Analyse und objektorientierter Entwurf eines integrierten Portalsystems für das Wissensmanagement

Dem Promotionsausschuß der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften
vorgelegte Dissertation

von

Holm Wegner

aus Hamburg

24. Januar 2002

Zusammenfassung

Die Industriegesellschaft verwandelt sich zunehmend in eine Wissensgesellschaft und so wird Wissensmanagement zu einem entscheidenden Wettbewerbsfaktor. Im Mittelpunkt dieser Arbeit steht die informationstechnische Unterstützung durch Internet-basierte Wissensportale.

Als Ausgangspunkt der Untersuchung werden verschiedene Arten von Informationssystemen beschrieben und klassifiziert, die Beiträge zur Unterstützung des Wissensmanagements versprechen. Dazu zählen die Systemklassen der Dokumenten-Management- und Content-Management-Systeme (DMS und CMS) sowie Unternehmensportale. Dabei konnte die im Bereich der Portale sonst sehr uneinheitlich gebrauchte Terminologie konsolidiert und eine Taxonomie von Portalklassen eingeführt werden.

Die zentrale Erkenntnis dieser Untersuchung ist, daß im Aufgabenbereich der Portale für das Wissensmanagement mit der Kernanforderung der Integration von Informationsbeständen aus verschiedensten Informationssystemen ein problemadäquates Dokumentenmodell fehlt, das die uniforme Repräsentation und Nutzung unterschiedlichster Informationsarten (unstrukturiert, semistrukturiert, strukturiert) zur Erschließung von explizitem und implizitem Wissen ermöglicht. Neben dieser Lücke wurde durch die Betrachtung der Systemklassen der DMS, CMS und Portale sowie ausgewählter konkreter Systeme deutlich, daß die Abdeckung der fachlichen Anforderungen gerade im Bereich der Wissensportale noch relativ gering ist und mit einem breiteren Ansatz unter Nutzung eines integrierten Dokumentenmodells deutlich umfassender gestaltet werden kann.

Als erster Lösungsbeitrag dieser Arbeit wurde dazu zunächst ein feingranular strukturierter Katalog von fachlichen Anforderungen an die Funktionalität und an die Benutzungsoberfläche eines Wissensportals aufgestellt. Er wurde durch wichtige Anforderungen an das Systemmanagement ergänzt, um dem Integrationsgedanken auf dieser Ebene Rechnung zu tragen. Dieser Katalog zeigt, daß die Fülle der Anforderungen erst mit Hilfe eines uniformen Dokumentenmodells adäquat umgesetzt werden kann, das die Nutzbarmachung und tiefe Erschließung aller Informationsartefakte erlaubt. Außerdem wird deutlich, daß für alle Aufgaben ein konsistenzerhaltendes und bidirektionales Linkmanagement zwingend nötig ist, das dann integraler Bestandteil des Dokumentenmodells sein muß.

Der zweite Lösungsbeitrag dieser Arbeit besteht aus der Entwicklung eines problemadäquaten Dokumentenmodells für die Aufgaben eines integrierenden Wissensportals. Dazu wurden zunächst die von den klassischen Software- und Informationssystemen verwendeten Datenmodelle analysiert, bewertet und als Synthese das Modell der *Information Assets* entworfen. Es zeichnet sich durch die flache Aggregation von Attributen aus, die es dem relationalen Modell sehr ähnlich macht, übernimmt aber zugleich die Vorteile der Vererbung von objektorientierten Modellen und die Dynamik und Reflexivität von dokumentenorientierten Modellen. Dieses eigentlich als Metamodell zu bezeichnende Modell stellt das Bindeglied zwischen dem konzeptuellen Modell im objektorientierten Entwurf und seiner Implementierung auf der einen Seite und den verschiedenen zu integrierenden Systemen und Modellen auf der anderen Seite dar und federt so die Modell- und Medienbrüche wirkungsvoll ab. Neben dem Dokumentenmodell wurde unter Bezugnahme auf das aus dem Bereich der kooperativen Informationssysteme stammende Modell der *Business Conversations* ein Prozeßmodell formuliert, das als wesentliche Abstraktion eine Kunde-Dienstleisterbeziehung in Form einer durch formale Spezifikationen ermöglichten Konversation besitzt. Die medien- und aktorenunabhängige Modellierung der Konversation nimmt dabei eine zentrale Position ein, die zu der Kapselung der Anwendungslogik in rollenbasierten Regeln führt.

