ansatz kann die individuelle Anpassbarkeit an verschiedene Lernszenarien durch vielfältige Kombinationen von Komponenten erheblich verbessert werden; womit E-Learning zielgerichteter eingesetzt werden kann. Darüber hinaus erlaubt die verwendete Nutzungsmetapher virtueller Wissensräume per se ein selbstorganisiertes Lernen und Arbeiten. Die Entwicklung der Komponente als Sicht hierauf schränkt das Maß an Selbstorganisation nur mehr oder weniger stark ein. Dadurch können stark strukturierte Lernszenarien auch mit selbstorganisierten Gruppen, Arbeitsräumen und Werkzeugen kombiniert werden, was in Folge einen medienbruchfreien Austausch von Informationen zwischen formalen und informellen Szenarien unterstützt und im Vergleich zu den meisten existierenden Plattformen den Studierenden deutlich in den Mittelpunkt stellt. Die Erfahrungen an der Universität Paderborn zeigen, dass diese Verbesserungen letztendlich die Akzeptanz von E-Learning im Allgemeinen und von einem zentral betriebenen hochschulweiten Angebot im Besonderen erhöhen können.

Aus *technologischer Sicht* muss Technologie als Mittel zum Zweck aufgabenangemessen eingesetzt werden, um die pädagogisch-didaktischen Ziele zu erreichen und Methodeneffizienz, Lernerfolg und Akzeptanz zu sichern. Speziell der Einsatz anerkannter Technologien sowie die Verwendung von Standards und wieder verwendbaren Modulen können diese Ziele gewährleisten. Die in dieser Arbeit konzipierte Rahmenarchitektur verwendet hauptsächlich Programmiersprachen-neutrale, XML-basierte Standards wie zum Beispiel SOAP, VSQL oder XHTML. Der hier vorgestellte Ansatz ist per se auf Modularität ausgerichtet. Wiederverwendbarkeit bei hohem Integrationsgrad des Systems macht einen Kernaspekt dieses Konzepts aus. Auch verteilte Komponenten sind bei verhältnismäßig geringem technischen Aufwand flexibel integrierbar. Wartung und Pflege sowie Weiterentwicklung werden vereinfacht; ggf. ist auch ein Outsourcing der Entwicklung und Wartung von Komponenten denkbar. Durch einfach gerichtete Abhängigkeiten zwischen Komponenten wird eine enge Kopplung zwischen Architekturelementen vermieden und die Durchführung von Änderungen an einzelnen Komponenten erleichtert.

Aus *ökonomischer Sicht* hat der hier vorgestellte Ansatz im Vergleich zu anderen Formen der Individualentwicklung langfristig positive Effekte auf alle Kostenarten, sowohl die monetär als auch die nicht-monetär quantifizierbaren Kosten des Einsatzes von E-Learning: Eine Unterstützung individueller Lernszenarien durch E-Learning kann durch die verkürzten Entwicklungszeiten relativ schnell bereitgestellt werden[1]. Netzwerkeffekte und damit Skalenerträge können durch einen möglichen hochschulweiten Einsatz stark zum Tragen kommen. Mit der verkürzten Entwicklungszeit ist auch eine Reduzierung der Entwicklungskosten verbunden. Die Kosten der Bereitstellung, Wartung und Weiterentwicklung sind in Bezug auf die potenzielle Nutzerzahl relativ gering. Ebenfalls begünstigt der komponentenbasierte Architekturansatz einen reduzierten Wartungsaufwand und eine Verbesserung der Kostenschätzung.

Darüber hinaus ist eine organisatorische Integration von Service- und Verwaltungsleistungen sehr gut möglich. Gleichzeitig ist die Einbindung in die bestehenden organisatorischen Strukturen sowie in die bestehende technologische Infrastruktur flexibel und kosteneffizient möglich. Zusatznutzen wird dadurch gewährt, dass Komponenten, Medien bzw. Konzepte hochschulweit genutzt werden können, die ansonsten nur an einzelnen Fachbereichen oder Lehrstühlen verfügbar wären. Gleichzeitig kann durch die Verwendung des vorgestellten Ansatzes eine Abgrenzung von anderen Bildungseinrichtungen erfolgen — eine Profilierung der Hochschule wird so möglich. Nicht zuletzt kann durch Erforschung und Einsatz innovativer, zukunftsweisender Schlüsseltechnologien im E-Learning die Führungsrolle von Hochschulen ausgebaut bzw. bestätigt werden, was als übergeordnetes Nutzenpotenzial berücksichtigt werden muss.

7.2.2. Risiken

Die komponentenbasierte, dienstorientierte Entwicklung von E-Learning-Werkzeugen ist im Kontext universitärer Informationsarchitekturen ein zukunftsweisendes Konzept, das in Deutschland bislang nur in Pilotprojekten zum Einsatz kam. Es ist zwar auch isoliert an einzelnen Lehrstühlen oder Fachbereichen einsetzbar, kann seinen vollen Nutzen jedoch nur im Zusammenhang mit Kooperationen bieten. Dabei sind nicht nur Kooperationen zwischen Hochschulen, sondern bereits Kooperationen auf Fakultäts- oder Lehreinheitsebene relevant.

Der Ansatz unterliegt hierbei direkten Netzwerkeffekten, da die Anzahl bereits vorhandener Werkzeuge und Komponenten und damit indirekt auch die Anzahl der Nutzer als ausschlaggebend für die Vorteile des Entwicklungsansatzes angesehen werden müssen. Aufgrund dieser Netzwerkexternalitäten kann der so genannte „Pinguineffekt“ (vgl. [FS85]) zum Tragen kommen. Dieser Effekt besagt, dass frühe Anwender einer neuen, überlegenen Technologie nur einen geringen Nutzen ziehen, weil noch nicht genügend andere Nutzer daran beteiligt sind. In der Folge verharren potenzielle Anwender in abwartender Haltung, wenngleich Netzwerkeffekte und Skalenerträge Nutzen für alle Anwender bieten würden.

[1] Entwurf und Entwicklung der ersten zum Testbetrieb freigegebenen Version von koaLA hat auf Grundlage der bereits vorhandenen Standardarchitektur 4 Personenmonate gedauert. Diese hatte bereits einen Großteil der Funktionen der aktuellen, hochschulweit ausgerollten Version.

Im Zusammenhang mit dieser eventuellen Abwartehaltung kann es auch zu Akzeptanzproblemen des neuen Ansatzes kommen. Dies kann unter anderem auch in Zusammenhang mit dem Know-how stehen das erforderlich ist, um den dienstorientierten Komponentenansatz zu unterstützen. Dabei ist nicht nur ein Architekturmanagement vonnöten, das neben den zu entwickelnden Anwendungsarchitekturen auch die Produktstandardarchitektur und die universitären Infrastrukturdienste umfasst, sondern auch die Fähigkeit gefragt, neue Komponenten auf dieser Basis zu entwickeln und zu integrieren.

Wie bei jedem Produkt, das nicht „von der Stange“ gekauft sondern individuell entwickelt wird, entsteht weiterhin Aufwand für die Spezifikation und Entwicklung grundlegender Basisszenarien bzw. domänenspezifischer Dienste. Hierbei ist speziell auf die Definition von Prozessen in Lehre und Veranstaltungsorganisation zu verweisen, die sich für viele Dozierende schwierig gestaltet. Ohne eine grundlegende Prozess- bzw. Szenariendefinition lassen sich Lehrprozesse und -szenarien jedoch nur schwer effektiv und effizient unterstützen. Der Einsatz der normsprachlichen Spezifikation hat sich hierfür in der Praxis jedoch als geeignetes Mittel erwiesen.

7.2.3. Erfolgsfaktoren

Die dargelegten Risiken bei der Einführung einer zentralen Anwendungsplattform und einer dienstorientierten Individualentwicklung erlauben die Ableitung kritischer Erfolgsfaktoren; jene Faktoren, die ausschlaggebend für den erfolgreichen und damit effektiven, effizienten und nachhaltigen Einsatz von universitärem E-Learning mithilfe komponentenorientierter, integrativer Werkzeuge. Im Wesentlichen betreffen die im Folgenden kurz dargestellten Erfolgsfaktoren die Probleme einer eventuellen Abwartehaltung, möglicher Akzeptanzprobleme und gegebenenfalls fehlenden Know-hows:

Strategische Planung Die Entwicklung und Einführung einer dienstorientierten Komponentenarchitektur zur Unterstützung universitären E-Learnings sollte in Form eines gute geplanten Projektes erfolgen. Essenziell ist dabei, alle beteiligten bzw. betroffenen Organisationseinheiten und Personen mit ihren Anforderungen und Bedürfnissen zu berücksichtigen und neben den technologischen auch pädagogische und ökonomische Gesichtspunkte einzubeziehen. Das für den Ansatz notwendige Change Management sollte auf Basis eines umfassenden, konsistenten und mit allen relevanten Planungsebenen abgestimmten strategischen Konzeptes erfolgen (vgl. [Hop05a]). Auch organisatorische Aspekte müssen dabei bedacht werden (vgl. [Hop05b]). Dies gilt insbesondere für Entwurf, Entwicklung, Integration, Betrieb und Support von Individualentwicklungen auf Basis der Standardplattform; dies könnte als Dienstleistung bspw. von einem Multimedia- oder E-Learning-Kompetenzzentrum angeboten und durch ein geeignetes System vergütet werden.

Systematisches Architekturmanagement Um die Komponentenentwicklung im Spannungsfeld zwischen strategischen Zielen und den konkreten fachlichen Anforderungen an ein integriertes Informationsmanagement am Campus effizient und nachhaltig durchführen zu können, ist ein systematisches Architekturmanagement erforderlich

(vgl. [Der03], [Foe03]). Hierunter ist ein Prozess zu verstehen, der ein methodisch fundiertes Vorgehen zur Gestaltung einer umfassenden Architektur (und ihrer Subarchitekturen) liefert, unter Beachtung im Vorfeld gesetzter mittel- oder langfristiger Architekturziele und technologischer Rahmenbedingungen. Dabei müssen zwei Ebenen betrachtet werden: Zum einen die IT-Infrastruktur, welche die Hard- und Softwarekomponenten beschreibt und zum anderen die Architektur sozialer Systeme, welche die Organisationen, Prozesse und Ressourcen betrachtet.

Zentrale Vermarktung Teil der aufbauorganisatorischen Konzeption ist auch die Überlegung, inwieweit die eingesetzten E-Learning-Systemkomponenten vermarktet werden können und wie diese Vermarktung ggf. vonstatten gehen soll. Speziell im Hinblick auf den komponentenorientierten, integrativen Ansatz ist eine zentrale Vermarktung erfolgskritisch. Da Integrität, Wiederverwendbarkeit und Kooperation im Vordergrund dieses Ansatzes stehen, kann nur eine Zusammenarbeit aller Akteure die Nutzenpotenziale zur vollen Entfaltung bringen. Dies gilt nicht nur innerhalb einer Hochschule, sondern auch nach außen, im Verbund mit anderen Bildungseinrichtungen. Zentrale Vermarktung (und ggf. auch weitere zentrale Aufgaben und Aktivitäten) könnten ebenfalls durch ein eigenes Multimedia- oder E-Learning-Kompetenzzentrum wahrgenommen werden (vgl. zu dieser Auffassung auch [Bac01]). Eine Vermarktung von E-Learning-Komponenten ist sowohl hochschulintern als auch hochschulextern denkbar; langfristig ist die verstärkte Entwicklung in Richtung eines Komponentenmarktes zu erwarten, in den Hochschulen dann eintreten können.

Promotoren Um Nutzenpotenziale neuer Lösungen zu verdeutlichen, ist es sinnvoll, Promotoren einzusetzen. Speziell der Hochschulleitung kommt eine große Bedeutung als Einflussfaktor auf die Nachhaltigkeit des E-Learning-Einsatzes zu (vgl. auch [SE04]). Entscheidungspersonen müssen voll hinter dem neu einzuführenden bzw. neu eingeführten System stehen und dessen erfolgreichen Einsatz „vorleben“, um allen anderen Beteiligten als Vorbild zu dienen.

Qualifikationsmaßnahmen Um zu gewährleisten, dass die Vorteile des zukunftsweisenden Ansatzes komponentenorientierter Lernplattformen auch ausgeschöpft werden können, muss das Wissen zur Konfiguration, Nutzung, Integration und ggf. auch Entwicklung von Komponenten vorhanden sein. Speziell in Bezug auf die Definition von Prozessen und Szenarien ist die Kompetenz notwendig, über eine Sprache bzw. ein Modell zur Abbildung von Lehr- und Lernszenarien zu verfügen, die als gemeinsame Diskussionsgrundlage für Architekten, Entwickler und Nutzer dienen kann. Um dies zu erreichen und gleichzeitig unabhängig von Dritten zu sein, sind entsprechende Qualifikationsmaßnahmen für ausgewählte Beteiligte nötig. Dadurch wird gleichzeitig positiv auf Verständnis und Akzeptanz eingewirkt.

7.3. Ausblick

Die Ziele der vorliegenden Arbeit wurden erfüllt. Aus den Ergebnissen ergeben sich jedoch gleichzeitig neue Fragestellungen technischer, organisatorischer und betriebswirtschaftlicher Art, die den Bedarf an weiterer Forschungsarbeit begründen.

Komponentenmärkte für E-Learning-Technologien Ähnlich wie Lerninhalte können auch E-Learning-Komponenten in anderen Kontexten wieder verwendet werden. Die Voraussetzung dazu ist ihre Interoperabilität, d. h. eine zur Architektur passende technische Basis (*architectural match*) und Semantik. Im Gegensatz zu Lerninhalten sind Komponenten und Werkzeuge zumindest vom Grundsatz her didaktisch „neutraler", so dass Lehrende mit anderen Zielgruppen und/oder anderen didaktischen Präferenzen ein solches Angebot eher übernehmen werden als Inhalte[2]. Plattformen zum Austausch von Softwarekomponenten könnten einerseits organisatorischer Natur sein, wie bspw. die bereits in Kapitel 2 vorgestellte Open Source-Softwarebörse CampusSource e.V., oder technischer Natur, wie die von Microsoft zur Verfügung gestellte Austauschplattform *Microsoft Solution Sharing Network*[3], über die auf Basis von Sharepoint entwickelte E-Learning-Komponenten zwischen Universitäten ausgetauscht werden können. Je mehr Universitäten zukünftig auf eine serviceorientierte Umsetzung eines integrierten Campus-Managements setzen, desto eher wird sich ein Markt für kommerzielle und freie Softwarebausteine entwickeln, die in lokale Architekturen eingebunden werden können. Die daraus resultierende Fragestellung ist, wie und unter welchen Rahmenbedingungen sich Universitäten auf einem solchen Markt behaupten wollen bzw. können.

Generative Programmierung Die in Kapitel 4 erarbeitete Normsprache ist zur formalen Anforderungsbeschreibung von Lern- und Arbeitsszenarien geeignet, die nach der vorgestellten Methodik in entwicklungsspezifische Modellierungssprachen transformiert und auf dieser Basis implementiert werden. Aus softwaretechnischer Sicht ist eine weitere Formalisierung der Normsprache interessant, um von rekonstruierten Anforderungsbeschreibungen und einem Komponentenrepository automatisch bestehende Komponenten konfigurieren und neue Komponenten generieren zu können (vgl. [CE00]).

Interne Organisation Das Management einer serviceorientierten Komponentenarchitektur erfordert einen systematischen Ansatz, der einen technischen als auch organisatorischen Paradigmenwechsel impliziert. Die Voraussetzungen dafür sind die Sicherstellung eines ganzheitlichen Überblicks über die technische Infrastruktur, die Übernahme von Planungs-, Entwicklungs- und Integrationsaufgaben von einer zentralen Stelle sowie die Implementierung eines Vergütungssystems für entsprechende Dienstleistungen. Hier stellt sich die Frage nach geeigneten Organisationsformen.

[2] Aus den vom BMBF geförderten Content-Projekten hat man u. a. das Resümee gezogen, dass Lerninhalte unter Hochschullehrern nur in einem sehr begrenzten Umfang wieder verwendet werden. In einschlägigen Kreisen grassiert daher der Witz, dass Hochschullehrer eher die Zahnbürste einer Kollegin/eines Kollegen verwenden, als dessen Content.

[3] Im Internet verfügbar unter `http://www.microsoft.com/ssneduca`. Letzter Zugriff: 31.07.2008.

Referenzimplementierungen für verteilte Wissensräume Verbundhochschulen stellen als virtuelle Organisationen höhere Anforderungen an die technische Umsetzung. Hierbei ist auf Ebene der Wissensräume ebenfalls ein Verbund zu schaffen, der den Benutzern – unabhängig von der darunter liegenden Serverstruktur – einen transparenten Wechsel zwischen lokalen und hochschulübergreifenden Rauminstanzen gewährt. Diese Problemstellung umfasst neben der Mobilität auch Aspekte der Skalierbarkeit, der Verteilung von Resourcen und der Verfügbarkeit von Servern im Verbund. Erste Lösungsansätze für dieses Problem sind in [Bop06] zu finden, Referenzimplementierungen stehen allerdings noch aus.

7.4. Fazit

Der Paradigmenwechsel von der kooperativen IT-Versorgung hin zu einem integrierten Informationsmanagement und dessen technische Realisierung durch serviceorientierte Architekturen machen langfristig eine komponentenorientierte Entwicklung von Lernwerkzeugen erforderlich. Bislang ist bis auf wenige Ausnahmen eine Wiederverwendung von Systemkomponenten oder -modellen im universitären E-Learning nicht geglückt. Diese Arbeit liefert einen ganzheitlichen Ansatz. Sie stellt eine Terminologie und Grammatik zur bereichsübergreifenden Anforderungserhebung vor, spezifiziert eine auf den Problembereich zugeschnittene Rahmenarchitektur und ein dazu passendes Vorgehensmodell zur Entwicklung von E-Learning-Werkzeugen. Die Arbeit zeigt somit eine systematische Herangehensweise auf, die auf dem Einsatz verfügbarer Technologien und innovativer Konzepte beruht. Fehlende Bausteine werden auf die speziellen Anforderungen universitären E-Learnings ausgerichtet und fußen auf anerkannten softwaretechnischen Konzepten und Technologien.

Der vorgestellte systematische Ansatz wurde mit der Implementierung und Einführung der hochschulweiten koaLA-Plattform an der Universität Paderborn praktisch angewendet. Damit wurde im Locomotion-Projekt der Spagat geschafft, der Vielfalt von E-Learning für Forschung und Profilierung eine integrierende Plattform zu bieten, und darüber hinaus allen Lehreinheiten, die dies aus eigener Kraft bzw. finanziellen Ressourcen nicht stemmen können, ein niedrigschwelliges, zentrales Angebot. So wurde bewiesen, dass das in dieser Thesis erarbeitete Konzept in der Praxis umsetzbar ist und als mögliche Alternative zur Einführung einer klassischen, geschlossenen Lernumgebung an Hochschulen in Betracht gezogen werden kann.

Literaturverzeichnis

[A+02] Ackermann, J. u. a.: Vereinheitlichte Spezifikation von Fachkomponenten. Memorandum des Arbeitskreises 5.10.3 Komponentenorientierte betriebliche Anwendungssysteme / Gesellschaft für Informatik. 2002. – Forschungsbericht

[A+03] Anderson, T. u. a.: IMS Abstract Framework / IMS Global Learning Consortium. Version: 2003. `http://www.imsglobal.org/af/afv1p0/imsafwhitepaperv1p0.html`. 2003. – White Paper

[A+07] Aumann, S. u. a.: Personalisierte Webportale für Hochschulen. Eine Empfehlung der DINI-AG "Webportale" / Deutsche Initiative für Netzwerkinformation e. V. Göttingen, 2007. – Forschungsbericht

[AJ05] Apostolopoulos, N. ; Juhnke: FUeL – FU e-Learning. In: Fellbaum, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens.* Aachen : Shaker Verlag, 2005, S. 25–34

[AKTZ04] Arnold, P. ; Kilian, L. ; Thillosen, A. ; Zimmer, G.: *E-Learning, Handbuch für Hochschulen und Bildungszentren.* Nürnberg : BW-Verlag, 2004

[Alb03] Albrecht, R.: *E-Learning in Hochschulen. Die Implementierung von E-Learning an Präsenzhochschulen aus hochschuldidaktischer Perspektive.* Berlin : Verlag im Internet, 2003

[Alb07] Alby, T.: *Web 2.0. Konzepte, Anwendungen, Technologien.* München, Wien : Carl Hanser Verlag, 2007

[Ale06] Alexander, B.: Web 2.0 – A New Wave of Innovation for Teaching and Learning. In: *Educause Review* (2006), S. 33–44

[AMM06] Arnold, P. ; Mayrberger, K. ; Merkt, M.: E-Learning als Prozessinnovation zwischen Strategie und Didaktik. In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 27–38

[And76] Anderson, J. R.: *Language, memory and thought.* Hillsdale, NJ, USA : Erlbaum, 1976

[And03] Andresen, A.: *Komponentenbasierte Softwareentwicklung – mit MDA, UML und XML.* München, Wien : Hanser, 2003

[AP00] Arnold, P. ; Putz, P.: Communities of Practice als Orientierungsrahmen für die Gestaltung virtueller Lernumgebungen. In: Scheuermann, F. (Hrsg.): *Campus 2000 – Lernen in neuen Organisationsformen.* Münster : Waxmann Verlag, 2000

[ASD07] ASD: ASD Simplified Technical English Specification ASD-STE 100: International specifiaction for the preparation of maintenance documentation in a controlled language / AeroSpace and Defence Industries Association of Europe. Brüssel, 01 2007 (4). – Forschungsbericht

[AWKL06] Affolter, B. ; Wilding, B. ; Korner, M. ; Lautenschlager, P.: Video-Streaming und -Podcasting – universitäre Bildung für unterwegs? In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 276–286

[B+96] Boles, D. u. a.: Zwischenbericht der Projektgruppe Multimedia-Präsentationen im Gesundheitswesen / Universität Oldenburg, Fachbereich Informatik. Oldenburg, April 1996. – Forschungsbericht

[Baa99] Baacke, D.: Medienkompetenz als zentrales Operationsfeld von Projekten. In: *Handbuch Medien: Medienkompetenz – Modelle und Projekte.* Bonn : Bundeszentrale für Politische Bildung, 1999, S. 31–35

[Bac01] Bachmann, G.: Strategische Planung, Förderprogramme und Institutionen an Schweizer Hochschulen. In: *Tagungsband 9. Europäischer Kongress und Fachmesse für Bildungs- und Informationstechnologie LearnTec 2001* Bd. 2. Karlsruhe : Karlsruher Messe- und Kongress-GmbH, 2001, S. 461–468