Als dritter Lösungsbeitrag wurde der objektorientierte Entwurf eines Wissensportalsystems vorgenommen. Eine der wesentlichen Entwurfsentscheidungen war dabei, das gesamte System vollständig neu zu entwerfen und nicht durch Zusammenstellung bestehender Softwarekomponenten aufzubauen, um eine möglichst bruchlose Umsetzung der Konzepte des Dokumenten- und Prozeßmodells zu erreichen. Der Entwurf der mehrschichtigen Client/Server-Architektur

wird detailliert beschrieben und durch zahlreiche UML-Diagramme illustriert. Schwerpunkte des entstandenen Entwurfs sind:

- Die Umsetzung des medien- und aktorenunabhängigen Konzepts des Prozeßmodells in der Kommunikationsschicht durch Server-Module und Geräteklassen. Es erlaubt die ökonomische Bedienung einer großen Anzahl von Endgerätetypen mit unterschiedlichen Eigenschaften, den simultanen Betrieb von mehreren insbesondere bezüglich der Rechte frei konfigurierbaren HTTP-Server-Modulen, e-Mail-Servern, FTP-Servern etc.
- Die strikte Trennung von Inhalt, Gestaltung und Funktionalität durch die Konzepte der Vorlagen, Platzhalter und Regeln auf der Interaktionsschicht. Dies stellt einen deutlichen Fortschritt gegenüber anderen gebräuchlichen Mechanismen dar und wurde durch eine sowohl entwurfstechnisch als auch programmiersprachlich elegante Lösung umgesetzt, die die Wiederverwendung von Regeln und Vorlagen in großem Umfang ermöglicht und in konkreten Projekten bereits eine hohe Produktivität zur Folge hatte.
- Die Umsetzung des objektorientiert beschriebenen konzeptuellen Modells durch eine reichhaltige Schnittstelle und ihre Implementierung auf der Ebene der Dienstschicht. Diese stellen sowohl eine klassische objektorientierte Schnittstelle für die Interaktionsschicht dar als auch eine Umsetzung des uniformen und reflexiven Dokumentenmodells der *Information Assets*. Die in der Praxis erprobten und leistungsfähigen Implementierungen umfassen u.a. konsistenzsichernde Automatismen für bidirektionale Verknüpfungen, ein effektives Konzept zur Sicherstellung der Konsistenz bei nebenläufigen Zugriffen und die Anbindung eines generischen Klassifikations-Rahmenwerks.
- Ein Modell für die Speicherungsschicht, das aus abstrakt beschriebenen Managern, Containern und Inhaltsobjekten gemäß eines explizit konstruierten, dynamischen Inhaltstyps besteht. Die für die wichtigsten Informationssysteme (Datenbanken, Dateisysteme etc.) vorhandenen Implementierungen ermöglichen die Integration bestehender Informationsquellen und die Nutzung verschiedenster Systeme zur Informationsablage. Das abstrakte Inhaltsmodell erlaubt die effiziente, bruchlose und transparente Abbildung des Modells *Information Assets* auf die jeweiligen Persistenzsysteme. Von großem Vorteil ist die Kombinierbarkeit und Austauschbarkeit der verschiedenen Implementierungen in wechselnden Einsatzszenarien. Erst dadurch wird der Integrations- und Skalierungsgedanke wirkungsvoll umgesetzt.

Insgesamt gesehen wurde durch die Bruchlosigkeit des Entwurfs eine enge und effizienzbewahrende Kopplung aller Schichten erreicht – beginnend bei den Strukturen der Speicherungsschicht über die generischen *Asset*-Implementierungen bis hin zu den Regeln, Vorlagen und Server-Modulen. Diese zentrale Errungenschaft des entstandenen Systems rechtfertigt den vollständigen Neuentwurf eindeutig.