[Bal01] Balzert, H.: *Lehrbuch der Software-Technik: Softwareentwicklung.* Bd. 1. 2. Auflage. Heidelberg, Berlin : Spektrum Verlag, 2001

[Bau03a] Baumgartner, P.: audit Förderprogramm Neue Medien in der Bildung – Förderschwerpunk Hochschule / DLR Projektträger Neue Medien in der Bildung. St. Augustin, 2003. – Forschungsbericht

[Bau03b] Baumgartner, P.: Evaluation mediengestützten Lernens. Theorie – Logik – Modelle. In: Kindt, M. (Hrsg.): *Projektevaluation in der Lehre. Multimedia an Hochschulen zeigt Profil(e).* Münster : Waxmann Verlag, 2003, S. 63–99

[Bau06a] Baudry, A.: *Ein Modell zur strukturellen Beschreibung von formatierbaren Lernmodulen für die orthogonale Konstruktion konsistenter Lerneinheiten*, Universität Paderborn, Diss., 2006

[Bau06b] Baumgartner, P.: E-Learning-Szenarien. Vorarbeiten zu einer didaktischen Taxonomie. In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 238–247

[Bau06c] Baumgartner, P.: Web 2.0: Social Software & E-Learning. In: *Computer + Personal (CoPers)* 14 (2006), Nr. 8, S. 20–34

[BBSS01] Back, A. ; Bendel, O. ; Stoller-Schai, D.: *E-Learning im Unternehmen. Grundlagen – Strategien – Methoden – Technologien.* Zürich : Orell Füssli, 2001

[BCG03] Baving, Tracy ; Cook, Donald ; Green, Trevor: Integrating the Educational Enterprise / Department of Computer Science, University of Cape Town. 2003 (CS03-11-00). – Forschungsbericht

[BCK03] Bass, L. ; Clements, P. ; Kazman, R.: *Software Architecture in Practice.* 2. Auflage. Boston, MA, USA : Addison-Wesley, 2003

[BD05] Beuschel, W. ; Draheim, S.: Potenziale kooperativer Medien für neue Lehr- und Lernformen – das Beispiel Weblogs. In: Fellbaum, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens.* Aachen : Shaker Verlag, 2005, S. 225–236

[Bec03] Beck, J.: Elektronische Geschäftsprozesse mit Web Services. In: *Online 2003 Messe Düsseldorf – Web Services: Schlüssel für eBusiness Integration.* Online GmbH, 2003

[BEF+56] Bloom, B.S. ; Engelhart, M.B. ; Furst, E.J. ; Hill, W.H. ; Krathwohl, D.R.: Taxonomy of educational objectives. The classification of educational goals. In: *Handbook I., Cognitive Domain.* New York, NY, USA : Longman, 1956

[Beu05] Beuche, D.: Softwareproduktlinien-Entwicklung mit Merkmalmodellen. In: *OBJEKTspektrum* (2005), Nr. 6, S. 74–79

[BFH+04] Brennecke, A. ; Förster, A. ; Hohenhaus, M. ; Meyer, M. ; Oevel, G. ; Szegunis, J.: Pflichtenheft für eine im Rahmen von Uni-Mobilis konzipierte Dienste-Infrastruktur / Universität Paderborn. Paderborn, 2004. – Forschungsbericht

[BG94] Bandler, R. ; Grinder, J.: *Struktur der Magie: Metasprache und Psychotherapie.* 6. Paderborn : Junfermann, 1994

[BGKT07] Bazijanec, B. ; Gausmann, O. ; Klöckner, S. ; Turowski, K.: Analyse von Risikofaktoren bei der Einführung, Integration und Migration von integrierten Informationssystemen an mittelgroßen deutschen Hochschulen. In: Gaedke, M. (Hrsg.) ; Borgeest, R. (Hrsg.): *Integriertes Informationsmanagement an Hochschulen. Quo vadis Universität 2.0?* Karlsruhe : Universitätsverlag Karlsruhe, 2007, S. 37–53

[BHH+06] Bopp, T. ; Hampel, T. ; Hinn, R. ; Lützenkirchen, F. ; Prpitsch, C. ; Richter, H.: Alltagstaugliche Mediennutzung erfordert Systemkonvergenzen in Aus- und Weiterbildung. In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 87–96

[BK02] Bunse, C. ; Knethen, A. von: *Vorgehensmodelle kompakt.* Heidelberg, Berlin : Spektrum Verlag, 2002

[BL99] Berners-Lee, T.: *Transkript der Rede von Tim Berners-Lee am 14.04.1999: MIT Laboratory for Cmputer Science (LCS) 35th Aniversary celebrations, Cambridge Massachusetts.* `http://www.w3.org/1999/04/13-tbl.html`. Version: 1999, Abruf: 30.06.2008

[BLF99] Berners-Lee, T. ; Fischetti, M.: *Weaving the Web.* New York, NY, USA : Harper Collings Publishers, 1999

[BLK01] BLK: Lebenslanges Lernen / Bund-Länder-Kommission für Bildungsplanung und Forschungsförderung. Version: 2001. `http://www.blk-bonn.de/papers/heft88.pdf`. Bonn, 2001 (Heft 88). – Forschungsbericht

[BLK03] BLK: Lebenslanges Lernen – ein Zwischenresümee der Wissenschaftlichen Begleitung / Bund-Länder-Kommission für Bildungsplanung und Forschungsförderung. Version: 2003. `http://www.blk-lll.de/LLL/LIT/NEWSLETTER(Sond).pdf`. 2003. – Forschungsbericht

[Blu98] Blumstengel, A.: *Entwicklung hypermedialer Lernsysteme.* Berlin : Wissenschaftlicher Verlag Berlin, 1998

[BMB00] BMBF: Förderprogramm Neue Medien in der Bildung – Lehr- und Lernsoftware / Bundesministerium für Bildung und Forschung. Bonn, 2000. – Forschungsbericht

[BMB04a] BMBF: Bundesbericht Forschung 2004 / Bundesministerium für Bildung und Forschung. Version: 2004. `http://www.bmbf.de/pub/bufo2004.pdf`. Bonn/Berlin, 2004. – Forschungsbericht

[BMB04b] BMBF: Kursbuch eLearning 2004 – Produkte aus dem Förderprogramm / Bundesministerium für Bildung und Forschung. Bonn, 2004. – Forschungsbericht

[BMRS96] Buschmann, F. ; Meunier, R. ; Rohnert, H. ; Sommerland, P.: *Pattern Oriented Software Architecture: A System of Patterns.* 1. Auflage. Chichester, UK : John Wiley & Sons, 1996

[Böl02] Böllert, K.: *Objektorientierte Entwicklung von Software-Produktlinien zur Serienfertigung von Software-Systemen*, Universität Ilmenau, Diss., 2002

[Bön04] Bönkost, K. J.: Studienreform und Multimedia: Wissens-Upgrade für die Hochschullehrer. In: Krylov, A. (Hrsg.) ; Oberliesen, R. (Hrsg.): *Zukunft gestalten. Transnationaler Dialog zur Entwicklung von Bildung und Gesellschaft.* National Business Institute, 2004, S. 97–116

[Bop06] Bopp, T.: *Verteilte kooperative Wissensräume*, Universität Paderborn, Diss., 2006

[Bos00] Bosch, J.: *Design & Use of Software Architectures – Adopting and evolving a product-line approach*. 1. Auflage. New York, NY, USA : ACM Press, 2000

[BP94] Baumgartner, P. ; Payr, S.: *Lernen mit Software*. Innsbruck : Österreichischer Studienverlag, 1994

[BP04] Baumgartner, P. ; Preussler, A.: "Wir wären nicht hier, wo wir jetzt sind!" Medidaprix im Spiegel der Community. In: Brake, C. (Hrsg.) ; Topper, M. (Hrsg.) ; Wedekind, J. (Hrsg.): *Der MEDIDAPRIX: Nachhaltigkeit durch Wettbewerb*. Münster : Waxmann Verlag, 2004, S. 165–176

[Brä03] Brännback, M.: R&D collaboration: role of Ba in knowledge creating networks. In: *Knowledge Management Reserach & Practice* 1 (2003), S. 28–38

[Bre98] Bremer, G.: Genealogie von Entwicklungsschemata. In: Müller-Luschnat, G. (Hrsg.) ; Oberweis, A. (Hrsg.): *Vorgehensmodelle für die Betriebliche Anwendungsentwicklung*. Teubner, 1998, S. 32–59

[Bre04] Bremer, C.: E-Learning-Strategien im Spannungsfeld von Hochschulentwicklung, Kompetenzansätzen und Anreizsystemen. In: Bremer, C. (Hrsg.) ; Kohl, K. (Hrsg.): *E-Learning Strategien – E-Learning Kompetenzen an Hochschulen*. Bielefed : AHD, 2004, S. 9–30

[Bro04] *Kapitel* Begriff. In: *Der Brockhaus*. Bd. 1. Leipzig, Mannheim : F. A. Brockhaus, 2004, S. 231

[Bru04] Brugger, R.: Auswahl und Betrieb von Lernplattformen. In: Euler, D. (Hrsg.) ; Seufert, S. (Hrsg.): *E-Learning in Hochschulen und Bildungszentren*. Oldenbourg, 2004, S. 423–438

[Bue01] Buechtemann, C. F.: Perspektiven der Hochschul- und Wissenschaftspolitik – Deutsche Nachwuchswissenschaftler in den USA / Bundesministerium für Bildung und Forschung. Bonn, 2001. – Forschungsbericht

[BV03] Birkhölzer, T. ; Vaupel, J.: *IT-Architekturen – Planung, Integration, Wartung*. Berlin-Offenbach : VDE Verlag, 2003

[BWHW05] Braun, C. ; Wortmann, F. ; Hafner, W. ; Winter, R.: Method construction – a core approach to organizational engineering. In: *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA : ACM Press, 2005, S. 1295–1299

[Cas05] Castells, M.: *Die Internet-Galaxie. Internet, Wirtschaft und Gesellschaft*. 1. Auflage. Wiesbaden : Verlag für Sozialwissenschaften, 2005

[CB04] Carstensen, D. ; Barrios, B.: *Campus 2004 – Kommen die digitalen Medien an den Hochschulen in die Jahre?* Münster : Waxmann Verlag, 2004

[CE00] Czarnecki, K. ; Eisenecker, U. W.: *Generative Programming – Methods, Tools, and Applications*. 1. Auflage. Boston, MA, USA : Addison-Wesley, 2000

[CET05] CETIS: *The E-Learning Framework*. `http://www.elframework.org/framework`. Version: 2005, Abruf: 09.01.2008

[CG98] Chroust, G. ; Grünbacher, P.: Werkzeugunterstützung beim Einsatz von Vorgehensmodellen. In: Kneuper, R. (Hrsg.): *Vorgehensmodelle für die betriebliche Anwendungsentwicklung*. Stuttgart : Teubner, 1998

[CMO72] Cohen, M. D. ; March, J. G. ; Olsen, J. P.: A Garbage Can Model of Organizational Choice. In: *Administrative Science Quarterly* 17 (1972), S. 1–25

[Coa91] Coad, P.: OOD Critera, Part 2. In: *Journal of Object-Oriented Programming* 4 (1991), S. 64–66

[Coe02] Coenen, Olaf: *E-Learning-Architektur für universitäre Lehr- und Lernprozesse.* Lohmar, Köln : Josef Eul Verlag, 2002

[Dav05] Davenport, T. H.: *Thinking for a Living.* Boston, MA, USA : Harvard Business School Press, 2005

[Deg04] Degkwitz, A.: Das Informations-, Kommunikations- und Medienzentrum (IKMZ) der BTU Cottbus als Infrastruktureinrichtung für multimediales Lehren und Lernen. In: Fellbaum, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens.* Aachen : Shaker Verlag, 2004, S. 3–12

[DEH+02] Doberkat, E.-E. ; Engels, G. ; Hausmann, J. ; Lohmann, M. ; Veltmann, C.: Anforderungen an eine eLearning-Plattform – Innovation und Integration / Ministerium für Schule, Wissenschaft und Forschung in NRW. Dortmund, Paderborn, 2002. – Forschungsbericht

[Der03] Dern, G.: *Management von IT-Architekturen. Informationssysteme im Fokus von Architekturplanung und -entwicklung.* 1. Auflage. Wiesbaden : Vieweg, 2003

[DiN99] DiNucci, D.: Fragmented Future. In: *AllBusiness – Champions of Small Business* (1999), Nr. July Issue

[DIN04a] DIN: DIN PAS 1032-1:2004: Referenzmodell für Qualitätsmanagement und Qualitätssicherung – Planung, Entwicklung, Durchführung und Evaluation von Bildungsprozessen und -angeboten / DIN Deutsches Institut für Normung e. V. Berlin, 2004. – Forschungsbericht

[DIN04b] DIN: DIN PAS 1032-2:2004: Didaktisches Objektmodell – Modellierung und Beschreibung didaktischer Szenarien / DIN Deutsches Institut für Normung e. V. Berlin, 2004. – Forschungsbericht

[DL04] Dyson, P. ; Longshaw, A.: *Architecting Enterprise Solutions. Patterns for High-Capability Internet-based Systems.* Weinheim : Wiley Verlag, 2004

[Dow87] Dowson, M.: Iteration in the Software Process. In: *Proceedings of the 9th Int. Conference on Software Engineering.* Washington, DC, USA : IEEE Computer Society Press, 1987

[DS07] Degkwitz, A. ; Schirmbacher, P.: *Informationsinfrastrukturen im Wandel. Informationsmanagement an deutschen Universitäten.* Bad Honnef : Bock und Herchen Verlag, 2007

[Dür06] Dürscheid, C.: Neue Lernwelten, neue Kommunikationsformen – ein Blick in die Zukunft. In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 15

[Ede00] Edelmann, W.: *Lernpsychologie.* 6. Auflage. Kempten : Köser Verlag, 2000

[Ede02] Ederleh, J.: Nachhaltigkeitsstrategien für E-Learning im Hochschulbereich – Länder, Hochschulen, Projekte / HIS-Hochschul-Informations-System GmbH. Hannover, 2002. – Forschungsbericht

[EGHP05] Ehlers, U. ; Goertz, L. ; Hildebrandt, B. ; Pawlowski, J.: Qualität im E-Learning / Europäisches Zentrum für die Förderung der Berufsbildung. 2005. – Forschungsbericht

[EKW05] Ederleh, J. ; Kleinmann, B. ; Wannemacher, K.: E-Learning-Strategien deutscher Universitäten – Fallbeispiele aus der Hochschulpraxis / Hochschul-Informations-System. Hannover, 2005. – Forschungsbericht

[Epp01] Eppler, M. J.: Kognitive Werkzeuge als Instrumente des persönlichen Wissensmanagements. In: Reinmann-Rothmeier, G. (Hrsg.) ; Mandl, H. (Hrsg.): *Psychologie des Wissensmanagements: Perspektiven, Theorien und Methoden.* Göttingen : Hogrefe, 2001, S. 251–258

[Erp06] Erpenbeck, C.: Social Skills durch Social Software – Ein Mythos oder wie ändert sich die soziale Kompetenz durch den Einsatz von eLearning/Neuen Medien? In: *Keynote auf der 2nd EduMedia Conference 2006, 23. und 24. Mai 2006 in Salzburg*. Salzburg, 2006

[Ert01] Ertel, W.: *Angewandte Kryptographie*. Leipzig : Carl Hanser Verlag, 2001

[ES04] Euler, D. ; Seufert, S.: Von der Pionierphase zur nachhaltigen Implementierung – Facetten und Zusammenhänge einer pädagogischen Innovation. In: Seufert, S. (Hrsg.) ; Euler, D. (Hrsg.): *E-Learning in Hochschulen und Bildungszentren*. München : Oldenbourg, 2004, S. 1–24

[ES05] Euler, D. ; Seufert, S.: *E-Learning in Hochschulen und Bildungszentren*. Oldenbourg, 2005

[F+02] Fischer, J. u. a.: *Bausteine der Wirtschaftsinformatik: Grundlagen, Anwendung, PC-Praxis*. Berlin : Erich Schmidt Verlag, 2002

[Fär94] Färber, G.: *Prozessrechentechnik*. 3. Auflage. Berlin : Springer, 1994

[FB06] Fischer, A. ; Breiter, A.: Prozessorientiertes IT-Service-Management an Hochschulen. In: Seiler Schied, E. (Hrsg.) ; Kälin, S. (Hrsg.) ; Sengstag, C. (Hrsg.): *E-Learning – alltagstaugliche Innovation?* Münster : Waxmann Verlag, 2006, S. 58–67

[FBB+99] Fowler, M. ; Beck, K. ; Brant, J. ; Opdykeand, D. R. ; Roberts, D.: *Refactoring: Improving the Design of Existing Code*. Amsterdam, NL : Addison-Wesley Longman, 1999

[FBML98] Fischer, T. ; Biskup, H. ; Müller-Luschnat, G.: Begriffliche Grundlagen für Vorgehensmodelle. In: Kneuper, R. (Hrsg.): *Vorgehensmodelle für die betriebliche Anwendungsentwicklung*. Stuttgart : Teubner, 1998

[FH77] Flechsig, K. H. ; Haller, H. D.: *Einführung in didaktisches Handeln. Ein Lehrbuch für Einzel- und Gruppenarbeit*. 2. Auflage. Stuttgart : Klett, 1977

[FH96] Feenstra, R. C. ; Hanson, G. H.: Globalization, Outsourcing, and Wage Inequality. In: *American Economic Review* 86 (1996), S. 240–245

[Fie00] Fielding, R. T.: *Architectural Styles and the Design of Network-based Software Architectures*. Irvine, CA, USA, University of California, Diss., 2000

[Fil05] Filss, Christian: *Vergleichsmethoden für Vorgehensmodelle*. Dresden, Technische Universität Dresden, Diplomarbeit, 2005

[FL05] Fenn, J. ; Linden, A.: Understanding Gartner's Hype Cycles / Gartner Group. Stamford, CT, USA, 2005. – Forschungsbericht

[FM06] Feldstein, M. ; Masson, P.: *Unbolting the Chairs: Making Learning Management Systems More Flexible*. `http://elearnmag.org/subpage.cfm?section=tutorials\&article=22-1`. Version: 2006, Abruf: 30.06.2008

[Foe03] Foegen, M.: Architektur und Architekturmanagement – Modellierung von Architekturen und Architekturmanagement in der Softwareorganisation. In: *HMD – Praxis der Wirtschaftsinformatik* (2003), Nr. 232

[FP04] Floyd, C. ; Pape, B.: Softwareentwicklung als Wissensprojekt. In: Pape, B. (Hrsg.) ; Krause, D. (Hrsg.) ; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster : Waxmann Verlag, 2004, S. 389–409

[Fra94] Frank, U.: *Multiperspektivische Unternehmensmodellierung – Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung*. Oldenbourg, 1994

[FS85] Farrell, J. ; Saloner, G.: Standardization, Compatibility, and Innovation. In: *The Rand Journal of Economics* 16 (1985), S. 70–83

[G+08] Gerholz, K.-H. u. a.: Auswertung der Evaluation im koaLA Pilotbetrieb WS 07/08 / Universität Paderborn, Locomotion Projekt. Paderborn, 2008. – Forschungsbericht

[GAO95] Garlan, D. ; Allen, R. ; Ockerbloom, J.: Architectural Mismatch: Why Reuse is So Hard. In: *IEEE Software* 12 (1995), Nr. 6, S. 17–26

[GB98] Gavrilla, S. I. ; Barkley, J. F.: Formal Specification for Role Based Access Control User/Role and Role/Role Relationship Management. In: *Proceedings of 3rd ACM workshop on Role-based access control*. Fairfax, VA, USA : ACM Press, October 1998, S. 81–90

[GGA06] Gehtland, J. ; Galbraith, B. ; Almaer, D.: *AJAX: Eine Pragmatische Einführung in Web 2.0*. München, Wien : Carl Hanser Verlag, 2006

[GHJV96] Gamma, E. ; Helm, R. ; Johnson, R. ; Vlissides, J.: *Entwurfsmuster – Elemente wiederkehrender objektorientierter Software*. Bonn : Addison-Wesley, 1996

[GL05] Graf, S. ; List, B.: An Evaluation of Open Source E-Learning Platforms Stressing Adaption Issues. In: *Proceedings of 5th IEEE International Conference on Advanced Learning Technologies (ICALT05)*, 2005

[Gla94] Glasersfeld, E. von: Piagets konstruktivistisches Modell: Wissen und Lernen. In: Rusch, G. (Hrsg.) ; Schmidt, S. J. (Hrsg.): *Piaget und der radikale Konstruktivismus*. Frankfurt a. M. : Suhrkamp Verlag, 1994, S. 16–42

[GMS03] Gerhke, M. ; Meyer, M. ; Schäfer, W.: Eine Rahmenarchitektur für verteilte Lehr- und Lernsysteme / CampusSource e.V. Version: 2003. `http://www.campussource.de/projekte/rahmenarchitektur.html`. Hagen, Paderborn, 2003. – Forschungsbericht

[Göp07] Göpferich, S.: Textqualität steuern mit kontrollierter Sprache. Sprachstandard oder Kontrollmechanismus? In: *technische kommunikation* 29 (2007), Nr. 4

[GPS93] Garzotto, F. ; Paolini, P. ; Schwabe, D.: HDM: A model-based approach to hypertext application design. In: *ACM Transactions on Information Systems* 11 (1993), Nr. 1

[GR98] Greenberg, S. ; Roseman, M.: Using a Room Metaphor to Ease Transitions in Groupware / Department of Computer Science, University of Calgary. Alberta, Canada, 1998 (98/611/02). – Research Report

[GR07] Gossen, H. ; Roth, A.: Die Datenintegration zwischen HIS-LSF und universitären Lern- und Arbeitsplattformen – Ein Praxisbericht aus dem Locomotion Projekt / DS&OR-Lab, Universität Paderborn. Paderborn, 2007 (WP073333). – Arbeitspapier

[Gre03] Greiffenberg, S.: Methoden als Theorien der Wirtschaftsinformatik. In: *Wirtschaftsinformatik* Band 2 (2003), S. 947–968

[GS83] Gentner, D. ; Stevens, A.: *Mental Models*. Hillsdale, NJ, USA : Erlbaum, 1983

[GS96] Grob, L. ; Seufert, S.: Vorgehensmodelle bei der Entwicklung von CAL-Software / Institut für Wirtschaftsinformatik der Westfälischen Wilhelms-Universität Münster. Münster, 1996. – Forschungsbericht