Folgende quantitative Aussagen über das bestehende System lassen sich abschließend treffen: Es ist vollständig in JAVA unter Nutzung des *Java Development Kit* 1.2.2 entwickelt worden. Derzeit existieren Anbindungen der Speicherungsschicht an drei relationale Datenbanksysteme, eine persistente hauptspeicherbasierte Datenbank, Dateisysteme und das kommerzielle CMS COREMEDIA. Das implementierte System ist relativ kompakt und besteht aus derzeit etwas weniger als 900 JAVA-Klassen. Im normalen Betrieb bietet das System durchweg sehr gute Antwortzeiten; der typische Abruf einer dynamisch generierten Seite dauert zwischen 200 ms und 1000 ms und liegt im Mittel unter 500 ms, was für ein System mit dynamischer Seitengenerierung ein äußerst guter Wert ist. Das Serversystem selbst ist sehr kompakt und benötigt mit minimal 32 MB vergleichsweise wenig Hauptspeicher.

Einsatzbeispiele des entstandenen Systems in verschiedenen Projekten und einige Erfahrungsberichte verdeutlichen zudem nicht nur die Erreichung der technisch wünschenswerten Eigenschaften wie Stabilität, Effizienz und Sicherheit, sondern dokumentieren genauso die Erfüllung der aufgestellten fachlichen Anforderungen.

Danksagung

Das Gelingen dieser Arbeit wäre nicht möglich gewesen ohne die Hilfe einer ganzen Reihe von Personen, denen ich an dieser Stelle dafür danken will.

Vorbemerkung

Auch wenn es notwendig ist, im Bereich der Informatik eine international einheitliche Fachsprache – die eben englisch ist – zu entwickeln und zu benutzen, ist die übermäßige Verwendung dieser Fachausdrücke der Lesbarkeit deutscher Arbeiten abträglich. Vielfach ist die unkritische und unnötige Verwendung englischer (Fach)begriffe und die Verwendung vieler englisch-deutscher Komposita zu beobachten. Einige lesenswerte Überlegungen dazu finden sich auch in [Bal96].

Andererseits wird es immer schwieriger, deutschsprachige Arbeiten in Informatik zu schreiben, da die Anzahl der englischen Fachbegriffe ständig steigt, deutsche Entsprechungen teilweise gar nicht mehr existieren und bestimmte Bereiche von Begriffen mittlerweile sogar starken Eingang in die Allgemeinsprache finden. Dies ist gerade im Umfeld aller Themen des Internet besonders auffällig. Deshalb wird auch die bisher häufig praktizierte Lösung, englische Begriffe wenigstens durch andere Schriftarten auszuzeichnen, immer problematischer, denn es leidet das Schriftbild der puren Menge der Begriffe wegen darunter und zusätzlich wird es eben immer fraglicher, ob es überhaupt ein Fremdwort ist, das besonderer Auszeichnung bedarf.

Als Zugeständnis an die sich weiterentwickelnde Sprache werden in dieser Arbeit einige Begriffe, für die es keine oder nur umständliche deutsche Entsprechungen gibt, nicht weiter ausgezeichnet. Dazu zählen Begriffe wie Internet, Intranet, Server, Client, e-Mail und daraus abgeleitete wie Web-Server oder Web-Browser, die auch schon de facto zum normalen Sprachgebrauch gehören. Sie werden konsequenterweise groß geschrieben. Für alle anderen wird folgendermaßen verfahren: Für die meisten englischen Begriffe existieren deutsche Synonyme, die stattdessen verwendet werden; im Zweifel wird das Fremdwort zusätzlich in Klammern angegeben. Läßt es sich in Ermangelung einer guten deutschen Entsprechung nicht vermeiden, so werden englische Fremdwörter auch im laufenden Text verwendet. In allen Fällen jedoch werden sie durch eine *geneigte* Schriftart abgehoben. Außerdem werden neu eingeführte oder besonders wichtige deutsche Begriffe *kursiv* ausgezeichnet; Firmen- und Produktnamen sowie Personennamen werden in KAPITÄLCHEN gesetzt. Um ein ausgeglichenes Schriftbild zu erhalten und damit die Lesbarkeit des Textes zu erhöhen, wird nur noch eine weitere Schriftart verwendet: Inhärent englische Bezeichnungen wie Klassennamen und Programmfragmente sowie Schlüsselwörter von Programmiersprachen werden durchgängig in `Schreibmaschinenschrift` gesetzt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Thou art not for the fashion of these times,
Where none will sweat but for promotion.
– WILLIAM SHAKESPEARE, As You Like It, Akt II, Szene iii

Ich hätte gerne ein gutes Buch hervorgebracht.
Es ist nicht so ausgefallen; aber die Zeit ist vorbei,
in der es von mir verbessert werden könnte.
– LUDWIG WITTGENSTEIN

Kapitel 1

Einleitung

1.1 Motivation

Far from being new topics, knowledge and intangibles have been important throughout history. The difference is that, today, a firm's intangible assets are often the key element in its competitiveness. Increasingly, the capacity to combine external and internal sources of knowledge to exploit commercial opportunities has become a distinctive competency. Firms possess many different types of knowledge, which may be codified or tacit – codified knowledge can be bought, sold, stocked and valued, tacit cannot.
– [EHB+00]

Die Industriegesellschaft verwandelt sich zunehmend in eine Wissensgesellschaft. Neben den klassischen drei Produktionsfaktoren Rohstoff, Arbeit und Kapital wird der Faktor Wissen immer wichtiger und wird zugleich zum wettbewerbsentscheidenden Faktor in wissensintensiven Unternehmen [KMP01]. Dieser Trend und die damit verbundenen Erwartungen werden besonders deutlich, wenn man den Marktwert (Börsenkapitalisierung) reiner Software-Unternehmen (als wissensintensive Unternehmen par excellence) im Vergleich zu klassischen Industrieunternehmen betrachtet. Hier tritt beispielsweise die neuartige Situation auf, daß der Wert eines Industrieunternehmens trotz Anlagevermögens, Umsatzes und Mitarbeiterzahl, die um ein Vielfaches höher sind, für geringer als der eines Software-Unternehmen erachtet wird. Neben einer anfänglich gewiß übertriebenen (und mittlerweile wieder teilweise zu stark geschwundenen) Begeisterung des Marktes für Werte der „neuen Ökonomie“ ist der Unterschied zwischen Marktwert und Buchwert (als Summe der materiellen Werte) in den immateriellen Werten des Unternehmens zu sehen. Dieses Wissenskapital, auch *intangible assets* oder *intellectual capital* genannt, bestimmt in der Wissensgesellschaft oder der neuen Wissensökonomie (*knowledge economy*) sowohl die Wettbewerbsfähigkeit eines Unternehmens als auch dessen Wert für die Besitzer (*shareholder value*) neben den klassischen finanziellen Indikatoren [Sve98].

Eine Reihe von Basisinnovationen, die jeweils industrielle Revolutionen auslösten (Kondratieff-Zyklen), hat die Gesellschaft dorthin geführt [Nor98]: Die Dampfmaschine löste um 1800 die erste industrielle Revolution aus, die Eisenbahn und die Stahlindustrie 1850 die zweite, Elektrotechnik und Chemie um die Jahrhundertwende die dritte, und nach dem 2. Weltkrieg führten die Petrochemie, die Automobilindustrie, die Kernenergie, Fernsehen und Telefon zur vierten. Die 5. industrielle Revolution durch die Dienstleistungsgesellschaft, Informationstechnik (Computer, Multimedia) und Bio- und Gentechnik ist bereits im Gange, und die Grenzen der traditionellen Industrie zeigen sich bereits heutzutage: Rohstoffe und Kapital sind begrenzt, die Rationalisierungspotentiale der Arbeitskraft sind nicht mehr signifikant zu erhöhen, und der Konsum läßt

sich nicht mehr grenzenlos steigern. Die bekannte Studie „Grenzen des Wachstums“ des *Club of Rome* [MMZM73] hat dies bereits 1972 beschrieben. Zusammen mit der Entwicklung der o.g. Dienstleistungs- und Informationsgesellschaft, also einer Wissensgesellschaft, läßt sich konstatieren, daß der wichtigste Wachstumsfaktor damit das Wissen geworden ist. Dies bedeutet, daß das Wissen nun auch seinerseits neben den klassischen Produkten der Industriegesellschaft gesammelt, gekauft, geliehen und bewertet wird: Information wird zum Wirtschaftsgut.