[GTA05] Guicking, A. ; Tandler, P. ; Avgeriou, P.: Agilo: A Highly Flexible Groupware Framework. In: *Groupware: Design, Implementation, and Use*. Berlin-Heidelberg : Springer, 2005, S. 49–56

[Gül01] Güldenberg, S.: *Wissensmanagement und Wissenscontrolling in lernenden Organisationen: Ein systemtheoretischer Ansatz*. 3. Auflage. Wiesbaden : Deutscher Universitäts-Verlag, 2001

[Gut94] Gutzwiller, T. A.: *Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen*. Heidelberg : Physica Verlag, 1994

[Haa99] Haake, J. M.: Facilitating Orientation in Shared Hypermedia Workspaces. In: *Proceedings of Group '99*. ACM Press, 1999

[Hab80] Haberfellner, R.: Organisationsmethodik. In: Grochla, E. (Hrsg.): *Handwörterbuch der Organisation*. 2. Auflage. Stuttgart : Poeschel, 1980, S. 1701–1710

[Häf02] Häfele, H.: E-Learning Standards aus didaktischer Perspektive. In: Bachmann, G. (Hrsg.) ; Haefeli, O. (Hrsg.) ; Kindt, M. (Hrsg.) ; Gesellschaft für Medien in der Wissenschaft (Veranst.): *Campus 2002 – Die virtuelle Hochschule in der Konsolidierungsphase*. Münster : Waxmann Verlag, 2002

[Ham02] Hampel, T.: *Virtuelle Wissensräume – Ein Ansatz für die kooperative Wissensorganisation*, University of Paderborn, Diss., 2002

[Ham04] Hampel, T.: Access Rights – The Keys to Cooperative Work/Learning. In: Hicks, D.L. (Hrsg.): *Metainformatics. International Symposium, MIS*. Salzburg : Springer, 2004, S. 14–31

[Ham05a] Hambach, S.: Die Entwicklung von Bildungsangeboten mit E-Learning-Komponenten – Kategorisierung von Vorgehensmodellen. In: *Rostocker Informatik Berichte* Heft 29 (2005), S. 49–60

[Ham05b] Hammerschall, U.: *Verteilte Systeme und Anwendung*. Pearson Studium, 2005

[Har03] Harrer, A.: Software Engineering Methods for re-use of Components and Design in Educational Systems. In: *International Journal of Computers & Applications* 25 (2003), S. 17–23

[Has95] Hasebrook, J.: *Multimedia-Psychologie: Eine neue Perspektive menschlicher Kommunikation*. Heidelberg : Spektrum Verlag, 1995

[Hau02] Haun, M.: *Handbuch Wissensmanagement*. 1. Auflage. Berlin : Springer, 2002

[HD02] Harrer, A. ; Devedzic, V.: Design and Analysis Patterns in ITS Architectures. In: Moore, J. D. (Hrsg.) ; Redfield, C. (Hrsg.) ; Johnson, W. L. (Hrsg.): *Proceedings of the International Conference on Computers in Education (ICC02)*. Los Alamitos, CA, USA : IEEE Computer Society Press, 2002, S. 523

[Hei03] Heidenreich, M.: Die Debatte um die Wissensgesellschaft. In: Böschen, S. (Hrsg.) ; Schulz-Schaeffer, I. (Hrsg.): *Wissenschaft in der Wissensgesellschaft*. Wiesbaden : Westdeutscher Verlag, 2003

[Hel97] Hellmuth, Thomas W.: *Terminologiemanagement – Aspekte einer effizienten Kommunikation in der computerunterstützten Informationsverarbeitung*, Universität Konstanz, Diss., 1997

[Hel05] Held, A.: *Oracle 10g Hochverfügbarkeit – Ausfallsichere Datenbank mit RAC, Data Guard und Flashback*. München : Addison-Wesley, 2005

[Hes04] Hesse, F.W.: Eine kognitionspsychologische Analyse aktiven Lernens mit neuen Medien. In: Carstensen, D. (Hrsg.) ; Barrios, B. (Hrsg.): *Campus 2004 – Kommen die digitalen Medien an den Hochschulen in die Jahre?* Münster : Waxmann Verlag, 2004, S. 15–23

[HH05] Hampel, T. ; Heckmann, P.: Deliberative Handling of Knowledge Diversity – The Pyramid Discussion and Position-Commentary-Response Methods as Specific Views of Collaborative Virtual Knowledge Spaces. In: *Proceedings of SITE 2005*. Phoenix, AZ, USA, 2005

[Hil00] Hilliard, R.: An Overview of IEEE P1471, Recommended Practice for Architectural Description. In: *Delaware Valley Chapter*. Seattle, WA, USA : International Council of Systems Engineering (INCOSE), November 2000

[HKSE03] Hampel, T. ; Keil-Slawik, R. ; Essmann, B.: Jour Fixe – Structuring of Semantic Spaces as a Didactic Concept. In: Rosset, A. (Hrsg.) ; AACE (Veranst.): *Proceedings of E-Learn 2003* AACE, AACE Press, 2003, S. 225–232

[HM97] Höfling, S. ; Mandl, H.: *Lernen für die Zukunft – Lernen in der Zukunft – Wissensmanagement in der Bildung*. München : Hanns-Seidel-Stiftung, 1997

[HM05] Harrer, A. ; Martens, A.: Ansatz zur Definition einer Mustersprache für Lehr-/Lernsysteme. In: *DelFI 2003, 3. Deutsche e-Learning Fachtagung der Gesellschaft für Informatik*. Bonn : Köllen Druck + Verlag, 2005, S. 177–188

[Hop05a] Hoppe, G.: *Entwicklung strategischer Einsatzkonzepte für E-Learning an Hochschulen*. Lohmar, Köln : Eul Verlag, 2005

[Hop05b] Hoppe, G.: Organisatorische Verankerung von E-Learning in Hochschulen. In: Tavangarian, D. (Hrsg.) ; Nölting, K. (Hrsg.): *Auf zu neuen Ufern! E-Learning heute und morgen*. Münster : Waxmann Verlag, 2005, S. 237–246

[HPD05] Hauswirth, M. ; Podnar, I. ; Decker, S.: On P2P Collaboration Infrastructures. In: *Proceedings of the 14th IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises*. IEEE Computer Society, 2005, S. 66–71

[HPP07] Hüvelmeyer, J. ; Pohl, C. ; Postel, M.: Konzeption des integrierten Informationsmanagements von HIS-LSF/POS und weiteren IT-Systemen mit der CampusSource Engine / CampusSource e.V. Dortmund, 2007. – Forschungsbericht

[HR05] Hampel, T. ; Roth, A.: Rapid Development of Non-Monolithic CSCL-Applications – About the Benefits of Using a Prescribed Terminology in Web Programming. In: *Proceedings of E-Learning in Corporate Government, Healthcare, & Higher Education*. Vancouver, Canada, 2005, S. 2095–2102

[HRKK05] Hampel, T. ; Roth, A. ; Kahnwald, N. ; Köhler, T.: An Adaptable Platform for Evolving Communities of Practice. In: *International Reports on Socio-Informatics (IRSI). Special Issue Digital Communities* 2 (2005), Nr. 2

[HS90] Halasz, F. G. ; Schwartz, M.: The Dexter Hypertext Reference Model. In: *Proceedings of the NIST Hypertext Standardization Workshop*. Gaithersburg, ML, USA : NIST, 1990, S. 95–133

[HS00] Herzum, P. ; Sims, O.: *Business Component Factory: A Comprehensive Overview of Component-Based Development for the Enterprise*. Chichester, UK : John Wiley & Sons, 2000

[HSWH07] Höllrigl, T. ; Schell, F. ; Wenske, H. ; Hartenstein, H.: Föderatives und dienstorientiertes Identitätsmanagement im universitären Kontext. In: Gaedke, M. (Hrsg.) ; Borgeest, R. (Hrsg.): *Integriertes Informationsmanagement an Hochschulen. Quo vadis Universität 2.0?* Karlsruhe : Universitätsverlag Karlsruhe, 2007, S. 75–90

[IEE01] IEEE: IEEE P1484.1/D9, 2001-11-30 Draft Standard for Learning Technology – Learning Technology Systems Architecture (LTSA) / Learning Technology Standards Committee of the IEEE Computer Society. New York, NY, USA, 2001. – Forschungsbericht

[IM94] Ikeda, M. ; Mizoguchi, R.: FITS, A Framework for ITS – A computational model of tutoring. In: *Journal of Artifical Intelligence in Education* 5 (1994), Nr. 3, S. 319–348

[IMS03a] IMS: IMS Digital Repositories Interoperability – Core Functions Information Model / IMS Global Learning Consortium. Version: 2003. `http://www.imsglobal.org/digitalrepositories/`. 2003 (Version 1.0 Final Specification). – Forschungsbericht

[IMS03b] IMS: IMS Learning Design Information Model / IMS Global Learning Consortium. 2003. – Forschungsbericht

[ISB95] Isakowitz, T. ; Stohr, E. A. ; Balasubramaian, P.: RMM: A Methodology for Structured Hypermedia Design. In: *Communications of the ACM* 38 (1995), August, Nr. 8, S. 34–44

[Iss97] Issing, L. J. ; Issing, L. J. (Hrsg.) ; Klimsa, P. (Hrsg.): *Instruktionsdesign für Multimedia*. Weinheim : Beltz Psychologie Verlags Union, 1997

[Jän04a] Jänig, C.: *Wissensmanagement*. Berlin : Springer, 2004

[Jan04b] Janneck, M.: Themenzentrierte Interaktion und Projektmethode. In: Pape, B. (Hrsg.) ; Krause, D. (Hrsg.) ; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht*. Münster : Waxmann Verlag, 2004, S. 55–73

[JG07] Jaeger, M. ; Grützmacher, J.: Weniger Freiräume. Evaluation von Forschung und Lehre im Bolognaprozess. In: *Forschung & Lehre* (2007), Nr. 4, S. 214

[JHM07] Juling, W. ; Hartenstein, H. ; Maurer, H.: Karlsruher Integriertes Informations-Management KIM. In: Degkwitz, A. (Hrsg.) ; Schirmbacher, P. (Hrsg.): *Informationsinfrastrukturen im Wandel. Informationsmanagement an deutschen Universitäten*. Bad Honnef : Bock und Herchen Verlag, 2007, S. 116–129

[JIS08] JISC: *e-Learning frameworks and tools programme*. `http://www.jisc.ac.uk/whatwedo/programmes/elearning_framework.aspx`. Version: 2008, Abruf: 11.01.2008

[JL83] Johnson-Laird, P.: *Mental Models*. Cambridge, MA, USA : Harvard University Press, 1983

[JN05] Jacobson, I. ; Ng, P.-W.: *Aspect-Oriented Software Development with Use Cases*. Amsterdam, NL : Addison-Wesley Longman, 2005

[Kai01] Kaib, M.: *Enterprise Application Integration (EAI)*. Gastvortrag am 08.05.2001, Phillips Universität Marburg, 2001

[Kal03] Kaltenbaek, J.: *Berliner Beiträge zum E-Learning*. Bd. 1: *E-Learning und Blended-Learning in der betrieblichen Weiterbildung*. Berlin : Weißensee, 2003

[Kar95] Karlsson, E.-A.: *Software Reuse. A Holistic Approach*. Chichester, UK : John Wiley & Sons, 1995

[KdE00] KdEG: Arbeitsdokument der Kommissionsdienststellen: Memorandum über Lebenslanges Lernen / Kommission der Europäischen Gemeinschaften. Brüssel, 2000. – Forschungsbericht

[Keh01] Kehm, B.M.: Die Funktionserweiterung der Hochschulen durch lebenslanges Lernen – Reaktionen angesichts hochkomplexer Erwartungen. In: Kehm, B.M. (Hrsg.) ; Pasternack, P. (Hrsg.): *Hochschulentwicklung als Komplexitätsproblem. Fallstudien des Wandels*. Weinheim, Basel : Beltz Verlag, 2001, S. 122–143

[Kei07] Keil, Reinhard: Medienqualitäten beim eLearning: Vom Transport zur Transformation von Wissen. In: *BIBLIOTHEK Forschung und Praxis* 31 (2007), Nr. 1

[Ker01] Kerres, M.: *Multimediale und telemediale Lernumgebungen.* München, Wien : Oldenbourg, 2001

[Ker04a] Kerres, M.: Beyond Learning Platforms: Infrastructures for the Digital Campus. In: *Proceedings of 6th International Conference on New Educational Environments (ICNEE04)*, 2004

[Ker04b] Kerres, M.: Zur Integration digitaler Werkzeuge in die Hochschule. In: Kruse, E. (Hrsg.) ; Küchler, U. (Hrsg.) ; Kuhl, M. (Hrsg.): *Unbegrenztes Lernen – Lernen über Grenzen?* Münster : LIT Verlag, 2004

[Ker06] Kerres, M.: Potenziale von Web 2.0 nutzen. In: Hohenstein, A. (Hrsg.) ; Wilbers, K. (Hrsg.): *Handbuch E-Learning. Expertenwissen aus Wissenschaft und Praxis.* München : DWD, 2006

[Ker07] Kerres, M.: Microlearning as a challenge for instructional design. In: Hug, T. (Hrsg.) ; Lindner, M. (Hrsg.): *Didactics of Microlearning.* Münster : Waxmann Verlag, 2007

[KGHV04] Kirchhof, A. ; Gurzki, T. ; Hinderer, H. ; Vlachakis, J.: Was ist ein Portal? Definition und Einsatz von Unternehmensportalen / Fraunhofer IAO. 2004. – Forschungsbericht

[KIN00] Krogh, G. von ; Ichijo, K. ; Nonaka, I.: *Enabling Knowledge Creation: How to Unlock the Mistery of Tacit Knowledge and Release the Power of Innovation.* New York, NY, USA : Oxford University Press, 2000

[KL96] Kamlah, W. ; Lorenzen, P.: *Logische Propädeutik – Vorschule des vernünftigen Redens.* 3. Auflage. Stuttgart, Weimar : Metzler, 1996

[KL03] Klünter, D. ; Laser, J.: *OpenLDAP einsetzen. Grundlagen, Praxiseinsatz und Single Sign-On Mechanismen.* Heidelberg : Dpunkt, 2003

[Kla04] Klassen, M.: UDDI Revisited. In: *XML Magazin & Web Services* (2004), Nr. 2

[Kle02] Klein, M.: *Courseware Engineering – Ein Vorgehensmodell zur Erstellung von wiederverwendbaren, hypermedialen Kursen*, Universität Karlsruhe, Diss., 2002

[Kli93] Klimsa, P.: *Neue Medien und Weiterbildung: Anwendung und Nutzung in Lernprozessen der Weiterbildung.* Weinheim : Deutscher Studien Verlag, 1993

[KLT98] Kaindl, H. ; Lutz, B. ; Tippold, P.: *Methodik der Softwareentwicklung.* Braunschweig : Vieweg, 1998

[KM07] Kopp, B. ; Mandl, H.: Verteilte Kognition und ihre Bedeutung für E-Learning. In: Baumgartner, P. (Hrsg.) ; Reinmann, G. (Hrsg.): *Überwindung von Schranken durch E-Learning.* Innsbruck : Studienverlag, 2007, S. 101–119

[KNW03] Kerres, M. ; Nattland, A. ; Weckmann, H.-D.: Hybride Lernplattformen und integriertes Informationsmanagement an der Hochschule. In: Dittrich, K. (Hrsg.) ; Gesellschaft für Informatik (Veranst.): *Informatik 2003, Innovative Informatikanwendungen* Bd. 2 Gesellschaft für Informatik, 2003, S. 90–96

[Kol96] Kolb, B. ; Lippert, W. (Hrsg.): *Netvertising – Werbung auf dem Internet.* Düsseldorf, München : Metropolitan, 1996

[Kop01] Koper, R.: Modeling units of study from a pedagogical perspective – The pedagocial metamodel behind EML / Educational Technology Expertise Centre, Open University of the Netherlands. 2001. – Forschungsbericht

[Kös05] Köster, A.: Tagebücher für die Forschung. In: *duz – Das unabhängige Hochschulmagazin* (2005), Nr. 07/2005

[KS01] Klein, M. ; Stucky, W.: Ein Vorgehensmodell zur Erstellung virtueller Bildungsinhalte. In: *Wirtschaftsinformatik* 43 (2001), S. 35–45

[KS05a] Keil-Slawik, R.: Dienste-Infrastrukturen als Mittel der Wissensorganisation. In: Kerres, M. (Hrsg.) ; Keil-Slawik, R. (Hrsg.): *Hochschulen im digitalen Zeitalter: Innovationspotenziale und Strukturwandel*. Münster : Waxmann Verlag, 2005, S. 13–28

[KS05b] Kuszpa, M. ; Schern, E.: Mobile Learning – Modetrend oder wesentlicher Bestandteil lebenslangen Lernens? / FernUniversität in Hagen. 2005. – Forschungsbericht

[Kub04] Kubicek, H.: Organisatorische Einbettung von E-Learning an deutschen Hochschulen / Institut für Informationsmanagement. Bremen, 2004. – Forschungsbericht

[KZ02] Krammer, A. ; Zaha, J. M.: Komponentenfindung in monolithischen betrieblichen Anwendungssystemen / Lehrstuhl für Wirtschaftsinformatik II, Universität Augsburg. 2002. – Forschungsbericht

[LAGS99] Lee, S. D. ; Armitage, S. ; Groves, P. ; Stephens, C.: Online Teaching: MUDs, MOOs, WOOs and IRC / JISC Technology Applications Programme. 1999. – Forschungsbericht

[Lam74] Lampson, B. W.: Protection. In: *ACM SIGOPS Operating Systems Review* 8 (1974), Januar, Nr. 1, S. 18–24

[Lam07] Lamb, B.: Dr. Mashup – or, Why Educators Should Learn to Stop Worrying and Love the Remix. In: *Educause Review* (2007), Nr. July/August, S. 12–24

[Lan66] Lane, R.E.: The Decline of Politics and Ideology in a Knowledge Society. In: *American Sociological Review* 31 (1966), S. 650

[Leh98a] Lehmann, F. R.: Aktuelles Schlagwort: Normsprache. In: *Informatik Spektrum* 21 (1998), S. 366–367

[Leh98b] Lehmann, Frank R.: Materialsprachliche Entwicklung von Workflow-Management-Anwendungen. In: *Informationssystem-Architekturen, Rundbrief des GI-Fachausschusses 5.2* 5 Jg. (1998), Nr. 2, S. 88–94

[Lei01] Leidhold, W.: Die Wissensgesellschaft / Forschungszentrum für Politische Wissenschaft und Europäische Fragen. Version: 2001. `http://www.politik.uni-koeln.de/leidhold/pdf/Wissensgesellschaft.pdf`. Köln, 2001. – Forschungsbericht

[Les03] Leslie, S.: *Some Uses of Blogs in Education*. `http://www.edtechpost.ca/gems/matrix2.gif`. Version: 2003, Abruf: 25.09.2007

[Lin04] Lindner, R.: Spezifikationen, Normen und Standards für Lernmaterialien. In: Haake, J. (Hrsg.) ; Schwabe, G. (Hrsg.) ; Wessner, M. (Hrsg.): *CSCL-Kompendium*. München, Wien : Oldenbourg, 2004, S. 341–356

[Lit07] Little, J. K.: Podcasting: A Teaching with Technology White Paper. In: *Educause Connect* (2007), Juni

[LL04] Ley, T. ; Lindstaedt, S.: Integrating Knowledge Management and eLearning: A Competency Perspective. In: Reich, S. (Hrsg.): *Digital Content Engineering – Content Plattformen in Theorie und Praxis*. Linz : Trauner Verlag, 2004, S. 42–56

[LMW05] Lankes, J. ; Matthes, F. ; Wittenburg, A.: Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000. In: Liggesmeyer, P. (Hrsg.) ; Pohl, K. (Hrsg.) ; Goedicke, M. (Hrsg.): *Software Engineering 2005, Fachtagung des GI-Fachbereichs Softwaretechnik* Bd. 64. Bonn : Köllen Druck + Verlag, 2005, S. 43–54

[Lor95] Lorenz, K.: Methode. In: Mittelstrass, J. (Hrsg.): *Enzyklopädie Philosophie und Wissenschaftstheorie* Bd. 2. Mannheim : Bibliographisches Institut, 1995, S. 876–879

[Lor00] Lorenzen, P.: *Lehrbuch der konstruktiven Wissenschaftstheorie*. Stuttgart, Weimar : J. B. Metzler et al., 2000 (Metzler Reprint)

[LS04] Ley, T. ; Schachner, W.: eLearning Check: Sind Sie bereit für eLearning? / Know-Center Graz. Graz, 07/2004 2004. – Ergebnisbericht

[Luc96] Luchner: DBV-OSI-II: Projektericht Realisierungsphase. In: *BIBLIOTHEKS-DIENST* (1996), Nr. 5

[Luh58] Luhn, H. P.: Automatic Creation of Literature Abstracts. In: *IBM Journal of Research and Development* 2 (1958), Nr. 2, S. 159–165

[LW05] Leszczensky, Michael ; Wolter, Andrä: Der Bologna-Prozess im Spiegel der HIS-Hochschulforschung / HIS-Hochschul-Informations-System GmbH. Hannover, 2005. – Forschungsbericht

[MA07] Meister, J. ; Appelrath, H.-J.: Produktgetriebene Entwicklung von Softwareproduktlinien. In: *Wirtschaftsinformatik* 49 (2007), Nr. 3, S. 180–187

[Man04] Mandl, H.: E-Learning – Trends und zukünftige Entwicklungen. In: Rebensburg, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens; 2. Workshop GML 2004*. Norderstedt : BoD, 2004, S. 17–29

[Mar08] Marx, W.: *Evaluation der koaLA-Plattform*. Paderborn, Universität Paderborn, Diplomarbeit, 2008

[Mas05] Masak, D.: *Moderne Enterprise Architekturen*. Berlin, Heidelberg : Springer, 2005

[Mat96] Mattson, M.: Object-Oriented Frameworks – A survey of methodological issues. In: *Proceedings of the 1996 Triennal IFAC World Congress*. San Francisco, California, USA : Elsevier Science, 1996, S. 259–264

[McD99] McDermott, R.: Nurturing three-dimentional communities of practice – how to get the most out of human networks. In: *Knowledge Management Review* (1999), Nr. 11/12, S. 26–29