Es gibt weitere Gründe dafür, daß Wissen ein wichtigerer Faktor in der Wirtschaft wird. Sie resultieren auch aus der eben beschriebenen Entwicklung und sind unter anderem darin zu suchen, daß

- Geschwindigkeit, Wachstum und Vernetzung in Unternehmen immer höher werden,
- die Verteilung von Mitarbeitern, Kunden und Partnern mittlerweile global geschieht,
- die Wissensbestände schneller als je zuvor wachsen und die Langlebigkeit gerade technischen Wissens immer mehr abnimmt (sinkende Halbwertszeit des Wissens [Gro00b, S. 10]), die Ausdifferenzierung aber immer mehr zunimmt,
- zunehmend Kooperationen, Allianzen und Partnerschaften (z.B. durch Akquisitionen oder Verschmelzungen von Firmen und ganzen Konzernen) gebildet werden,
- Organisationsstrukturen schlanker und immer schnellebiger werden (unter dem Einfluß des *Lean Management*) und früher funktionale Abteilungsgliederungen mittlerweile häufig durch prozess- bzw. kundenorientierte Strukturen ersetzt werden,
- Experten mit ähnlichen Wissensgebieten oft unvernetzt über das ganze Unternehmen bzw. die ganze Welt verteilt sind.

Dies alles wirft meist gravierende Probleme auf, denn Wissen ist im Vergleich mit den traditionellen Ressourcen nur äußerst schwer zu handhaben. Das Wissensmanagement tritt daher als interdisziplinäre Fachrichtung auf, die Aspekte aus Informationstechnik, Wirtschaftswissenschaften und Arbeits- und Sozialwissenschaften verbindet, um die auftretenden Probleme zu untersuchen und Lösungen dafür anzubieten.

Eine weitere Entwicklung, die im Rahmen der Wandlung zur Wissensgesellschaft geschieht – und bestimmt Ausdruck des Wandels ist –, ist die des Internet und der damit verbundenen Technik (Protokolle, Formate) [MNSS99]. Insbesondere kommt hier das *World-Wide-Web* (WWW) mit seinem Protokoll HTTP und dem Medium des Hypertextsystems HTML zum Einsatz; erst diese beiden Entwicklungen haben das Internet so erfolgreich gemacht.[1]

Seit dem ersten Auftreten von Web-Servern und HTML um 1992 hat in diesem Bereich eine dramatische Entwicklung eingesetzt. Zum einen gab es eine gewaltige *quantitative* Entwicklung, die sich in anfangs exponentiellem Wachstum der Anzahl der dem Internet angeschlossenen Rechner, der betriebenen Web-Server und der angebotenen Seiten niederschlug, zum anderen ist auch eine – immer noch anhaltende – *qualitative* Entwicklung im Gange, die die eingesetzten Techniken betrifft: Waren anfangs sehr einfache Web-Server im Einsatz, die lediglich statische HTML-Seiten

[1] Oftmals wird auch – gerade in populärwissenschaftlichen Publikationen und der Tagespresse – nicht oder kaum zwischen den Begriffen *Internet* und *World-Wide-Web* unterschieden. Als Internet wird der weltweite Rechner- und Netzverbund bezeichnet, der auf einer Sammlung von gemeinsamen Protokollen (z.B. TCP/IP) und Standards aufsetzt. Darauf aufbauend sind anwendungsspezifische Protokolle wie POP, SMTP (für e-Mail), HTTP (für das WWW [FGM+99]), FTP (für Dateiübertragung) und viele mehr definiert. Zusätzlich kommen noch viele proprietäre Protokolle zum Einsatz, die nicht genormt sind und Spezialanwendungen einzelner Hersteller oder Benutzer dienen. Das WWW wird durch die globale Hypertext-Referenzstruktur der HTML-Seiten, die von Web-Servern stammen, gebildet und ist somit ein mehr virtuelles Konstrukt.