[MG05] Messerschmidt, R. ; Grebe, R.: Zwischen visionärer Euphorie und praktischer Ernüchterung. In: *QUEM-Report Schriften zur beruflichen Weiterbildung* (2005), Nr. 91

[Mit06] Mitchell, P.: Wikis in education. In: Klobas, J. (Hrsg.): *Wikis: Tools for Information Work and Collaboration*. Oxford, UK : Chandos Publishing, 2006, S. 119–147

[MJ05] Malys, B. ; Juhnke, N.: Hochschulweite Integration von eLearning an der BTU Cottbus. In: Fellbaum, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens*. Aachen : Shaker Verlag, 2005, S. 13–24

[MK01] Mandl, H. ; Krause, U. M.: Lernkompetenz für die Wissensgesellschaft (Forschungsbericht Nr. 145) / Ludwig-Maximilians-Univesität, Lehrstuhl für Empirische Pädagogik und Pädagogische Psychologie. München, 2001. – Forschungsbericht

[MMF07] Majer, F. ; Meinecke, J. ; Freudenstein, P.: Die Landkarte – Rahmenwerk zur Unterstützung von Evolution und Betrieb serviceorientierter Architekturen. In: Gaedke, M. (Hrsg.) ; Borgeest, R. (Hrsg.): *Integriertes Informationsmanagement an Hochschulen. Quo vadis Universität 2.0?* Karlsruhe : Universitätsverlag Karlsruhe, 2007, S. 19–35

[MN96] Moser, S. ; Nierstrasz, O.: The Effect of Object-Oriented Frameworks on Developer Productivity. In: *Computer* 29 (1996), Nr. 9, S. 45–51

[Möl05] Möller, E.: *Die heimliche Medienrevolution – Wie Weblogs, Wikis und freie Software die Welt verändern.* Hannover : Heise, 2005

[Moo05] Moog, H.: *IT-Dienste an Universitäten und Fachhochschulen – Reorganisation und Ressourcenplanung der hochschulweiten IT-Versorgung.* Hannover : HIS-Hochschul-Informations-System GmbH, 2005 (Hochschulplanung Band 178)

[Mos06] Mosel, S.: Self Directed Learning With Personal Publishing And Microcontent. In: Hug, T. (Hrsg.) ; Lindner, M. (Hrsg.) ; Bruck, P. A. (Hrsg.): *Proceedings of Microlearning 2005. Learning & Working in New Media.* Insbruck : Innsbruck University Press, 2006, S. 99–109

[MTH98] Mühlenbrock, M. ; Tewissen, F. ; Hoppe, U.: A Framework System for Intelligent Support in Open Distributed Learning Environments. In: *Journal of Artifical Intelligence in Education* (1998), Nr. 9, S. 256–274

[Nag03] Nagl, M.: EAI heißt insbesondere Integration: Schwierige Probleme und die Rolle technischer Hilfsmittel. In: *Online 2003 Messe Düsseldorf – Web Services: Schlüssel für eBusiness Integration.* Online GmbH, 2003

[NBS+99] Nagl, M. ; Balzert, H. ; Six, H. W. ; Schäfer, W. ; Kelter, U.: Softwaretechnische Anforderungen an multimediale Lehr- und Lernsysteme / Forschergruppe SofTec NRW. 1999. – Forschungsbericht

[Nie01] Niegemann, H.: *Neue Lernmedien. Konzipieren, entwickeln, einsetzen.* Bern : Huber, 2001

[Nie04] Niegemann, H.: *Kompendium E-Learning.* Berlin : Springer, 2004

[NIS02] NISO: Information Retrieval (Z39.50): Application Service Definition and Protocol Specification / National Information Standards Organization. Bethesda, ML, USA, 2002. – Forschungsbericht

[Non91] Nonaka, I.: The Knowledge-Creating Company. In: *Havard Business Review* (1991), 11/12, S. 96–104

[Non94] Nonnemacher, M. G.: *Schriften zum Controlling.* Bd. Band 14: *Informationsmodellierung unter Nutzung von Referenzmodellen: Die Nutzung von Referenzmodellen zur Implementierung industriebetrieblicher Informationssysteme*. Frankfurt a. M. : Peter Lang Verlag, 1994

[Now05] Nowaczyk, O.: *Explorationen: Ein Ansatz zur Entwicklung hochgradig interaktiver Lernbausteine.* Paderborn : Bonifatius, 2005

[NRP00] North, K. ; Romhardt, K. ; Probst, G.: Wissensgemeinschaften – Keimzellen lebendigen Wissensmanagements. In: *io management* 8 (2000), Nr. 7

[NS99] Noack, J. ; Schienmann, Bruno: Objektorientierte Vorgehensmodelle im Vergleich. In: *Informatik Spektrum* 22 (1999), S. 166–180

[NT97] Nonaka, I. ; Takeuchi, H.: *Die Organisation des Wissens – Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen.* Frankfurt, New York : Campus, 1997

[NT01] Nonaka, I. ; Teece, D.J.: *Managing Industrial Knowledge: New Perspectives on Knowledge-Based Firms*. Sage Publications, 2001

[NTK01] Nonaka, I. ; Toyama, R. ; Konno, N.: SECI, Ba and Leadershp: A Unified Model of Dynamic Knowledge Creation. In: Nonaka, I. (Hrsg.) ; Teece, D.J. (Hrsg.): *Managing Industrial Knowledge: Creation, Transfer and Utilization*. London : Sage Publications, 2001

[OBD04] O'Murchu, I. ; Breslin, J. G. ; Decker, S.: Online Social and Business Networking Communities / Digital Enterprise Research Institute, National University of Ireland. Galway, 2004. – Forschungsbericht

[Oes04] Oestereich, B.: *Objektorientierte Softwareentwicklung – Analyse und Design mit der UML 2.0*. 6. Auflage. München : Oldenbourg, 2004

[O'H07] O'Hear, S.: *e-learning 2.0 – how Web technologies are shaping education*. `www.readwriteweb.com/archives/e-learning_20.php`. Version: 2007, Abruf: 30.06.2008

[OKI06] OKI: Repositories Win Through O.K.I. / Open Knowledge Initiative. 2006. – Forschungsbericht

[O'R05] O'Reilly, T.: *Web 2.0 – Design Patterns and Business Models for the Next Generation of Software*. `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html`. Version: 2005, Abruf: 30.06.2008

[Ora02] Oravec, J. A.: Bookmarking the world: Weblog applications in education. In: *Journal of Adolescent and Adult Literacy* 45 (2002)

[Ora03] Oravec, J. A.: Weblogs as an Emerging Genre in Higher Education. In: *Journal of Computing in Higher Education* 14 (2003)

[Ort93] Ortner, E.: Informationsverarbeitung und Sprachkritik – Ein komplementärer Integrationsbereich für Benutzer, Management und Systeme / Universität Konstanz. Konstanz, 1993 (Bericht 17-93). – Forschungsbericht

[Ort95] Ortner, E.: Elemente einer methodenneutralen Konstruktionssprache für Informationssysteme. In: *Informatik Forschung und Entwicklung* 10 (1995), S. 148–160

[Ort97] Ortner, E.: *Methodenneutraler Fachentwurf*. Stuttgart, Leipzig : Teubner, 1997 (Wirtschaftsinformatik)

[Ort98] Ortner, E.: Normsprachliche Entwicklung von Informationssystemen. In: Pohl, K. (Hrsg.) ; Schürr, A. (Hrsg.) ; Vossen, G. (Hrsg.): *Modellierung* Bd. 9. Münster : CEUR-WS.org, 1998 (CEUR Workshop Proceedings)

[Ort00] Ortner, E.: Terminologiebasierte, komponentenorientierte Entwicklung von Anwendungssystemen. In: Flatscher, R.G. (Hrsg.) ; Turowski, K. (Hrsg.): *Tagungsband 2. Workshop komponentenorientierter betrieblicher Anwendungssysteme*. Wien, 2000, S. 1–20

[OWZ+05] Orlich, S. ; Werner, M. ; Zeitz, P. ; Apitzsch, J. ; Raith, C. ; Lietz, G.: Sichere Integration von E-Government-Anwendungen. In: *E-Government-Handbuch*. Bundesamt für Sicherheit in der Informationstechnik (BSI), 2005

[Paw01] Pawlowski, J.M.: *Das Essener-Lern-Modell (ELM): Ein Vorgehensmodell zur Entwicklung computerunterstützter Lernumgebungen*, Universität Essen, Diss., 2001

[PM02] Pfister, H. R. ; Mühlenpfordt, M.: Supporting discourse in a synchronous learning environment: The learning protocol approach. In: *Proceedings of CSCL 2002*. Hillsdale, NJ, USA : Erlbaum, 2002, S. 581–589

[Poh07] Pohl, K.: *Requirements Engineering – Grundlagen, Prinzipien, Techniken.* 1. Heidelberg : Dpunkt, 2007

[Pol66] Polanyi, M.: *The Tacit Dimension.* London, UK : Routlege & Kegan Paul, 1966

[Pol85] Polanyi, M.: *Implizites Wissen.* Frankfurt a. M. : Suhrkamp Verlag, 1985

[Pos07] Postel, M.: CampusSource Engine: Integrationslösung für IT-Systeme. In: *CHECK.point eLearning* (2007), 09/2007

[PR04] Pape, B. ; Rolf, A.: Integrierte Organisations- und Softwareentwicklung für kooperative Lernplattformen in der Hochschule. In: Pape, B. (Hrsg.) ; Krause, D. (Hrsg.) ; Oberquelle, H. (Hrsg.): *Wissensprojekte – Gemeinschaftliches Lernen aus didaktischer, softwaretechnischer und organisatorischer Sicht.* Münster : Waxmann Verlag, 2004, S. 287–310

[PRH06] Pauleickhoff, F. ; Roth, A. ; Hampel, T.: Structuring Organizational Knowledge in Virtual Knowledge Rooms at Philips Semiconductors. In: Tochtermann, K. (Hrsg.) ; Maurer, H. (Hrsg.): *Journal of Universal Computer Science, Special Issue on Knowledge Management* Bd. 12. Graz, 2006, S. 367–374

[Pro87] Probst, G.: *Selbst-Organisation: Ordnungsprozesse in sozialen Systemen aus ganzheitlicher Sicht.* Berlin, Hamburg : Parey, 1987

[PRR03] Probst, G. ; Raub, S. ; Romhardt, K.: *Wissen managen. Wie Unternehmen ihre wertvollste Ressource optimal nutzen.* 4. Auflage. Wiesbaden : Gabler, 2003

[PSBWW99] Pfister, H.R. ; Schuckmann, C. ; Beck-Wilson, J. ; Wessner, M.: The Metaphor of Virtual Rooms in the Cooperative Learning Environment CLear. In: *Lecture Notes in Computer Science* 1370 (1999), S. 107–113

[Pus04] Puschmann, T.: *Prozessportale – Architektur zur Vernetzung mit Kunden und Lieferanten.* Berlin, Heidelberg : Springer, 2004

[Rad05] Rademacher, M.: CSCL mit Comets. Kooperatives Lernen an freien Online-Tutorials. In: Fellbaum, K. (Hrsg.): *Grundfragen multimedialen Lehrens und Lernens.* Aachen : Shaker Verlag, 2005, S. 187–197

[Rau04] Rauner, F.: *Praktisches Wissen und berufliche Handlungskompetenz.* Bremen : Institut Technik und Bildung, Universität Bremen, 2004 (ITB - Forschungsberichte 14/2004)

[RB04] Rinn, U. ; Bett, K.: Revolutioniert das "E" die Lernszenarien an deutschen Hochschulen? – Eine empirische Studie im Rahmen des Bundesförderprogramms "Neue Medien in der Bildung". In: Carstensen, D. (Hrsg.) ; Barrios, B. (Hrsg.): *Campus 2004 – Kommen die digitalen Medien an den Hochschulen in die Jahre?* Münster : Waxmann Verlag, 2004, S. 428–437

[Reg04] Reglin, T.: Zwischen Effizienzversprechen und Sachzwang – Auf dem Weg zu einer systematischen Zielreflexion im eLearning. In: *E-Learning: Theorien und betriebliche Praxis. Fallstudien auf der betrieblichen Bildungsarbeit.* Köln : Institut der deutschen Wirtschaft Köln, 2004, S. 9–34

[Rei05a] Reinmann, G.: *Blended Learning in der Lehrerbildung. Grundlagen für die Konzeption innovativer Lernumgebungen.* Lengerich : Pabst Science Publishers, 2005

[Rei05b] Reinmann, G.: Wissensmanagement und Medienbildung – neue Spannungsverhältnisse und Herausforderungen. In: *MedienPädagogik* (2005), Nr. 5

[Rei07] Reinmann, G.: Bologna in Zeiten des Web 2.0. Assessment als Gestaltungsfaktor / Universität Augsburg, Philosophisch-Sozialwissenschaftliche Fakultät. Augsburg, 2007 (16). – Arbeitsbericht

[Ren03] Renzl, B.: *Wissensbasierte Interaktion. Selbst-evolvierende Wissensströme in Unternehmen.* Deutscher Universitäts-Verlag, 2003

[Rep02] Reppert, I.: E-Learning: Versuchen wir es mal mit Blended Learning. In: *Financial Times Deutschland* (2002), Nr. 08.02.

[RG93] Roseman, M. ; Greenberg, S.: Building Flexible Groupware Through Open Protocols. In: *Proceedings of the conference on Organizational computing systems.* New York, NY, USA : ACM Press, 1993, S. 279 – 288

[RH04] Rubart, J. ; Hampel, T.: Structuring Cooperative Spaces: From Static Templates to Self-Organization. In: Hicks, D. L. (Hrsg.): *Metainformatics. International Symposium, MIS 2003.* Berlin, Heidelberg : Springer, 2004, S. 119–125

[RH05] Roth, A. ; Hampel, T.: Konfigurierbare Softwarekomponenten zur Unterstützung dynamischer Lern- und Arbeitsumgebungen für virtuelle Gemeinschaften. In: *Tagungsband zum Workshop GeNeMe 2005: Gemeinschaften in Neuen Medien.* Dresden, 2005, S. 373–384

[RH06] Roth, A. ; Hoppe, G.: Approaching Heterogeneity within Educational Technology - Concepts, Risks, and Success Factors of Launching Service-oriented Architectures. In: Pearson, E. (Hrsg.) ; Bohman, P. (Hrsg.) ; AACE (Veranst.): *Proceedings of Ed-Media 2006, 26.06.-30.-06.05, Orlando, Florida.* Orlando, FL, USA, 2006, S. 2228–2233

[RH07] Roth, A. ; Hoppe, G.: Technologische und organisatorische Aspekte von dienstorientierten E-Learning-Infrastrukturen an Hochschulen. In: Breitner, M.H. (Hrsg.) ; Bruns, B. (Hrsg.) ; Lehner, F. (Hrsg.): *Neue Trends im E-Learning – Aspekte der Betriebswirtschaftslehre und Informatik.* Heidelberg : Physica-Verlag, 2007

[RHG05] Roth, P. ; Hinz, D. ; Gast, C.: Integration von Learning Management in bestehende IT-Infrastrukturen. In: *Information Management & Consulting* 20 (2005), Nr. 1, S. 69–76

[RHQ+04] Rupp, C. ; Hahn, J. ; Queins, S. ; Jeckle, M. ; Zengler, B.: *UML 2 – glasklar.* München, Wien : Hanser, 2004

[RHS05] Roth, A. ; Hampel, T. ; Suhl, L.: Von serverzentrierten Lernobjekten zu kooperativen Wissensobjekten – Ein wissensbasierter Integrationsansatz verteilter Lernplattformen am Beispiel des virtuellen Studienfachs Virtual Operations Research/Management Science. In: Fellbaum, K. (Hrsg.): *Tagungsband des 3. Workshops Grundfragen multimedialen Lehrens und Lernens.* Aachen : Shaker Verlag, 2005, S. 177–186

[RHS06] Roth, A. ; Hampel, T. ; Strauch, T.: Supporting the Business Game of a TV Production by LMS - About the Implementation of Highly Configurable Submission Rooms for Various Learning Scenarios. In: Pearson, E. (Hrsg.) ; Bohman, P. (Hrsg.) ; AACE (Veranst.): *Proceedings of Ed-Media 2006, 26.06.-30.-06.05, Orlando, Florida.* Orlando, FL, USA, 2006, S. 818–823

[Ric06] Richardson, W.: *Blogs, wikis, podcasts, and other powerful web tools for classrooms.* Thousand Oaks Calif. : Corwin Press, 2006

[Roe02] Roehl, H.: *Organisation des Wissens – Anleitung zur Gestaltung.* Stuttgart : Klett-Cotta, 2002

[Rog04] Rogner, L.: *Weiterbildung in virtuellen Lernumgebungen – Grundlage, Entwicklung und Evaluation eines Konzepts*, Universität Paderborn, Diss., 2004

[Rom02] Romhardt, K.: *Wissensgemeinschaften – Orte lebendigen Wissensmanagements.* Zürich : Versus Verlag, 2002

[RR99] Robertson, S. ; Robertson, J.: *Mastering the Requirements Process*. 1. Auflage. Amsterdam, NL : Addison-Wesley Longman, 1999

[RR03] Reinmann-Rothmeier, G.: *Didaktische Innovationen durch Blended Learning. Leitlinien anhand eines Beispiels aus der Hochschule*. Bern : Huber, 2003

[RRM97] Reinmann-Rothmeier, G. ; Mandl, H.: Kompetenzen für das Leben in einer Wissensgesellschaft. In: Höfling, S. (Hrsg.) ; Mandl, H. (Hrsg.): *Lernen für die Zukunft – Lernen in der Zukunft*. München : Hanns-Seidel-Stiftung, 1997, S. 97–107

[RRM01] Reinmann-Rothmeier, G. ; Mandl, H.: *Virtuelle Seminare in der Hochschule und Weiterbildung*. 1. Auflage. Bern : Huber, 2001

[RS05] Roth, A. ; Suhl, L.: Plattformübergreifende Architekturen in föderativen E-Learning-Umgebungen. In: Breitner, M.H. (Hrsg.) ; Hoppe, G. (Hrsg.): *E-Learning – Einsatzkonzepte und Geschäftsmodelle*. Heidelberg : Physica-Verlag, 2005, S. 143–152

[RSLC95] Rossi, G. ; Schwabe, D. ; Lucena, C. J. P. ; Cowan, D. D.: A Object-Oriented Model for Designing the Human-Computer Interface Of Hypermedia Applications. In: Fraisse, F. (Hrsg.) ; Garzotto, F. (Hrsg.) ; Isakowitz, T. (Hrsg.): *Hypermedia Design: Proceedings of the International Workshop on Hypermedia Design*. Berlin, Heidelberg : Springer, 1995

[RSS04] Roth, A. ; Scholz, M. ; Suhl, L.: Webbasiertes Lehrveranstaltungsmanagement – Effizienzsteigerung durch horizontale Integration von Lehr-/Lerntechnologien. In: Carstensen, D. (Hrsg.) ; Barrios, B. (Hrsg.) ; Gesellschaft für Medien in der Wissenschaft (Veranst.): *Campus 2004 – Kommen die digitalen Medien an den Hochschulen in die Jahre?* Münster : Waxmann Verlag, 2004, S. 438–449

[Rup07] Rupp, C.: *Requirements-Engineering und -Management. Professionelle, iterative Anforderungsanalyse für die Praxis*. 4. München : Carl Hanser Verlag, 2007

[S+00] Sherlund, B. u. a.: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems / Institute of Electric and Electronics Engineers, Inc. Computer Society. New York, NY, USA, 2000 (Std 1471-2000). – Description

[S+04] Stepping, M. u. a.: CampusSourceEngine – Die Schnittstelle von e-Learning Systemem zum HIS-GX System der HIS GmbH / CampusSource. Hagen, Hannover, 2004. – Forschungsbericht

[S+05] Simon, B. u. a.: A Simple Query Interface Specification for Learning Repositories / Europäisches Komitee für Normung. 2005 (CWA 15454). – CEN Workshop Agreement

[SAP02] SAP: Weltweite Kundennähe gewährleisten und Technologietrends in SAP-Lösungen umsetzen / SAP AG. Walldorf : SAP, 2002. – Forschungsbericht

[Sch92] Schnupp, P.: *Handbuch der Informatik*. Bd. 10.1: *Hypertext*. München : Oldenbourg, 1992

[Sch95] Schienmann, B.: Fachentwurf mit TAOS. Ein Terminologie-basierter Ansatz für die Objektorientierte Spezifikation. In: *Berichte der Informationswissenschaft der Universität Konstanz* (1995)

[Sch96] Schachtl, S.: Requirements for Controlled German in Industrial Applications. In: *Proceedings of the First International Workshop on Controlled Language Applications (CLAW 96)*. Leuven, BL, 1996, S. 143–149

[Sch97a] Schienmann, B.: *Teubner-Texte zur Informatik*. Bd. Bd. 20: *Objektorientierter Fachentwurf: ein terminologiebasierter Ansatz für die Konstruktion von Anwendungssystemen*. Stuttgart, Leipzig : B. G. Teubner Verlagsgesellschaft, 1997

[Sch97b] Schulmeister, R.: *Grundlagen hypermedialer Lernsysteme*. 2. Auflage. München, Wien : Oldenbourg, 1997

[Sch00a] Schindler, Martin: *Wirtschaftsinformatik*. Bd. 32: *Wissensmanagement in der Projektabwicklung*. Lohmar, Köln : Josef Eul Verlag, 2000

[Sch00b] Schmidt, M. P.: *Knowledge Communities*. München : Addison-Wesley, 2000

[Sch01a] Schryen, G.: *Komponentenorientierte Softwareentwicklung in Softwareunternehmen*. 1. Auflage. Wiesbaden : Deutscher Universitäts-Verlag, 2001

[Sch01b] Schulmeister, R.: *Virtuelle Universität Virtuelles Lernen*. München : Oldenbourg, 2001

[Sch02] Schienmann, B.: *Kontinuierliches Anforderungsmanagement*. 1. München : Addison-Wesley, 2002

[Sch04a] Scharsich, A.: Richtlinien über die Förderung der Entwicklung und Erprobung von Maßnahmen der Strukturentwicklung zur Etablierung von eLearning in der Hochschullehre im Rahmen des Förderschwerpunkts Neue Medien in der Bildung / Bundesministerium für Bildung und Forschung. 2004. – Forschungsbericht

[Sch04b] Schmitz, S.: E-Learning für alle? Wie lässt sich Diversität in Technik umsetzen? In: Carstensen, D. (Hrsg.) ; Barrios, B. (Hrsg.): *Campus 2004 – Kommen die digitalen Medien an den Hochschulen in die Jahre?* Münster : Waxmann Verlag, 2004, S. 123–133

[Sch04c] *Kapitel* Einleitung. In: Schönherr, M.: *Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen*. Berlin : Gito, 2004

[Sch05] Schramm, J.: SHRM workplae forecast. In: *Workplace Visions* (2005), Nr. Nr. 3

[Sch06] Schmidt, J.: Social Software. Onlinegestütztes Informations-, Identitäts- und Beziehungsmanagement. In: *Forschungsjournal Neue Soziale Bewegungen* (2006), Nr. 02.06., S. 37–46

[Sch08] Schmid, K.: Gleichheit in Vielfalt: Produktlinien in der industriellen Softwareentwicklung. In: *ix* (2008), Nr. 5, S. 110–114

[SE04] Seufert, S. ; Euler, D.: Nachhaltigkeit von eLearning-Innovationen. Ergebnisse einer Delphi-Studie / SCIL St. Gallen. 2004. – Forschungsbericht

[SF01] Süss, C. ; Freitag, B.: IFIS Report 2001/03: Learning Material Markup Language LMML / Institut für Informationssysteme und Softwaretechnik, Universität Passau. 2001. – Forschungsbericht

[SH99] Stahlknecht, P. ; Hasenkamp, U.: *Einführung in die Wirtschaftsinformatik*. 9. Auflage. Berlin : Springer, 1999

[SHB04] Schmidt, C. ; Hampel, T. ; Bopp, T.: We've Got Mail! – Eine neue Qualität der Integration von Nachrichtendiensten in die kooperative Wissensorganisation. In: Engels, G. (Hrsg.) ; Seehusen, S. (Hrsg.): *DelFI 2004, Die 2. E-Learning Fachtagung Informatik*, 2004 (Lecture Notes in Informatics), S. 211–222

[Shn83] Shneiderman, B.: Direct Manipulation: A Step Beyond Programming Languages. In: *IEEE Computer* 8 (1983), Nr. 16, S. 57–69

[SIF07] SIFA: Schools Interoperability Framework Implementation Specification, V.2.1 / Schools Interoperability Framework Association. Washington, DC, USA, 2007. – Specification

[SK03] Schwinger, W. ; Koch, N.: Modellierung von Web-Anwendungen. In: Kappel, G. (Hrsg.) ; Pröll, B. (Hrsg.) ; Reich, S. (Hrsg.) ; Retschitzegger, W. (Hrsg.): *Web-Engineering. Systematische Entwicklung von Web-Anwendungen*. Heidelberg : Dpunkt, 2003, Kapitel 3, S. 49–76

[SKRS01] Severing, E. ; Keller, C. ; Reglin, T. ; Spies, J.: *Betriebliche Bildung via Internet. Konzeption, Umsetzung und Bewertung. Eine Einführung für Praktiker.* 1. Auflage. Bern : Huber, 2001

[SM02] Seufert, S. ; Mayr, P.: *Fachlexikon e-learning. Manager Seminare.* Bonn : Gerhard May Verlag, 2002

[SMD04] Simon, B. ; Massart, D. ; Duval, E.: Simple Query Interface Specification / CEN/ISSS Workshop on Learning Technologies. 2004. – Forschungsbericht

[SMVAD05] Simon, B. ; Massart, D. ; Van Assche, F. ; Duval, E.: Authentication and Session Management / CEN/ISSS Workshop on Learning Technologies. 2005. – Forschungsbericht

[Som07] Sommerville, I.: *Software Engineering.* 8. Auflage. Pearson Studium, 2007

[SOP01] SOPHISTen: Die psychotherapeutischen Grundlagen des SOPHISTEN-Regelwerks für die natürlichsprachliche Methode / SOPHIST GmbH. Nürnberg, 2001. – Forschungsbericht

[SP97] Szyperski, C. ; Pfister, C.: Workshop on Component-Oriented Programming, Summary. In: Mühlhäuser, M. (Hrsg.): *Special Issues in Object-Oriented Programming – ECOOP96 Workshop Reader.* Heidelberg : Dpunkt, 1997

[SPM99] Schwabe, D. ; Pontes, R. ; Moura, I.: OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW. In: *ACM SigWEB Newsletter* 8 (1999), 6, Nr. 2

[Spo07] Spool, J. M.: Web 2.0: The Power Behind the Hype. In: *User Interface Engineering* (2007), 08/2007

[SR01] Seiler, T. B. ; Reinmann, G.: Der Wissensbegriff im Wissensmanagement: Eine strukturgenetische Sicht. In: Reinmann, G. (Hrsg.) ; Mandl, H. (Hrsg.): *Psychologie des Wissensmanagements: Perspektiven, Theorien und Methoden.* Göttingen : Hogrefe, 2001, S. 11–23

[SS01] Schümmer, J. ; Schuckmann, C.: Synchrone Softwarearchitekturen. In: Schwabe, G. (Hrsg.) ; Streitz, N. A. (Hrsg.) ; Unland, R. (Hrsg.): *CSCW-Kompendium: Lehr- und Handbuch zum computergestützten kooperativen Arbeiten.* Heidelberg : Springer, 2001, S. 297–307

[SSB04] Sauter, A. M. ; Sauter, W. ; Bender, H.: *Blended Learning. Effiziente Integration von E-Learning und Präsenztraining.* 2. Auflage. Neuwied, Kriftel : Luchterhand, 2004

[SSW+06] Schaeper, H. ; Schramm, M. ; Weiland, M. ; Kraft, S. ; Wolter, A.: International vergleichende Studie zur Teilnahme an Hochschulweiterbildung / HIS-Hochschul-Informations-System GmbH. Hannover, 2006. – Forschungsbericht

[Ste00] Steinmetz, R.: *Multimedia-Technologie: Grundlagen, Komponenten und Systeme.* 3. Auflage. Berlin, Heidelberg : Springer, 2000

[Str07] Ströhlein, G.: Mobile Learning Using Mobiles: Hype or Tripe? In: Breitner, M. H. (Hrsg.) ; Bruns, B. (Hrsg.) ; Lehner, F. (Hrsg.): *Neue Trends im E-Learning – Aspekte der Betriebswirtschaftslehre und Informatik.* Heidelberg : Physica-Verlag, 2007, S. 1–15

[Sve98] Sveiby, K. E.: *Wissenskapital – Das unentdeckte Vermögen.* Landsberg am Lech : Verlag Moderne Industrie, 1998

[SVS03] Six, H. W. ; Voss, J. ; Schäfer, W.: Architekturschema für VU-Systeme / CampusSource e.V. Version: 2003. `http://www.campussource.de/projekte/architektur_vusysteme.html`. Hagen, 2003. – Forschungsbericht

[SW04] Seufert, S. ; Wessner, M.: Werkzeuge für spezielle Lernmethoden. In: Haake, J. (Hrsg.) ; Schwabe, G. (Hrsg.) ; Wessner, M. (Hrsg.): *CSCL-Kompendium*. München, Wien : Oldenbourg, 2004, S. 127–136

[Tap97] Tapscott, D.: *The Digital Economy: Promise and Peril In The Age of Networked Intelligence*. New York, NY, USA : McGraw-Hill, 1997

[Ter02] Tergan, S.-O.: Hypertext und Hypermedia: Konzeption, Lernmöglichkeiten, Lernprobleme. In: Issing, L. J. (Hrsg.) ; Klimsa, P. (Hrsg.): *Information und Lernen mit Multimedia*. 3. Auflage. Weinheim : Beltz Verlag, 2002, S. 99–112

[TG03] Thelen, T. ; Gruber, C.: Kollaboratives Lernen mit WikiWikiWebs. In: Kerres, M. (Hrsg.): *Digitaler Campus – vom Medienprojekt zum nachhaltigen Medieneinsatz in der Hochschule*. Münster : Waxmann Verlag, 2003, S. 356–365

[Tha02] Thaller, G. E.: *Software-Anforderungen für Webprojekte – Vorgehensmodelle, Spezifikation, Design*. 1. Bonn : Galileo Press, 2002

[THH+96] Tulodziecki, G. ; Hagemann, W. ; Herzig, B. ; Leufen, S. ; Mütze, C.: *Neue Medien in den Schulen: Projekte-Konzepte-Kompetenzen*. Gütersloh : Bertelsmann Stiftung, 1996

[Tur99] Turowski, K.: Ordnungsrahmen für komponentenbasierte betriebliche Anwendungssysteme. In: *Tagungsband des 1. Workshops Komponentenorientierte betriebliche Anwendungssysteme*. Magdeburg : Universität Magdeburg, 1999

[Tur03] Turau, V.: Die Rolle von Web-Services im Middleware Spektrum. In: *Online 2003 Messe Düsseldorf – Web Services: Schlüssel für eBusiness Integration*. Online GmbH, 2003

[Tyl03] Tyler, J. G.: Digital Training Systems Architecture: Proposal for Elaboration of the IEEE Learning Technology Systems Architecture, 1484.1 / ISO/IEC JTC1 SC36. 2003 (N0032). – Working Paper

[Vol04] Volkmer, R.: Blended Learning – Im Zeichen der Zeit. In: Baumbach, J. (Hrsg.) ; Kornmayer, E. (Hrsg.) ; Volkmer, R. (Hrsg.) ; Winter, H. (Hrsg.): *Blended Learning in der Praxis. Konzepte, Erfahrungen & Überlegungen von Aus- und Weiterbildungsexperten*. Dreieich : IMSELBST-Verlag, 2004, S. 21–27

[Völ06] Völter, M.: Services, Komponenten, Modelle. In: *OBJEKTspektrum* (2006), Nr. 5

[Wan07] Wannemacher, K.: Anreizsysteme zur Intensivierung von E-Teaching an Hochschulen. In: Eibl, C. (Hrsg.) ; Magenheim, J. (Hrsg.) ; Schubert, S. (Hrsg.) ; Wessner, M. (Hrsg.): *DeLFI 2007: 5. Deutsche e-Learning Fachtagung Informatik*. Bonn : Köllen Druck + Verlag, 2007, S. 161–172

[WBR04] Wilson, S. ; Blinco, K. ; Rehak, D.R.: Service-Oriented Frameworks – Modelling the infrastructure for the next generation of e-Learning Systems / DEST (Australia), JISC-CETIS (UK), and Industry Canada. 2004. – Forschungsbericht

[Wei76] Weick, K. E.: Educational Organizations as Loosely Coupled Systems. In: *Administrative Science Quarterly* 21 (1976), S. 1–19

[Wei93] Weidemann, B.: Instruktionsmedien / Institut für Erziehungswissenschaft und Pädagogische Psychologie, Universität der Bundeswehr. München, 1993. – Forschungsbericht

[Wen98a] Wenger, E.: Communities of Practice – Learning as a social System. In: *System Thinker* 6 (1998)

[Wen98b] Wenger, T.: *Communities of Practice. Learning, Meaning, and Identity*. Cambridge, MA, USA : Cambridge Univ. Press, 1998

[Wes01] Wessner, M.: Software für e-Learning: Kooperative Umgebungen und Werkzeuge. In: Schulmeister, R. (Hrsg.): *Virtuelle Universität Virtuelles Lernen*. München, Wien : Oldenbourg, 2001, Kapitel 7, S. 195–219

[Wie03] Wiedeking, M.: Sinn und Unsinn von Web-Services. In: *Online 2003 Messe Düsseldorf – Web Services: Schlüssel für eBusiness Integration*. Online GmbH, 2003

[Wil06] Wilson, S.: Personal Learning Environment. In: *Pushing the boundaries of the VLE (II)*. Utrecht, NL, Sept. 2006 2006

[WK07] Wang, E. ; Klebl, M.: *E-Learning-Standards und informelles Lernen – ein Unverhältnis?* Eintrag vom 07.02.2007 im LEARNTEC-Blog. `http://blog.learntec.de/?p=82`. Version: 02 2007, Abruf: 30.06.2008

[WKH+93] Weidenmann, B. ; Krapp, A. ; Hofer, M. ; Huber, G. ; Mandl, H.: *Pädagogische Psychologie*. Weinheim, Basel : Beltz Verlag, 1993

[WL99] Weiss, D. M. ; Lai, C. T. R.: *Software Product-Line Engineering – A Family-Based Software Development Process*. Reading, Masachusetts, USA : Addison-Wesley, 1999

[WS01] Wenger, E. C. ; Snyder, W. M.: Communities of Practice: The Organizational Frontier. In: *Harvard Business Review on Organizational Learning*. Boston, MA, USA : Harvard Business School Press, 2001, S. 1–20

[XYES03] Xu, Zhengfang ; Yin, Zheng ; El Saddik, A.: A Web Services Oriented Framework for Dynamic E-Learning Systems. In: *Proceedings of Canadian Conference on Electrical and Computer Engineering 2003* Bd. 2 IEEE Computer Society, 2003, S. 943–646

[Yas00] Yass, M.: *Entwicklung multimedialer Anwendungen: Eine systematische Einführung*. Heidelberg : Dpunkt, 2000

[Zac99] Zack, M.H.: Managing Codified Knowledge. In: *Sloan Management Review* Bd. 40. Cambridge, MA, USA, 1999, S. 45–58

[ZR05] Zawacki-Richter, O.: Einsatzkonzepte für E-Learning zur Integration in nachhaltige Supportstrukturen. In: Breitner, M. H. (Hrsg.) ; Hoppe, G. (Hrsg.): *E-Learning – Einsatzkonzepte und Geschäftsmodelle*. Heidelberg : Physica-Verlag, 2005, S. 37–52

A. Anhang

A.1. Anfragesprachen für Content-Repositories

A.1.1. XQuery

Die *XML Query* gilt in der Version 1.0 als W3C Recommendation. Sie kann zur Abfrage für XML-basierte Daten genutzt werden, aber auch für alle anderen Datenquellen, deren Inhalt mit Hilfe von XML repräsentiert werden kann. Zur besseren Benutzbarkeit existiert neben der XML-basierten Syntax auch eine verständliche nicht-XML-Syntax.

XQuery basiert auf dem selben Datenmodell wie XPath (*XML Path Language*) und nutzt XPath-Ausdrücke zur Navigation in XML. Um Anfragen zu formulieren, erlaubt XQuery die Bildung so genannter FLOWR-Ausdrücke[1] (vgl. Listing A.1.1). Weitere

Listing A.1: Beispiel für einen FLWOR-Ausdruck in XQuery

```
1 for $x in doc("buecher.xml")/bibliothek/semapp
  where $x/signatur>"TQM"
3 order by $x/titel
  return $x/titel
```

XPath-Funktionen können darüber hinaus zur Gestaltung eigener Funktionen und Kontrollstrukturen für Anfrage oder Ausgabe verwendet werden.

Insgesamt ist XQuery eine sehr ausdrucksstarke Anfragesprache, die auch viele Möglichkeiten bei der Ausgabe anbietet. Eine Kenntnis der XML-Struktur der anzufragenden Datenquelle ist allerdings notwendig, um die Anfrage zu formulieren. XQuery kann bei der Kommunikation zwischen Lernobjekt-Repositories genutzt werden, um XML-Repräsentationen der Metadaten über die Lernobjekte (bspw. LOM) zu durchsuchen.

A.1.2. CQL

Die *Common Query Language* bietet gegenüber XQuery eine intuitivere Formulierungsmöglichkeiten von Anfragen, die aber trotzdem eine hohe Ausdrucksstärke bietet. Dabei verlangt sie keine syntaktische Auszeichnung der einzelnen Bestandteile einer Anfrage, wie z. B. in XQuery durch die Schlüsselwörter *for*, *where*, etc. Für eine einfache Suchanfrage nach einer Zeichenkette ist es ausreichend, diese anzugeben[2]. Es besteht die Möglichkeit, diese Zeichenketten durch boolesche oder vergleichende Operatoren zu

[1] Hierbei können Anfragen mit den Ausdrücken *For*, *Let*, *OrderBy*, *Where* und *Return* definiert werden.

[2] Ein Beispiel für eine gültige CQL-Anfrage ist „repository“.

verknüpfen oder Existenzquantoren zu verwenden. Darüber hinaus erlaubt CQL die Nutzung von Metadate zur näheren Spezifizierung der Suche. Eine Suche nach einem Objekt, dass eine der Zeichenketten „repository“ und „E-Learning“ im Titel enthält, kann wie folgt formuliert werden: „*title any 'repository E-Learning'*“.

CQL erlaubt zusätzlich Platzhalter („*“, „?“), unterscheidet Datentypen (Zeichenketten, Zahl, Datum, etc.) und bietet sogar Operatoren, die eine gewisse Unschärfe zulassen[3]. Darüber hinaus können Namensräume genutzt und somit zusätzliche Operatoren oder Metadaten verwendet werden.

A.1.3. VSQL

Die *Very Simple Query Language* ist die derzeit von fast allen SQI-fähigen Repositories am häufigsten unterstützte Anfragesprache. Sie ist auf absolute Einfachheit ausgerichtet und erlaubt nur eine einfache Angabe von Suchtermen für eine Volltextsuche über alle Daten eines Repositories. Die Suche kann weder auf bestimmte Attribute eingeschränkt werden, noch die Ordnung oder Ausgabe der Ergebnisse beeinflussen. Ein Suchterm wird als Beispiel in Listing A.1.3 dargestellt.

Listing A.2: Beispiel-Anfrage für VSQL

```
<simpleQuery>
    <term>repository</term>
    <term>E-Learning</term>
</simpleQuery>
```

[3]Zum Beispiel die Schlüsselwörter *prox* und *stern*.

A.2. Geschehnis-Prädikatoren der Terminologie

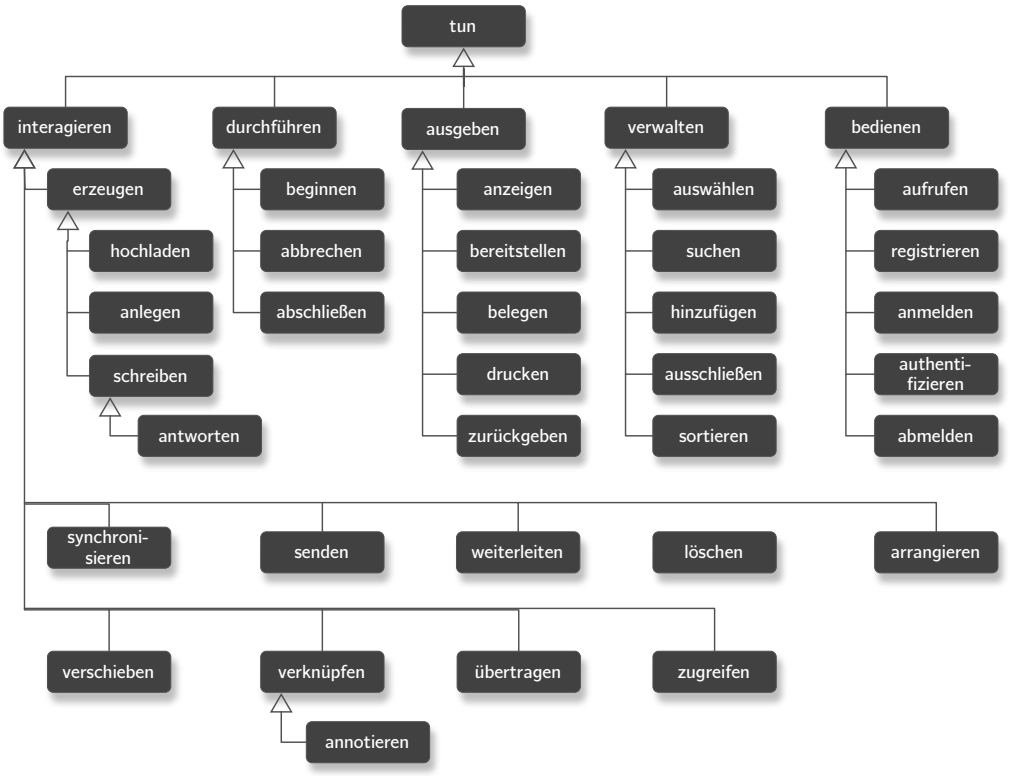

Abbildung A.1.: Die Basis-Geschehnisprädikatoren (P) der Terminologie

A.3. Ding-Prädikatoren der Terminologie

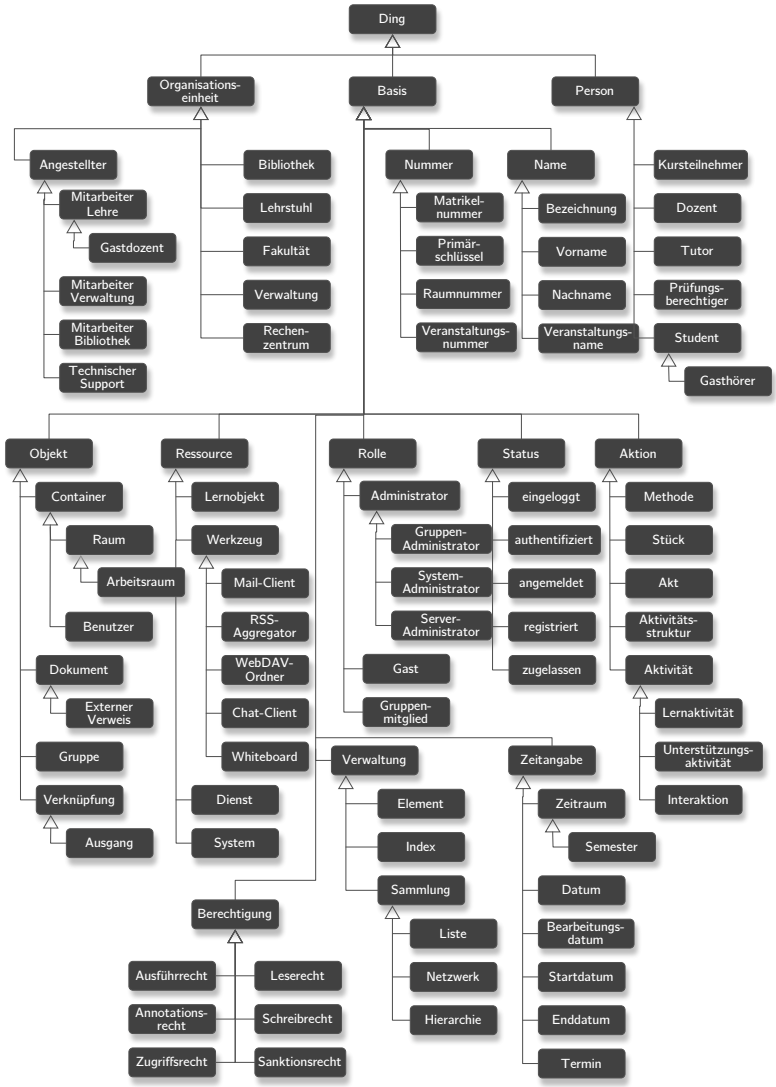

Abbildung A.2.: Die Basis-Dingprädikatoren (Q) der Terminologie

A.4. Alfresco-Modell zur Verwaltung eines Weblogs

Listing A.3: weblog-model.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<model
        name="koala:blogModel"
        xmlns="http://www.alfresco.org/model/dictionary/1.0">
        <description>Weblog Model</description>
        <version>1.0</version>

        <imports>
          <!-- Import Alfresco Dictionary Definitions -->
          <import uri="http://www.alfresco.org/model/dictionary/1.0"
                  prefix="d" />
          <!-- Import Alfresco Content Domain Model Definitions -->
          <import uri="http://www.alfresco.org/model/content/1.0"
                  prefix="cm" />
          <!-- Import koaLA Domain Model Definitions -->
          <import uri="http://www.provideal.net/koala/model/1.0"
                  prefix="koala" />
        </imports>

        <types>
          <type name="koala:blog">
            <parent>cm:folder</parent>
            <mandatory-aspects>
              <aspect>cm:titled</aspect>
            </mandatory-aspects>
          </type>

          <type name="koala:blogpost">
            <parent>cm:folder</parent>
            <mandatory-aspects>
              <aspect>cm:titled</aspect>
              <aspect>koala:blogclassifiable</aspect>
            </mandatory-aspects>
          </type>

          <type name="koala:blogcategory">
                <title>Blog Category</title>
                <parent>cm:cmobject</parent>
          </type>
        </types>

        <aspects>
          <aspect name="koala:blogclassifiable">
            <title>Blog Classifiable</title>
                <properties>
                  <property name="koala:category">
                    <title>Category</title>
                    <type>d:noderef</type>
                    <mandatory>false</mandatory>
```

```
50                          <multiple>false</multiple>
                            <index enabled="true">
52                                  <atomic>true</atomic>
                                    <stored>true</stored>
54                                  <tokenised>true</tokenised>
                            </index>
56                        </property>
                        </properties>
58                </aspect>
            </aspects>
60 </model>
```

A.5. Beschreibungsmodell der konzipierten Methode

A.5.1. Vorgehen zur Plattformentwicklung

ID	Kategorie	Prozess	Beschreibung
1.1		Analyse der E-Learning-Domäne	Festlegung der Anwendungsdomäne; Ableitung gemeinsamer, optionaler u. variabler Features; Präzisierung des Produktraums der Produktfamilie
Teilprozesse/Aspekte		Domänendefinition, Erstellung eines Lexikons, Beschreibung von Konzepten, Featuremodellierung	
Ziel		Definition der Anwendungsdomäne und des Produktraums, Spezifizierung des Funktionsumfangs herzustellender Produkte	
Techniken			
Ergebnis		Produktraumbeschreibung; Domänenlexikon; Konzeptmodelle; Featuremodelle	
Aktor		Softwarearchitekt; Softwareentwickler; Didaktiker; Fachexperte	
Bewertung/Kriterien		Bsp.: Mittel- und langfristige Realisierbarkeit von Anwendungen auf Basis der Plattform	

ID	Kategorie	Prozess	Beschreibung
1.1.1	Analyse der E-Learning-Domäne	Domänendefinition	Beschreibung des Umfangs einer Domäne und Charakterisierung des Produktraums der Domäne
Teilprozesse/Aspekte			
Ziel		Definition der Anwendungsdomäne und des Produktraums	
Techniken		Strategieanalyse, Marktanalyse, Bsp. existierender Systeme der Produktfamilie, Gegenbeispiele (z. B. Systeme außerhalb der Domäne), generische Regeln für die Inklusion oder Exklusion	
Ergebnis		Definition der Anwendungsdomäne und des Produktraums	
Aktor		Fachexperte, Didaktiker, Softwarearchitekt	
Bewertung/Kriterien		Expertenaustausch, Produktstrategie, Marktanalysen, mittel- und langfristige Realisierbarkeit	

ID	Kategorie	Prozess	Beschreibung
1.1.2	Analyse der E-Learning-Domäne	Erstellung eines Domänen-Lexikons	Beschreibung der in der Domäne verwendeten Begriffe
Teilprozesse/Aspekte			
Ziel		Sicherstellung einer gemeinsamen Kommunikationsbasis für alle Projektbeteiligten	
Techniken		Dokumentenanalyse, Interviews, Workshops, normsprachliche Rekonstruktion, Terminologie	
Ergebnis		Lexikon über Fachbegriffe der Domäne	
Aktor		Didaktiker, Softwarearchitekt, Fachexperte	
Bewertung/Kriterien		Klarheit, Widerspruchsfreiheit, Vollständigkeit, Allgemeinverständlichkeit	

ID	Kategorie	Prozess	Beschreibung
1.1.3	Analyse der E-Learning-Domäne	Beschreibung von Konzepten	Beschreibung der Konzepte einer Domäne mit Hilfe einer angemessenen Modellierung
Teilprozesse/Aspekte			
Ziel		Sicherstellung eines gemeinsamen Verständnisses der Konzepte für alle Projektbeteiligten	
Techniken		normsprachliche Spezifikation, Objektdiagramme, Interaktionsdiagramme, ERM, Datenflussdiagramme, informeller Text	
Ergebnis		Modell der Domänenkonzepte, Objektstruktur des OO-Frameworks	
Aktor		Didaktiker, Fachexperte, Softwarearchitekt	
Bewertung/Kriterien		Best-Practice-Bsp., Abbildbarkeit der Objektstruktur auf das OO-Framework, Expertenreviews	

ID	Kategorie	Prozess	Beschreibung
1.1.4	Analyse der E-Learning-Domäne	Featuremodellierung	Identifizierung von Entitäten, Funktionen und ihrer Beziehungen; Analyse von Ähnlichkeiten, Variationen, Kombinationen, Konflikten
Teilprozesse/Aspekte		Modellierung gemeinsamer Features, Modellierung optionaler Features, Modellierung von Freiheitsgraden mit variablen Features	
Ziel		Transparenz von Abhängigkeiten, Entscheidungsbasis für die Zuordnung von Features zu Plattform oder Anwendung	
Techniken		Application Requirements-Matrix, Priority-Based Analysis, Checklist-Based Analysis, FODA, Feature Diagramme, XPath, Captain Feature	
Ergebnis		Featuremodell, Katalog von Konfigurationsmöglichkeiten für neu zu implementierende Anwendungen auf Basis der Plattform	
Aktor		Softwarearchitekt, Softwareentwickler, Didaktiker, Fachexperte	
Bewertung/Kriterien		Vollständigkeit, Granularität	

ID	Kategorie	Prozess	Beschreibung
1.2		Domänen-Design	Standardisierung der Entwicklungs- u. Laufzeitinfrastruktur; Entwurf der Plattformarchitektur und ihrer Komponenten
Teilprozesse/Aspekte		Standardisierung der Infrastruktur, Entwurf der statischen Plattformarchitektur, Entwurf wiederverwendbarer Komponenten	
Ziel		Beschreibung des SW-Entwurfs von Plattform u. Komponenten	
Techniken			
Ergebnis		Beschreibung einer standardisierten Entwicklungs- u. Laufzeitinfrastruktur, Entwurfsdokumente für die Plattform und ihrer Komponenten	
Aktor		Softwarearchitekt, Softwareentwickler, Techniker	
Bewertung/Kriterien		Bsp.: Anwendbarkeit von Standards, konzeptionelle Integrität, State of the art, Wiederverwendbarkeit, Vollständigkeit	

ID	Kategorie	Prozess	Beschreibung
1.2.1	Domänen-Design	Standardisierung der Infrastruktur	Standardisierung der Entwicklungsumgebung, der Art der Persistenz- u. Basisinfrastruktur, Auswahl von SWE-Pattern und Frameworks
Teilprozesse/Aspekte		Basisinfrastruktur für Schichten u. Vermittlungsschichten, genormte Schnittstellen zu den genutzten externen Systemen, OO-Framework, Programmiersprache(n), Application Server,	
Ziel		Definition der Entwicklungsumgebung und der Basisinfrastruktur, Vereinbarung über Architektur- und SW-Entwicklungskonzepte	
Techniken		UML, Marktanalysen, Evaluationen, Prototyping	
Ergebnis		Standardisierte Infrastruktur, Technologiestrategie, Architekturstrategie, Programmierrichtlinien	
Aktor		Softwarearchitekt, Softwareentwickler, Techniker	
Bewertung/Kriterien		vorhandene Erfahrungen mit Architekturen, Technologien, u. SWE; Aufgabenangemessenheit, Portabilität, Zuverlässigkeit, Änderbarkeit, Effizienz, Best Practices, State of the art, Referenzprojekte, konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
1.2.2	Domänen-Design	Entwurf der statischen Plattformarchitektur	Entwurf der Kernfunktionalität der Plattform, Entwurf eines Plug-In-Mechanismus zur Integration von Komponenten
Teilprozesse/Aspekte		Kernfunktionalität: Persistierung von Objekten, Authentifizierung, Autorisierung, Logging, Exception Handling, RSS-Engine, Tagging, Schnittstellen, Kommunikationswerkzeuge; Aspekte des Entwurfs: Nebenläufigkeit, Komponentenschnittstellen, Verteilung von Komponenten, Fehlertoleranz, Interaktion und Präsentation, Persistierung von Daten	
Ziel		Verringerung der Komplexität; Vorbereitung arbeitsteiligen Entwickelns; Einbindung existierender Lösungen; Ermöglichung der Weiterentwicklung	
Techniken		Brainstorming, Entwurfsmethoden, SW-/Architekturpatterns, Diagrammsprachen, z. B. UML; Traceability Links, HyperUML	
Ergebnis		Entwurfsdokumente für die Plattformarchitektur	
Aktor		Softwarearchitekt, Softwareentwickler	
Bewertung/Kriterien		Expertenaustausch, Nachvollziehbarkeit, Lesbarkeit, Grad der Anwendung/Anwendbarkeit von Standards, Wiederverwendbarkeit, konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
1.2.3	Domänen-Design	Entwurf wiederverwendbarer Komponenten	Entwurf allgemeiner und domänenspezifischer Dienste, die in verschiedenen Anwendungen wieder verwendet werden können
Teilprozesse/Aspekte		Funktionsblöcke: Gruppenstrukturen (öffentliche Gruppen, private Gruppen, Zutrittsmechanismen), Dokumentenorganisation (Gruppen- und private Arbeitsräume, Locking/Sperrmechanismen, Transformationen (Medientypen, Dateiumwandlung), Versionierung von Objekten, Kategorisierung), Kommunikation (Foren, Blogs, Wikis, Annotationen, internes Nachrichtensystem, Chat/VoIP/Audo/Video), Koordination(Kalender, Workflows), Community (Buddy-Listen, Awareness, Friend-of-a-friend (soziales Netzwerk), Profile), Aspekte des Entwurfs: Nebenläufigkeit, Komponentenschnittstellen, Verteilung von Komponenten, Fehlertoleranz/Behandlung von Fehlern und Ausnahmen, Interaktion und Präsentation, Persistierung von Daten	
Ziel		Verringerung der Komplexität; Vorbereitung arbeitsteiligen Entwickelns; Einbindung existierender Lösungen; Ermöglichung der Weiterentwicklung	
Techniken		Brainstorming, Entwurfsmethoden, SW-/Architekturpatterns, Diagrammsprachen, z. B. UML; Traceability Links, HyperUML	
Ergebnis		Entwurfsdokumente für die Plattformarchitektur	
Aktor		Softwarearchitekt, Softwareentwickler	
Bewertung/Kriterien		Expertenaustausch, Nachvollziehbarkeit, Lesbarkeit, Grad der Anwendung/Anwendbarkeit von Standards, Wiederverwendbarkeit, konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
1.3		Domänen-Implementierung	Aufbau der entworfenen Entwicklungs- und Laufzeitinfrastruktur, Umsetzung der Plattformarchitektur und ihrer Komponenten
Teilprozesse/Aspekte		Aufbau der Infrastruktur, Implementierung der Plattformarchitektur, Implementierung ihrer Komponenten	
Ziel		Bereitstellung der standardisierten Infrastruktur, Plattformarchitektur und Komponenten	
Techniken			
Ergebnis		Funktionierende Entwicklungs- und Laufzeitinfrastruktur, funktionierendes Basissystem inkl. Komponenten	

Aktor	Softwarearchitekt, Softwareentwickler, Techniker
Bewertung/Kriterien	Bsp.: Einsatz von Standards, Kompatibilität, Korrektheit, Stabilität, Performance, Erweiterbarkeit, Portierbarkeit

ID	**Kategorie**	**Prozess**	**Beschreibung**
1.3.1	Domänen-Implementierung	Aufbau der Infrastruktur	Installation und Integration verschiedenster Technologien, Basisinfrastrukturkomponenten, IDEs, Frameworks, Schnittstellen und Protokolle
Teilprozesse/Aspekte			
Ziel		Bereitstellung einer Entwicklungs- und Laufzeitumgebung für die Plattform und für ihre Komponenten	
Techniken		Basisinfrastruktur, Architekturprinzipien	
Ergebnis		Lauffähige Entwicklungsumgebung auf Basis einer standardisierten Infrastruktur	
Aktor		Softwarearchitekt, Softwareentwickler, Techniker	
Bewertung/Kriterien		Einsatz von Standards, Dokumentation der Konfiguration, Kompatibilität, Automatisierbarkeit	

ID	**Kategorie**	**Prozess**	**Beschreibung**
1.3.2	Domänen-Implementierung	Implementierung der Plattformarchitektur	Umsetzung der Kernfunktionalität, Umsetzung eines PlugIn-Mechanismus zur Integration von Komponenten
Teilprozesse/Aspekte			
Ziel		Umsetzung des Architekturentwurfs zu einem lauffähigen System	
Techniken		Programmiersprachen, SW-Entwicklungspattern, Entwicklungsumgebungen, Bibliotheken, Frameworks, Hyperspace-Ansatz, Hyper/J; Unit Tests	
Ergebnis		Standardplattform für die Anwendungsentwicklung im E-Learning	
Aktor		Softwarearchitekt, Softwareentwickler	
Bewertung/Kriterien		Dokumentation, Expertenreviews, Testbarkeit, Korrektheit, Stabilität, Lesbarkeit, Zweckmäßigkeit, Erweiterbarkeit, Portierbarkeit	

ID	Kategorie	Prozess	Beschreibung
1.3.3	Domänen-Implementierung	Implementierung wiederverwendbarer Komponenten	Implementierung allgemeiner und domänenspezifischer Funktionen als wiederverwendbare Komponenten
Teilprozesse/Aspekte			
Ziel			Umsetzung des Komponenten-Entwurfs zu lauffähigen Komponenten; Integration der Komponenten
Techniken			Programmiersprachen, SW-Entwicklungspattern, Entwicklungsumgebungen, Bibliotheken, Frameworks, Hyperspace-Ansatz, Hyper/J; Unit Tests
Ergebnis			Komponenten zur Unterstützung allgemeiner und domänenspezifischer Belange, die bei der Anwendungsentwicklung im E-Learning wieder verwendet werden können
Aktor			Softwarearchitekt, Softwareentwickler
Bewertung/Kriterien			Dokumentation, Expertenreviews, Testbarkeit, Korrektheit, Stabilität, Lesbarkeit, Zweckmäßigkeit, Erweiterbarkeit, Portierbarkeit

A.5.2. Vorgehen zur Anwendungsentwicklung

ID	Kategorie	Prozess	Beschreibung
2.1		Projektvorbereitung	Projektorganisation und Rahmenbedingungen festlegen, Projektziele definieren, Projektaufbau- und Ablaufstruktur planen, Termin- und Budgetplanung, Qualitäts- und Risikomanagement
Teilprozesse/Aspekte		Initiierung, Festlegung von Rollen und Verantwortlichkeiten, Identifikation der Stakeholder, Zieldefinition, Bedarfsanalyse, Anforderungen an Neuentwicklung (grob) und Priorisierung, Rahmenbedingungen ermitteln, Projektstruktur definieren, Termin- und Budgetplanung, Risikoanalyse und Qualitätssicherungsplanung	
Ziel		Durchführbarkeit des Projektes evaluieren bzw. herstellen	
Techniken			
Ergebnis		Projektorganisation, Projektziele, Projektaufbau- u. -ablaufplan, Termin- u. Budgetplan, Qualitätssicherungsplan, Risikoanalyse	
Aktor		Initiator, Fachexperte, Didaktiker, Organisationsspezialist, Projektmanager	
Bewertung/Kriterien		Bsp.: Projekterfahrung der Mitglieder, Projektzielerreichung, konzeptuelle Konsistenz, Orientierung an organisatorischen Schnittstellen, Orientierung am Erfolg, Sensibilisierung der Mitglieder bzgl. Qualität u. Projektrisiken	

ID	Kategorie	Prozess	Beschreibung
2.1.1	Projektvorbereitung	Initiierung	Initiierung durch Identifikation und Beschreibung von Bedarf und Bedürfnis
Teilprozesse/Aspekte			
Ziel		Identifikation und Beschreibung von Bedarf und Bedürfnis	
Techniken		Methoden der Bedarfsanalyse, Methoden der Marktforschung und -analyse, Methoden der Trendforschung, Umfragen, Balanced Scorecard, Skill Gap Analysen, Arbeitsplatzanalyse, Auditierung	
Ergebnis		Dokumentation von Bedarf und Bedürfnis	
Aktor		Initiator, Fachexperte, Didaktiker, Organisationsspezialist	
Bewertung/Kriterien		Prüfung der Ergebnisse der Bedarfs- und Bedürfnisidentifikation auf Plausibilität	

ID	Kategorie	Prozess	Beschreibung
2.1.2	Projektvorbereitung	Festlegung von Rollen und Verantwortlichkeiten	Definition von Auftraggeber, Lenkungsausschuss, Projektleitung, Projektteam, speziellen Gruppen; Kontrolle der Termine und ggf. internen u. externen Kosten; Einbindung der Fachbereiche; Einbindung von Projektmitarbeitern; Rollen von Ansprechpartnern
Teilprozesse/Aspekte			
Ziel		Bildung einer Projektorganisation und Herstellung ihrer Arbeitsfähigkeit; allg. Verständnis von Rollen und Verantwortlichkeiten	
Techniken		Workshops; Moderation und Mediation; Methoden der Entscheidungsfindung: Aussprache, Ansprache, Dialog; Organigramme; Workflow-Modellierung	
Ergebnis		Beschreibung von Rollen und Verantwortlichkeiten im Projektteam; Definition von Prozessen (Genehmigung von Aufwänden, Eskalationsmechanismen, Qualitätssicherung, etc.)	
Aktor		Initiator; Fachexperte; Didaktiker; Organisationsspezialist; Projektmanager	
Bewertung/Kriterien		Projekterfahrung der Projektmitglieder; gewinnbringende Integration individueller Kenntnisse und Fähigkeiten in die Projektorganisation; Funktionsfähigkeit von Koordinationsmechanismen; zielgerichtete Produktivität	

ID	Kategorie	Prozess	Beschreibung
2.1.3	Projektvorbereitung	Identifikation der Stakeholder	Identifikation von Aktoren, Interessenten und Nutzern; Beschreibung und Bewertung ihrer Ziele
Teilprozesse/Aspekte		Identifikation von Aktoren, Interessenten und Nutzern	
Ziel		Identifikation und Beschreibung der Gesamtheit der Stakeholder und deren Einfluss sowie Mitwirkung am Bildungsprozess und am Projekt.	
Techniken		Literaturanalyse, Analyse von Strategiekonzepten, Business-Plänen, Organigrammen, Richtlinien, Curricula, Rahmenplänen, Stoffverteilungsplänen, Prüfungsordnungen, Stundenplänen und Kriterien der Überprüfung	
Ergebnis		Benennung der Stakeholder sowie Dokumentation und Bewertung ihrer Interessen hinsichtlich Bildungsprozess und Projekt, Dokumentation der Anforderungen und Grobziele. Identifikation und Beschreibung der Bedeutung und der Einflüsse auf Entstehung, Akzeptanz und Marktfähigkeit des SPL-Produktes	
Aktor		Initiator; Organisationsspezialist; Projektmanager	
Bewertung/Kriterien		Auftragsklärung und -bestätigung, Workshops, Interviews	

ID	Kategorie	Prozess	Beschreibung
2.1.4	Projektvorbereitung	Zieldefinition	Identifikation, Beschreibung und Bewertung der Projektziele der relevanten Stakeholder
Teilprozesse/Aspekte			
Ziel		Identifikation, Beschreibung und Bewertung der Ziele der relevanten Stakeholder, Umsetzung in ein Qualitätssystem nach ISO 9000	
Techniken		Befragung, Interview, Umfrage, Workshop, Dokumentenanalyse, Assessment	
Ergebnis		Dokumentation und Bewertung der Ziele relevanter Stakeholder, Operationalisierbare, konsistente Zielsetzungen zur Qualitätsmessung nach ISO 9000	
Aktor		Initiator; Projektmanager; Evaluationsexperte; Organisationsspezialist	
Bewertung/Kriterien		Zielerreichung	

ID	Kategorie	Prozess	Beschreibung
2.1.5	Projektvorbereitung	Bedarfsanalyse	Spezifikation, Beschreibung und Bewertung des Bildungsbedarfs; Schwachstellen-/Potenzialanalyse (funktional/nicht-funktional); Erfassung von Ist-Mengengerüsten; Beschreibung der Zielsetzung des Projektes
Teilprozesse/Aspekte			
Ziel		Spezifikation, Beschreibung und Bewertung des Bildungsbedarfs und der Ziele, die mit dem Projekt verbunden werden und deren Beschreibung	
Techniken		Befragung, Interview, Umfrage, Workshop, Dokumentenanalyse, systematische und methodische Ausformulierung der Bedarfsanalyse (z.B. Verfahren der Markt- und Bildungsforschung, der Organisationsanayse)	
Ergebnis		Dokumentation und Bewertung des Bildungsbedarfs und der Projektziele; Schwachstellen/-potenzialanalyse; Ist-Mengengerüste	
Aktor		Initiator; Projektmanager; Evaluationsexperte; Organisationsspezialist; Fachexperte; Didaktiker	
Bewertung/Kriterien		Dokumentation und Bewertung des Bildungsbedarfs und der Projektziele	

ID	Kategorie	Prozess	Beschreibung
2.1.6	Projektvorbereitung	Anforderungen an Neuentwicklung (grob) und Priorisierung	Erhebung und Beschreibung der groben Systemanforderungen (funktional/nicht-funktional); Beschreibung des Bildungsbedarfs (Niveau und Inhalt) und der Art der Bildung (formal, nicht-formal, vorgeschrieben, Lernzielsystematik); Beschreibung der Soll-Mengengerüste;
Teilprozesse/Aspekte			
Ziel		Dokumentation und Priorisierung der groben Anforderungen als Basis für die Planung der Iterationen	
Techniken		Befragung, Interview, Dokumentenanalyse, Methoden des Requirements-Engineering	
Ergebnis		Beschreibung des Bedarfs und der Art der Bildung; Beschreibung funktionaler u. nicht-funktional Systemanforderungen; Soll-Mengengerüste	
Aktor		Initiator; Projektmanager, Softwarearchitekt; Evaluationsexperte; Didaktiker; Organisationsspezialist	
Bewertung/Kriterien		Anforderungsklärung und -bestätigung, Workshops, Interviews	

ID	Kategorie	Prozess	Beschreibung
2.1.7	Projektvorbereitung	Rahmenbedingungen ermitteln	Ermittlung finanzieller, zeitlicher, organisatorischer, technischer und juristischer Rahmenbedingungen; Analyse des internen u. externen Umsystems; Analyse der Zielgruppe,
Teilprozesse/Aspekte			**Internes Umsystem**: Hochschulweite E-Learning-/IT-/Medien-Strategie; bestehende Dienstleistungen; bestehende Basisinfrastruktur; (konkurrierende) Projekte; interne Einstellung von Entscheidungsträgern; Lernkultur; **Externes Umsystem**: Know-how in Literatur und bei Anbietern; Referenzmodelle und Rahmenarchitekturen; Standards; Marktstudien, Evaluationen, Trends; gesetzliche Vorgaben (Datenschutz); Bildungspolitik; **Analyse der Zielgruppe**:
Ziel			Ermittlung expliziter und impliziter Rahmenbedingungen für das Projekt; Identifikation, Beschreibung und Bewertung relevanter Einflüsse auf das Projekt bzw. der Phase des Betriebs, die sich aus dem internen und externen Umsystem ergeben
Techniken			Befragung; Interview; Dokumentenanalyse; Hinzunahme von Fachstellen, Marktanalysen, quantitative und qualitative Methoden der empirischen Sozial-/Bildungsforschung; Zielgruppenanalyse

Ergebnis	Dokumentation der Rahmenbedingungen; Dokumentation des organisatorischen und institutionellen Kontextes; Dokumentation und Bewertung relevanter Kategorien der externen Kontexte
Aktor	Initiator; Projektmanagement; Organisationsspezialist; Softwarearchitekt; Fachexperte; Softwareentwickler
Bewertung/Kriterien	Vollständigkeit, Qualität der Informationen, Plausibilitätsprüfung, Methoden zur Analyse und Bewertung der Aktoren

ID	**Kategorie**	**Prozess**	**Beschreibung**
2.1.8	Projektvorbereitung	Projektstruktur definieren	Aufbau- und Ablauforganisation für die Durchführung des Projektes festlegen; Beschreibung der Beziehungen zwischen Aufbau- und Ablauforganisationselementen
Teilprozesse/Aspekte			
Ziel		abgestimmte Aufbau- und Ablaufstruktur	
Techniken		DIN 69901, ICB, Organigramme, Phasenschemata, Vorgehensmodelle	
Ergebnis		Dokumentation der Projektaufbau- und Ablaufstruktur, Work Breakdown Structure	
Aktor		Initiator; Projektmanager; Softwarearchitekt; Fachexpert; Organisationspezialist	
Bewertung/Kriterien		Vollständigkeit; konzeptuelle Konsistenz; Granularität; Anwendbarkeit; Adaptivität	

ID	Kategorie	Prozess	Beschreibung
2.1.9	Projektvorbereitung	Termin- u. Budgetplanung	Beschreibung der zeitlichen und logischen Anordnung von Arbeitspaketen sowie der Anordnungsbeziehungen zwischen Arbeitspaketen; Aufwandsabschätzung und Ressourcenbedarfsermittlung; Kostenermittlung und Budgetplanung
Teilprozesse/Aspekte		**Anfangskosten**: Beratungskosten; tatsächliche Kosten für Kauf oder Leasing der Hardware; Kosten für Anpassung des Ausrüstungsstandortes (Klimaanlage, Sicherheitseinrichtungen etc.), Kosten der Betriebssystemsoftware, Installationskosten für Kommunikationseinrichtungen (Telefon- und Datenleitungen), Personalbeschaffungskosten, anfängliche Personalkosten, Kosten für die Störung der sonstigen Organisation, Managementkosten für die Anfangsaktivitäten, Kapitalkosten; **Projektkosten**: Beschaffungskosten für Anwendungssoftware, Kosten der Anpassung der Software an bestehende Systeme, Personal- und Gemeinkosten bei Erstellung von Individualsoftwae im eigenen Haus, Kosten für Anwenderschulung, Datenerfassungskosten, Dokumentationskosten, Kosten des Projektmanagements; **Betriebskosten**: Wartungskosten (Software, Hardware), Verbrauchsgebühren (Strom, Telefon, etc.), Abschreibungen auf Hardware, Personalkosten	
Ziel		Qualitative und quantitative Planung und Überprüfung der Projektentwicklung	
Techniken		Analogieschätzung, Algorithmische Schätzung: Function Point, COCOMO, INVAS, Schätzklausuren; Gantt-Diagramme, Netzplantechniken	
Ergebnis		Abgestimmter Budget- und Ablaufplan	
Aktor		Projektmanager; Softwarearchitekt; Fachexperte	
Bewertung/Kriterien		Qualität der Schätzung; Orientierung an organisatorischen Schnittstellen; Orientierung am Erfolg; Einhaltung von Terminen und Budgets	

ID	Kategorie	Prozess	Beschreibung
2.1.10	Projektvorbereitung	Risikoanalyse	Benennung von Projektrisiken, ihrer Faktoren und ihrer Indikatoren; Einschätzung der Wahrscheinlichkeiten des Eintritts der Risiken; Bestimmung der Auswirkung bei Eintritt; Auswahl bedrohlicher Risiken; Erarbeitung von Gegenmaßnahmen; Überwachung der Risiken und der eingeleiteten Maßnahmen

Teilprozesse/Aspekte	Arten von Risiken: Leistungsrisiken, Terminrisiken, Kostenrisiken, Resourcenrisiken, Projektmanagementrisiken, Qualitätsrisiken, Konfigurationsrisiken
Ziel	Absicherung des Projekterfolges gegenüber möglichen Risiken
Techniken	Brainstorming; Experteninterviews; Lessons learned; Auditierung; Methoden der Wahrscheinlichkeitsrechnung; Entscheidungsbäume; Controlling; Qualitative Risikoanalyse, PMBOK, DIN 69905, PMF
Ergebnis	Risikoanalyse, Plan von Gegenmaßnahmen
Aktor	Softwarearchitekt; Softwareentwickler; Projektmanagement; Organisationsspezialist; Techniker, Fachexperte
Bewertung/Kriterien	Projekterfahrung im Team, Qualität der Schätzung von Wahrscheinlichkeiten; Identifizierung von Risikoindikatoren, Sensibilisierung der Projektmitglieder bzgl. Risiken, Etablierung eines "Frühwarnsystems"

ID	**Kategorie**	**Prozess**	**Beschreibung**
2.1.11	Projektvorbereitung	Qualitätssicherungsplanung	Identifizierung der für das Projekt geltenden Qualitätsstandards und Festlegung von Qualitätssicherungsmaßnahmen sowohl für die (Zwischen-)Produkte als auch für die Prozesse. Festlegung von Vorgehen zur kontinuierlichen Verbesserung der Projektmanagementprozesse

Teilprozesse/Aspekte	**Aspekt Dokumentation**: Prozessdokumentation: Zeitpläne, Programmierrichtlinien, Entwicklungsentscheidungen, Schätzprotokolle, Testberichte; **Aspekt Tests**: Systemtests, Integritätstests, Uni-Tests, Benutzertests
Ziel	Erhöhung der Effizienz von Prozessen, sowie der Qualität von (Zwischen-) Produkten und Dienstleistungen
Techniken	ISO 9001, Bootstrap, ISO 15504 aka SPICE, EFQM, CMM, ITIL (eher Fokus auf Betrieb, weniger auf Entwicklung), Six Sigma, PSP u. TSP, Hermes, PMBOK, SOX (Sarbanes-Oxley Act), Fehlermöglichkeits- u. Einflussanalyse (FMEA, z.B. MIL-P-1629)
Ergebnis	Qualitätsmanagementplan; organisierte Maßnahmen zur Verbesserung von (Zwischen-)Produkten, Prozessen oder Leistungen
Aktor	Softwarearchitekt; Softwareentwickler; Projektmanager; Fachexperte; Didaktiker; Support; Techniker; Initiator
Bewertung/Kriterien	Umfang der Dokumentation, Art und Häufigkeit von Tests, Sensibilisierung der Projektmitarbeiter bzgl. Qualität, Qualitätskontrolle

ID	Kategorie	Prozess	Beschreibung
2.2		Analyse	Analyse und Spezifikation der zu unterstützenden Didaktik/Methodik, des Medieneinsatzes, des Medien- u. Interaktionsdesigns und der Statik, Funktionalität und Dynamik des Lern-/Arbeitsszenarios
Teilprozesse/Aspekte		Analyse der zu unterstützenden Methodik/Didaktik, Medieneinsatz, Kommunikationsmöglichkeiten u. Formen, Workflows, Rollen und Rechte, Medien-/Interaktionsdesign	
Ziel		Identifikation und Beschreibung von Projekt-/Produkt-relevanten Einflüssen und Anforderungen	
Techniken			
Ergebnis		Evaluation der Abbildbarkeit organisatorischer u. didaktischer Anforderungen auf das Domänenkonzept; Design-Prototyp; Pflichtenheft	
Aktor		Evaluationsexperte; Fachexperte; Didaktiker; Projektmanager; Organisationsspezialist; Softwarearchitekt; Mediendesigner	
Bewertung/Kriterien		Bsp.: Plausibilitätsprüfungen, Zweitgutachten, Abstimmung mit Aktoren, Verständlichkeit, Angemessenheit, Vollständigkeit, Machbarkeitsprüfung, Usability-Tests, heuristische Evaluation, konzeptionelle Konsistenz	

ID	Kategorie	Prozess	Beschreibung
2.2.1	Analyse	Analyse der zu unterstützenden Didaktik/Methodik	Analyse des didaktischen Gesamtkonzeptes, des Curriculums und der Lernszenarien; Analyse didaktischer Modelle und Konzepte
Teilprozesse/Aspekte		Lernziele, Inhaltliches Konzept, Didaktik/Methodik (Lerntheoretische Grundmodelle, didaktische(s) Modell(e), Curriculum, Lernszenarien, Methodische Konzepte zum Methodeneinsatz, Sozialformen, Aktivitäten); Rollen und Aktivitäten, organisatorisches Konzept (Lernort, Lehrort, Lernzeitpunkt, Gesamtdauer, Summer der Lernzeit)	
Ziel		Übertragung der didaktischen Modelle und methodischen Konzepte auf die Domänenkonzepte; Sicherung einer gemeinsamen Kommunikationsbasis zwischen Beteiligten bzgl. Didaktik/Methodik	

Techniken	Interviews, Workshops, Brainstorming, Methoden der Aufgabenanalyse; Modelle des Instruktionsdesigns; IMS LD (EML); MOT-Diagramme; UML Aktivitätendiagramme; ClassSync Modeling Language (CML), CoopLets, DIN PAS 1032-2:2004
Ergebnis	Beschreibung der didaktischen Modelle und methodischen Konzepte in einer angemessenen Modellierung
Aktor	Didakt, Softwarearchitekt, Fachexperte
Bewertung/Kriterien	Lesbarkeit, Verständlichkeit, Nachvollziehbarkeit, Angemessenheit, Vollständigkeit

ID	**Kategorie**	**Prozess**	**Beschreibung**
2.2.2	Analyse	Anforderungserhebung: Medieneinsatz	Auswahl und Beschreibung der einzusetzenden Medien; Erweiterung des Pflichtenhefts
Teilprozesse/Aspekte		Präsentation und Distribution von Informationen; Sammeln und Filtern von Informationen; Bearbeitung von Informationen, Interaktion; konstruktive Darstellung eigener Lernergebnisse; “Performance-Support”-Werkzeuge, Kommunikation	
Ziel		Festlegung der Medienauswahl, Festlegung des Medieneinsatzes, der für Funktionen, Zielgruppe, Lernziele, Lerninhalte, Vorgaben und Rahmenbedingungen angemessen ist	
Techniken		Wirkungsanalysen, Auswertung von Wirkungsanalysen, Arbeits- und Lernplatzanalysen	
Ergebnis		Dokumentation und Begründung der Medienauswahl und des Medieneinsatzes	
Aktor		Mediendesigner; Didaktiker; Softwarearchitekt	
Bewertung/Kriterien		Prüfung der methodisch-didaktischen Angemessenheit und technischen Machbarkeit des gewählten Medieneinsatzes vor der Einführung	

ID	Kategorie	Prozess	Beschreibung
2.2.3	Analyse	Anforderungserhebung: Medien- u. Interaktionsdesign	Entwicklung und Beschreibung des Medien- und Interaktionsdesigns auf Basis der Vorgaben zu Software-Ergonomie, HCI-Design und Usability, sowie ggf. des Corporate Design der Organisation; Erweiterung des Pflichtenhefts
Teilprozesse/Aspekte			
Ziel		Festlegung des Design-Konzeptes für alle relevanten Bereiche unter Berücksichtigung bestehender Vorlagen/Richtlinien	
Techniken		Analyse bestehender Benutzungsschnittstellen hinsichtlich Interaktion und Design	
Ergebnis		Dokumentation und Begründung des Design-Konzeptes (Design-Prinzipien, Styleguide), Design-Prototyp	
Aktor		Mediendesigner, Didaktiker, Softwarearchitekt	
Bewertung/Kriterien		Usability-Test anhand des Design-Prototyps, heuristische Evaluation, Prüfung der Konformität bzgl. bestehender Vorgaben	

ID	Kategorie	Prozess	Beschreibung
2.2.4	Analyse	Normsprachliche Spezifizierung	Entlang der gewählten Nutzungsmetapher (hier: Wissensraummetapher) werden die zu unterstützenden Nutzungsszenarien mittels normsprachlicher Konstrukte spezifiziert.
Teilprozesse/Aspekte			**Statik**: Gruppen-/Nutzerstrukturen, Inhalte- und Prozessstrukturen; **Funktionalität**: Funktionen fachlicher Objekte und (kontextabhängige) Benutzer- und Gruppenrechte, diese auszuführen; **Dynamik**: Zustände und Zustandsübergänge fachlicher Objekte
Ziel		Formale Erfassung	
Techniken		Aussagenbildung auf Basis einer Normsprache, MOT-Diagramme	
Ergebnis		Methodeninvariante Spezifikation der zu unterstützenden Nutzungsszenarios	
Aktor		Softwarearchitekt	
Bewertung/Kriterien		Vollständigkeit, Widerspruchsfreiheit, Korrektheit, Nachvollziehbarkeit, Lesbarkeit, konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
2.3		Konzeption	Herleitung der Produkt-Architektur aus der SPL-Architektur, Identifizierung und Anpassung benötigter SPL-Komponenten, Entwurf produktspezifischer Komponenten, Konzeption von Systemeinführung, Betrieb u. Support, Wartung u. Pflege
Teilprozesse/Aspekte		Produktarchitektur, plattformspezifische Komponenten, produktspezifische Komponenten, organisatorische Konzepte für Systemeinführung, Betrieb und Support, Wartung und Weiterentwicklung	
Ziel		Beschreibung des SW-Entwurfs von Produktarchitektur u. -komponenten; Klärung und Sicherstellung von Systemeinführung, Betrieb, Support, Wartung und Pflege	
Techniken			
Ergebnis		Entwurfsdokumente für die produktspezifische Softwarearchitektur, die Integration und Anpassung von SPL-Komponenten, produktspezifische Komponenten, Konzept für Systemeinführung, Betrieb, Support, Wartung und Pflege	
Aktor		Softwarearchitekt; Softwareentwickler; Techniker; Organisationsspezialist; Projektmanager; Support	
Bewertung/Kriterien		Bsp.: Nachvollziehbarkeit, Lesbarkeit, konzeptionelle Integrität; Reibungslose Systemeinführung u. Betrieb; Benutzerbefragungen; Auswertung von Fehlerstatistiken u. Zufriedenheitsumfragen	

ID	Kategorie	Prozess	Beschreibung
2.3.1	Konzeption	Herleitung der Produktarchitektur aus der Plattformarchitektur	Instanzierung der Produktarchitektur; Beschreibung der Konfiguration und der produktspezifischen Anpassungen und Erweiterungen
Teilprozesse/Aspekte			
Ziel		Wiederverwendung der SPL-Architektur für das Produkt; Verringerung der Komplexität; Vorbereitung arbeitsteiligen Entwickelns	
Techniken		Brainstorming, Entwurfsmethoden, SW-/Architektur-Patterns, Entwurfsprinzipien, Diagrammsprachen, z.B. UML	
Ergebnis		Entwurfsdokumente für die produktspezifische Softwarearchitektur	
Aktor		Softwarearchitekt; Softwareentwickler	
Bewertung/Kriterien		Expertenaustausch, Nachvollziehbarkeit, Lesbarkeit, Grad der Anwendung/Anwendbarkeit von Standards, Wiederverwendbarkeit; konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
2.3.2	Konzeption	Identifizierung relevanter Komponenten und Anpassung	Abgleich der erhobenen Anforderungen mit dem Feature-Katalog und Konfigurationsmöglichkeiten der SPL-Komponenten
Teilprozesse/Aspekte			
Ziel		Wiederverwendung von SPL-Komponenten für das Produkt; Verringerung der Komplexität; Vorbereitung arbeitsteiligen Entwickelns	
Techniken		Brainstorming, Entwurfsmethoden, SW-/Architektur-Patterns, Entwurfsprinzipien, Diagrammsprachen, z.B. UML	
Ergebnis		Entwurfsdokumente für die Integration und die Anpassung von SPL-Komponenten innerhalb der produktspezifische Softwarearchitektur	
Aktor		Softwarearchitekt; Softwareentwickler	
Bewertung/Kriterien		Expertenaustausch, Nachvollziehbarkeit, Lesbarkeit, Grad der Anwendung/Anwendbarkeit von Standards, Wiederverwendbarkeit; konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
2.3.3	Konzeption	Entwurf der produktspezifischen Komponenten	Entwurf produktspezifischer Funktionen als Komponenten auf Basis erhobener Anforderungen
Teilprozesse/Aspekte			
Ziel		Verringerung der Komplexität; Vorbereitung arbeitsteiligen Entwickelns; Einbindung existierender Lösungen; Ermöglichung der Weiterentwicklung	
Techniken		Brainstorming, Entwurfsmethoden, SW-/Architektur-Patterns, Entwurfsprinzipien, Diagrammsprachen, z.B. UML	
Ergebnis		Entwurfsdokumentation für die produktspezifischen Komponenten	
Aktor		Softwarearchitekt; Softwareentwickler	
Bewertung/Kriterien		Expertenaustausch, Nachvollziehbarkeit, Lesbarkeit, Grad der Anwendung/Anwendbarkeit von Standards, Wiederverwendbarkeit; konzeptionelle Integrität	

ID	Kategorie	Prozess	Beschreibung
2.3.4	Konzeption	Konzeption Systemeinführung	Migrationsstrategie bei vorhandenem Altsystem; Kriterien zur Ersetzung des Altsystems; Festlegung von Verantwortlichkeiten für Übergabeentscheidung; Festlegung von Tätigen bei der Umstellung; Konzeption eines Notfallplans für einen etwaigen Systemausfall
Teilprozesse/Aspekte			
Ziel		Klärung und Sicherstellung einer problemlosen Systemeinführung	
Techniken		Migrationsstrategien (bspw. "Database First" "Database Last" "Cold Turkey/Big-Bang"); Checklisten; Aktivitätendiagramme/EPKs; Einrichtung einer TaskForce; Informationsveranstaltungen; Workshops	
Ergebnis		Konzept für die Systemeinführung	
Aktor		Techniker; Organisationsspezialist; Softwarearchitekt; Softwareentwickler, Projektmanager	
Bewertung/Kriterien		Reibungslose Systemeinführung	

ID	Kategorie	Prozess	Beschreibung
2.3.5	Konzeption	Konzeption Betrieb und Support	Konzeption für den First- und Second Level Support; Konzeption für Endanwenderschulungen
Teilprozesse/Aspekte		**Organisationsform**: Versorgungsträger, Kooperationsformen, Leitungsstrukturen; **Aufgabenverteilung**: Zuordnungsprinzipien, Aufgabenverteilung zwischen IT-Diensten, Mediendiensten und Bibliotheken	
Ziel		Klärung und Sicherstellung von Support und Schulungen	
Techniken		Versorgungskonzepte; Best Practices; Lessons learned; Workshops; Moderation; Mediation, Methoden der Entscheidungsfindung; Organigramme	
Ergebnis		Konzept für den Betrieb und den Support; Schulungskonzept	
Aktor		Support; Techniker; Projektmanager; Softwarearchitekt; Softwareentwickler; Didaktiker; Fachexperte	
Bewertung/Kriterien		Auslastung des technischen Supports; Nachfrage nach didaktischem Support; Ausfallsicherheit;	

ID	Kategorie	Prozess	Beschreibung
2.3.6	Konzeption	Konzeption Wartung und Pflege	Konzeption der Pflege und Aktualisierung der Lernressourcen: Verantwortung für Wartung und Anpassung; methodisch-didaktische Aktualisierung; inhaltliche Aktualisierung; Existenz und Laufzeit von Wartungsverträgen, Verantwirtlichkeit für Änderungsanträge; Kriterien für durchzuführendeÄnderungen; Verfahren bei Änderungen; Kontrollverfahren
Teilprozesse/Aspekte			
Ziel			Klärung und Sicherstellung von Gültigkeit und Aktualität des Produktes und der Inhalte bei geänderten Rahmenbedingungen oder Anforderungen
Techniken			ISO 9001, Workflow-Modellierung, Checklisten, Wartungspläne und Vereinbarungen
Ergebnis			Konzept der Verantwortlichkeiten und Maßnahmen für Wartung und Pflege
Aktor			Techniker; Support; Softwarearchitekt; Softwareentwickler; Projektmanagement
Bewertung/Kriterien			Reibungsloser Betrieb und aktuelle Inhalte; Auswertung von Fehlerstatistiken und Zufriedenheitsumfragen, Benutzerbefragungen

ID	Kategorie	Prozess	Beschreibung
2.4		Implementierung, Integration und Test	Umsetzung der Produktarchitektur und produktspezifischer Komponenten, Wiederverwendung von SPL-Komponenten; Erweiterung der komponententübergreifenden Benutzungsschnittstelle und Integrationstests
Teilprozesse/Aspekte			Produktarchitektur, Konfiguration von Komponenten, Implementierung produktspezifischer Komponenten, Medien- u. Interaktionsdesign, Integrationstests
Ziel			Umsetzung der technischen und mediendidaktischen Konzeption
Techniken			
Ergebnis			Getestetes technisch funktionales System und dazugehörige Dokumentation
Aktor			Softwareentwickler; Softwarearchitekt; Mediendesigner; Support; Techniker
Bewertung/Kriterien			Bsp.: Expertenreviews, Testbarkeit, Korrektheit, Bedienbarkeit, Barrierefreiheit, Browserkompatibilität

ID	Kategorie	Prozess	Beschreibung
2.4.1	Implementierung	Umsetzung der Produktarchitektur	Umsetzung des Entwurfs der produktspezifischen Architektur; Installation und Integration verschiedenster Technologien: Basisinfrastrukturkomponenten, IDEs, Frameworks, Schnittstellen u. Protokolle
Teilprozesse/Aspekte			
Ziel		Umsetzung des Architekturentwurfs zu einem lauffähigen System	
Techniken		allg. Standards, anerkannte Methoden und Tools für das SW-Engineering: Booch, Nassi-Shneidermann oder Jackson, CASE, UML; Unit Tests	
Ergebnis		Produktspezifisches Rahmenwerk für SPL- und produktspezifische Komponenten; Rahmenwerk für Benutzungsschnittstelle; Dokumentation	
Aktor		Softwareentwickler; Softwarearchitekt	
Bewertung/Kriterien		Dokumentation, Expertenreviews, Testbarkeit, Korrektheit, Stabilität, Lesbarkeit, Zweckmäßigkeit, Erweiterbarkeit, Performance, Programmierrichtlinien	

ID	Kategorie	Prozess	Beschreibung
2.4.2	Implementierung	Instanzierung und Modifikation von Komponenten	Konfiguration von SPL-Komponenten und ihre Einbindung in die Produktarchitektur; Umsetzung der produktspezifischen Benutzungsoberfläche für SPL-Komponenten
Teilprozesse/Aspekte			
Ziel		Wiederverwendung von SPL-Komponenten in der Produktarchitektur; Integration der Komponenten	
Techniken		allg. Standards, anerkannte Methoden und Tools für das SW-Engineering: Booch, Nassi-Shneidermann oder Jackson, CASE, UML; Units Tests, HTML-Editoren, CSS-Werkzeuge, Checklisten, Interaktionsgraphen, Templates	
Ergebnis		Funktionale Erweiterung des Produktes durch SPL-Komponenten; Dokumentation	
Aktor		Softwareentwickler; Mediendesigner; Softwarearchitekt	
Bewertung/Kriterien		Dokumentation, Expertenreviews, Testbarkeit, Korrektheit, Stabilität, Lesbarkeit, Zweckmäßigkeit, Erweiterbarkeit, Performance, Programmierrichtlinien	

ID	Kategorie	Prozess	Beschreibung
2.4.3	Implementierung	Implementierung produktspezifischer Komponenten	Implementierung spezieller Funktionen als produktspezifische Komponenten; Integration in die Produktarchitektur (evtl. über Plugin-Mechanismus); Umsetzung der Benutzungsoberfläche für produktspezifische Komponenten
Teilprozesse/Aspekte		Umsetzung der Datenmodelle; Implementierung des Application Layers; Implementierung der Controller; Umsetzung des Medien- u. Interaktionsdesigns (HTML, AJAX, CSS, JS, etc.); Anpassung der Sprachdateien	
Ziel		Umsetzung des Komponenten-Entwurfs zu lauffähigen Komponenten; Integration der Komponenten	
Techniken		allg. Standards, anerkannte Methoden und Tools für das SW-Engineering: Booch, Nassi-Shneidermann oder Jackson, CASE, UML; Units Tests, HTML-Editoren, CSS-Werkzeuge, Checklisten, Interaktionsgraphen, Templates	
Ergebnis		Funktionale Erweiterung des Produktes durch produktspezifische Komponenten; Dokumentation	
Aktor		Softwareentwickler; Mediendesigner; Softwarearchitekt	
Bewertung/Kriterien		Dokumentation, Expertenreviews, Testbarkeit, Korrektheit, Stabilität, Lesbarkeit, Zweckmäßigkeit, Erweiterbarkeit, Performance, Programmierrichtlinien	

ID	Kategorie	Prozess	Beschreibung
2.4.4	Implementierung	Umsetzung Medien- u. Interaktionsdesign	Umsetzung von Navigationsstrukturen; Umsetzung des Design-Konzeptes
Teilprozesse/Aspekte			
Ziel		Umsetzung des Design-Konzeptes für alle relevanten Bereiche	
Techniken		HTML-Editoren, CSS-Werkzeuge, Checklisten, Interaktionsgraphen, Templates	
Ergebnis		Benutzungsschnittstelle für das Produkt; Dokumentation	
Aktor		Softwareentwickler; Mediendesigner; Softwarearchitekt;	
Bewertung/Kriterien		Design-Prinzipen; Expertenreviews; Bedienbarkeit; Barrierefreiheit; verwendete Standards; Browserkompatibilität	

ID	Kategorie	Prozess	Beschreibung
2.4.5	Implementierung	Integrationstest	Aufeinander abgestimmte Reihe von einzelnen Tests, um alle voneinander abhängigen Komponenten im Zusammenspiel miteinander zu testen
Teilprozesse/Aspekte		Test der Benutzungsschnittstelle, Test der Komponentenschnittstellen im Zusammenspiel mit den neuen produktspezifischen Komponenten, Test der Kommunikationsschnittstelle mit dem Basissystem	
Ziel		Sicherstellung des fehlerfreien Zusammenspiels aller Komponenten	
Techniken		Funktionstests, Schnittstellentests, "Bottom-Up-Methode" (Betrachtung bereits getesteter Komponenten als ein System)	
Ergebnis		Getestetes technisch funktionales System; Dokumentation	
Aktor		Softwareentwickler; Softwarearchitekt; Support; Techniker	
Bewertung/Kriterien		Prüfung auf Fehlerfreiheit; Korrektheit, Vollständigkeit der Testfälle; Automatisierbarkeit	

ID	Kategorie	Prozess	Beschreibung
2.5		Systemeinführung u. Abnahme	Überführung der Lernressourcen von der Entwicklungs- in die Betriebsumgebung
Teilprozesse/Aspekte		Einrichtung der technischen Infrastruktur, Systemtest, Anpassung, Bereitstellung, Abnahme- oder Übernahmetest, Organisation des Betriebs und der Nutzung	
Ziel		Einführung der technologischen Komponenten in den Bildungsprozess	
Techniken			
Ergebnis		Betriebsbereite Lernumgebung mit allen für den Bildungsprozess notwendigen Lernressourcen	
Aktor		Techniker; Softwareentwickler; Softwarearchitekt; Projektmanager; Didaktiker; Support; Support	
Bewertung/Kriterien		Bsp.: Auditierung der technischen Infrastruktur, der Testergebnisse u. Dokumentation, sowie der Aufbau- u. Ablauforganisation durch ein Kontrollgremium; Abnahmeprotokoll	

ID	Kategorie	Prozess	Beschreibung
2.5.1	Systemeinführung u. Abnahme	Einrichtung der technischen Infrastruktur	Installation, Konfiguration und Inbetriebnahme der technischen Basisinfrastruktur, die das neue System voraussetzt. Einrichtung von Schutzmßnahmen (bspw. Sicherheitskonzept, Datensicherung)
Teilprozesse/Aspekte		Sicherstellung von Infrastrukturkomponenten, die den Anforderungsspezifika entsprechen, z. B. Hardware-, Software-, und Netzwerkkomponenten bei Betreiber und Benutzer; Maßnahmen zur Ausfallsicherung, sicherem Zugriff und sicheren Transaktionen; Bereitstellung von Dokumentationen und Anleitungen für den Betrieb, den Zugriff und die Transaktionssicherheit	
Ziel		Schaffung der technischen Voraussetzungen anhand der Anforderungen für die Nutzung des SPL-Produktes	
Techniken		Soll-Ist-Vergleich, Implementierungsstrategie	
Ergebnis		Anforderungsgerechte technische Infrastruktur; Dokumentation der technischen Realisierung/Realisierbarkeit	
Aktor		Techniker; Softwareentwickler; Softwarearchitekten	
Bewertung/Kriterien		Auditierung der technischen Infrastruktur durch ein Kontrollgremium	

ID	Kategorie	Prozess	Beschreibung
2.5.2	Systemeinführung u. Abnahme	Systemtest	Überprüfung und Validierung der Lernressourcen
Teilprozesse/Aspekte			
Ziel		Feststellung, ob die Lernressource die definierten Anforderungen erfüllen und für eine endgültige Freigabe geeignet sind	
Techniken		Software Validation, z.B. nach IEEE; ISO 9000; Code Review; Moduldesigntest; Belastungstest; Integrationstest; Usabilitytest	
Ergebnis		Dokumentierte Testergebnisse und Änderungsanforderungen	
Aktor		Projektmanager; Techniker; Softwareentwickler; Softwarearchitekt; Didaktiker	
Bewertung/Kriterien		Bewertung der Kriterien durch ein Kontrollgremium	

ID	Kategorie	Prozess	Beschreibung
2.5.3	Systemeinführung u. Abnahme	Anpassung	Sicherstellung der Angemessenheit und Nachvollziehbarkeit der Anpassungen hinsichtlich Funktionalität, Gestaltung und Dokumentation
Teilprozesse/Aspekte			
Ziel		Sicherstellung der Umsetzung aller Anforderungen für Anpassungen an einer Lernressource bzw. einem definierten System in einer effektiven und kontrollierten Weise	
Techniken		Konfigurationsmanagement mit dokumentierter Programmversionskontrolle des archivierten Codes und der definierten Hardware- und Netzwerkkonfiguration; Dokumenversionskontrolle unter Einsatz eines Dokumenten- oder Contentmanagementsystems	
Ergebnis		Angepasste Lernressource (Update); Dokumentation der Anpassungen, z. B. Konfigurationsmanagementplan, Anpassungsplan, Dokumentationsplan, bearbeitete Anpassungsformulare, Design Review Bericht	
Aktor		Projektmanagement, Softwareentwickler; Softwarearchitekt; Techniker; Support; Mediendesigner	
Bewertung/Kriterien		Bewertung der Konfiguration und der Anpassungskontrolle durch ein Kontrollgremium	

ID	Kategorie	Prozess	Beschreibung
2.5.4	Systemeinführung u. Abnahme	Bereitstellung	Übergabe des Produkts und der Dokumentation an den Betreiber
Teilprozesse/Aspekte		Registrierung von Lizenzen, Installation und Konfiguration der getesteten Infrastruktur und des SPL-Produktes in der Betriebsumgebung, Dokumentation, Datenübernahme aus Altsystemen, Deaktivierung und/oder Entfernen von alten Versionen (Updating)	
Ziel		Bereitstellung des Systems in der Betriebsumgebung	
Techniken		Checklisten; Freischaltung im Netzwerk/Übergabe (z.B. Server)	
Ergebnis		Verfügbares funktionierendes System in der Betriebsumgebung;	
Aktor		Techniker; Support; Projektmanager; Softwareentwickler; Softwarearchitekt	
Bewertung/Kriterien		Bewertung der Kriterien durch ein Kontrollgremium; Dokumentation	

ID	Kategorie	Prozess	Beschreibung
2.5.5	Systemeinführung u. Abnahme	Abnahme- oder Übernahmetest	
Teilprozesse/Aspekte		Übernahme des Produkts und der Dokumentation durch den Betreiber und formale Freigabe	
Ziel		Überführung des Systems in den Betrieb	
Techniken		Soll-Ist-Vergleich; Funktions- u. Schnittstellentests; Belastungstests; Installationstest; User-Akzeptanztest	
Ergebnis		Abgenommenes und formal freigegebenes System in der Betriebsumgebung; Dokumentierter Nachweis der Erfüllung der Anforderungen	
Aktor		Projektmanager; Techniker; Initiator; Support; Didaktiker	
Bewertung/Kriterien		Abnahmeprotokoll; Abgleich mit dokumentierten Anforderungen; Bewertung der Testergebnisse und der Dokumentation durch ein Kontrollgremium	

ID	Kategorie	Prozess	Beschreibung
2.5.6	Systemeinführung u. Abnahme	Organisation des Betriebs u. der Nutzung	Schaffung der organisatorischen Voraussetzungen anhand der Konzepte für Systemeinführung, Betrieb, Support, Wartung und Pflege
Teilprozesse/Aspekte		Organisationsstruktur, Qualifikation des Lehr-/Betriebspersonals, Bedienkompetenz und Vorwissen des Nutzers, Support und Wartung	
Ziel		Sicherstellung des organisatorischen Betriebs der Lernumgebung	
Techniken		Workshops; Informationsveranstaltungen; Qualifizierungsmaßnahmen; Anwendung von Organisationskonzepten; Methoden des Change-Management; ISO 9001; ITIL	
Ergebnis		Organisatorischer Rahmen mit Aufbau- und Ablauforganisation	
Aktor		Projektmanagement; Support; Techniker; Organisationsspezialist	
Bewertung/Kriterien		Validierung der Aufbau- und Ablauforganisation durch ein Kontrollgremium	

ID	Kategorie	Prozess	Beschreibung
2.6		Betrieb	Durchführung und Nutzung eines Bildungsangebots: Administration, Support und Wartung
Teilprozesse/Aspekte		Administration, Support, Wartung und Weiterentwicklung	
Ziel		Bereitstellung von Administration, Support und Wartung	

Techniken	
Ergebnis	Durchführung und Dokumentation von Administration, Support und Wartung
Aktor	Techniker; Support; Softwarearchitekt; Softwareentwickler
Bewertung/Kriterien	Bsp.: Lern- und Transfererfolg; Kompetenzzuwachs; Funktional: Korrektheit, Benutzbarkeit; Nicht-funktional: Robustheit, Performance , Ausfallsicherheit; Reaktionszeit

ID	**Kategorie**	**Prozess**	**Beschreibung**
2.6.1	Betrieb	Administration	Bereitstellung der Administration und der begleitenden Maßnahmen; Benutzerverwaltung; Verwaltung und Freigabe von Inhalten (insb. Verwaltung organisatorischer Strukturen wie Semester, Kurse, Rollen und Rechte)
Teilprozesse/Aspekte		Vergabe von Rollen und Rechten, Sperrung/Löschung von Logins, Einrichtung und Freigabe von Inhalten (Semester, Kurse, Griuppen, etc.)	
Ziel		Bereitstellung der Administration und begleitender Maßnahmen für die Benutzung des Produktes	
Techniken		ISO 9000; entsprechend dem organisatorischen, ökonomischen und technologischen Betriebskonzept (Technischer Support, Organisatorischer Rahmen); ITIL	
Ergebnis		Durchführung und Dokumentation der Administration und der begleitenden Maßnahmen	
Aktor		Techniker; Support	
Bewertung/Kriterien		Evaluation erfolgt in Abhängigkeit von den gewählten Standards formativ bzw. summativ; gesetzliche Vorschriften (z. B. Datenschutz)	

ID	**Kategorie**	**Prozess**	**Beschreibung**
2.6.2	Betrieb	Support	Beratung und Hilfestellung der Benutzer
Teilprozesse/Aspekte		Information/Ankündigung, Lehr-/Lernberatung: Technologie/Lernorganisation, techn. und/oder didaktische Produktschulungen, Beschwerdemanagement, Buchung/Einschreibung/Anmeldung	
Ziel		Bereitstellung von Beratungsleistungen und Hilfestellung für Anwender bei der Benutzung des Systems	
Techniken		Ticketsystem; Einrichtung einer Hotline/eines Helpdesks; ITIL	
Ergebnis		Durchführung und Dokumentation der Beratung und Hilfestellung	
Aktor		Support	
Bewertung/Kriterien		Benutzerbefragung; Reaktionszeit; Individualität	

ID	Kategorie	Prozess	Beschreibung
2.6.3	Betrieb	Wartung und Weiterentwicklung	Veränderung des Produktes nach Auslieferung, zwecks Fehlerbehebung, Leistungssteigerung oder Anpassung an veränderte Rahmenbedingungen
Teilprozesse/Aspekte		**Softwarewartung**: Durchsicht der Log-Dateien; Updates einspielen; Sicherheitspatches; Hardware; **Softwareweiterentwicklung**: Strukturverbesserungen, funktionale Erweiterungen, Korrekturen/Bugfixing	
Ziel		Sicherstellung der Aktualität des Systems bzw. geänderter/neuer Anforderungen; Erhöhung der Ausfallsicherheit	
Techniken		Konzepte zur Softwarepflege/-wartung/-weiterentwicklung: Korrigierende Wartung, präventieve Wartung, adaptive Wartung, perfektionierende Wartung; Re-Engineering; IEEE 1219-98; ITIL	
Ergebnis		Durchführung und Dokumentation der Wartung	
Aktor		Techniker; Softwarearchitekt; Softwareentwickler	
Bewertung/Kriterien		Kompatibilität; Konsistenz; Ausfallzeiten; Reaktionszeit	

ID	Kategorie	Prozess	Beschreibung
2.7		Evaluation	Systematische Untersuchung der Verwendbarkeit bzw. Güte eines Bildungsangebotes
Teilprozesse/Aspekte		Planung, Durchführung, Auswertung, Optimierung	
Ziel		Evaluation und Verbesserung des Bildungsangebots und der Bildungsprozesse	
Techniken		ISO 9000; Evaluationsmodelle, z.B. Kirkpatrick; Summative und formative Evaluationsmethoden und -verfahren	
Ergebnis		Evaluationsplanung, Evaluationsberichte, Optimierungskonzept	
Aktor		Projektmanager; Support; Evaluationsexperte	
Bewertung/Kriterien			

ID	Kategorie	Prozess	Beschreibung
2.7.1	Evaluation	Planung	Parameter, Kriterien, Instrumente und Methoden sowie der organisatorischen Rahmenbedingungen zur Durchführung einer Evaluation
Teilprozesse/Aspekte		Zielsetzung, Art der Evaluation, Zeitraum/Anzahl der Evaluation(en), Festlegung von Parametern und Kriterien, Auswahl von Instrumenten und Methoden zur Evaluation	
Ziel		Konzeption der Evaluation im Hinblick auf Wirksamkeit und Wirtschaftlichkeit des Bildungsangebotes	
Techniken		Normen, unternehmensinterne/-übergreifende Verfahrensanweisungen; Evaluationsmodelle, z.B. Kirkpatrick	
Ergebnis		Dokumentation und Begründung des Evaluationsplans zur Durchführung der Evaluation	
Aktor		Evaluationsexperte; Initiator; Projektmanagement	
Bewertung/Kriterien		Durch Erfahrungsauswertungen nach der Durchführung der Evaluation	

ID	Kategorie	Prozess	Beschreibung
2.7.2	Evaluation	Durchführung	Umsetzung des Evaluationsplans
Teilprozesse/Aspekte		Realisation, Erfassung von Messdaten	
Ziel		Ermittlung von Daten zur Ergebniskontrolle und zur Kontrolle der eingesetzten Mittel und Ressourcen, z.B. im Kontext von Aufwand, Eignung und Wirksamkeit	
Techniken		Normen, unternehmensinterne/-übergreifende Verfahrensanweisungen; Einsatz der Instrumente und Methoden aus dem Evaluationsplan	
Ergebnis		Dokumentation der Umsetzung, Dokumentation der Messdaten	
Aktor		Support; Evaluationsexperte	
Bewertung/Kriterien		Erfahrungsauswertung nach der Durchführung der Evaluation	

ID	Kategorie	Prozess	Beschreibung
2.7.3	Evaluation	Auswertung	Auswertung der ermittelten Messdaten
Teilprozesse/Aspekte		Zusammenfassung, Analyse, Interpretation, Empfehlung	
Ziel		Gewinnung von Erkenntnissen über Methoden, eingesetzte Mittel und Ressourcen und erzielte Ergebnisse; Schlussfolgerungen zu Aufwand, Eignung und Wirksamkeit	
Techniken		Normen, unternehmensinterne und -übergreifende Verfahrensanweisungen; Beschreibung und Begründung der verwendeten Auswahlmethoden	
Ergebnis		Dokumentation und Begründung der Auswertungsergebnisse in einer Zusammenfassung; Dokumentation und Begründung von Analyse und Interpretation der Auswertungsergebnisse; Dokumentation und Begründung von Empfehlungen	
Aktor		Evaluationsexperte	
Bewertung/Kriterien		Erfahrungsauswertung nach der Durchführung der Evaluation; Soll-Ist-Vergleich nach Durchführung der Evaluation	

ID	Kategorie	Prozess	Beschreibung
2.7.4	Evaluation	Optimierung	Verbesserung von Produkten und Prozessen
Teilprozesse/Aspekte		**vorbeugende Maßnahmen**: Erfassung, Analyse, Umsetzung geänderter Rahmenbedingungen, **korrektive Maßnahmen**: Erfassung, Analyse, Umsetzung geänderter Rahmenbedingungen	
Ziel		Steigerung oder Erhaltung der Wirksamkeit und Wirtschaftlichkeit von Produkten und Prozessen	
Techniken		vorbeugende Maßnahmen zur Aktualisierung und Anpassung der Dimensionen Verfahren, Zeit, Personen und Inhalte an geänderte Rahmenbedingungen; Korrektve Maßnahmen zur Umsetzung von Empfehlungen aus der Evaluation zu den Dimensionen Verfahren, Zeit, Personen und Inhalte	
Ergebnis		Dokumentation der durchgeführten Optimierungsmaßnahmen; Steigerung oder Erhaltung der Wirksamkeit und Wirtschaftlichkeit von Produkten und Prozessen	
Aktor		Projektmanagement; Evaluationsexperte	
Bewertung/Kriterien		Erfahrungsauswertung nach der Durchführung der Evaluation	