auslieferten, wurde nach und nach die Mächtigkeit sowohl der Server als auch der Protokolle und Formate ständig erhöht. Derzeit entwickeln sich die Web-Browser zu einem universellen Endgerät (Client, *Frontend*) von Client/Server-Systemen. Viele kommerzielle Anwendungen wie Informationssysteme oder Buchungssysteme werden auf web-basierte Systeme umgestellt oder erhalten zusätzliche, öffentliche Zugänge über das Internet [Tur99]. Auch innerhalb der Unternehmen haben die Techniken des Internet, also Protokolle wie TCP/IP, SMTP, POP, FTP, HTTP etc. massiv an Bedeutung zu gewinnen begonnen und bilden so mittlerweile häufig die Struktur firmeneigener Intranets, die zumeist auch Anbindung über spezielle Sicherheitsknoten (*firewalls*) an das Internet haben. Hier entwickelt sich zudem ein ganz neues Paradigma, das auch die firmenübergreifende Kopplung von Informationssystemen gestattet [Att99].

Dadurch ist es wie nie zuvor möglich, gleichzeitig auf große Bestände an Wissen zuzugreifen und selber große Wissensbestände für andere verfügbar zu machen. Die damit entstehende Problematik, in der unstrukturierten Fülle von Informationen die Orientierung zu behalten, hat – nach einer Reihe von evolutionären Schritten – zur Entwicklung von *Portalen* im Internet und im Intranet geführt.

Parallel zu der Entwicklung des Internet und des WWW und bereits geraume Zeit davor haben die wissensintensiven Unternehmen begonnen, Wissensmanagement-Lösungen zu bauen, die vielfach auch als *Organizational Memory Information System* (OMIS) bezeichnet werden [ADK98]. Der Begriff des Wissensmanagement-Systems (WMS) wird von vielen Software-Herstellern inzwischen recht weit gefaßt und gerade in jüngster Zeit werden aufgrund der eingangs beschriebenen Entwicklung der Bedeutung des Wissens viele längst bestehende Anwendungen nachträglich mit diesem Attribut versehen [IW01c]. Eine große Anzahl von Systemen läßt sich heute zu den Wissensmanagement-Systemen zählen, darunter fallen – je nach Definition der Anbieter – auch Dokumentenmanagement-Systeme (DMS) sowie häufig auch *Groupware*-Systeme, Kommunikations-Systeme und Workflow-Systeme. Auf der anderen Seite positionieren sich DMS-Anbieter zusätzlich als CMS-Anbieter [IW00a, IW00b], indem sie die benötigten Funktionen ausbauen oder entsprechend ergänzen.

Allen Entwicklungen sowohl im DMS- und CMS-Markt als auch im WMS-Markt gemeinsam ist das verstärkte Engagement in Techniken, die auch für Internet-basierte Systeme wie Portale verwendet werden [Zöl00]. So gibt es inzwischen eine ganze Reihe von Client/Server-basierten Lösungen, die sich der Technik des WWW wie HTTP, Web-Server und Web-Browsern bedienen. Das bedeutet für die Benutzer der Systeme in den Unternehmen, daß sie einfach per Web-Browser auf die Systeme zugreifen können und die aufwendige und wartungsintensive Installation von spezieller Zugangssoftware nicht mehr nötig ist. Diese Entwicklung kommt somit dem Ruf nach Senkung der Kosten (*Total cost of ownership*) für die Arbeitsplatzausrüstung in besonderem Maße nach. Weiterhin erlaubt dies den uniformen Zugriff sowohl der Mitarbeiter im Haus und außer Haus als auch von Kunden und Partnern auf entsprechend freigegebene Informationen, und zusätzlich läßt sich auf diesem Wege eine große Zahl von separaten, evtl. auch externen Quellen (e-Mail, Nachrichten) nahtlos integrieren.

Die Zusammenfassung der bisherigen teilweise auch von verschiedenen Systemen bereitgestellten Dienste und Informationen unter einer gemeinsamen, im Internet und Firmen-Intranet erreichbaren Instanz, einem Portal, bildet den derzeitigen Stand der Entwicklung. Die Begriffe, die für solche System geprägt wurden, sind die der Unternehmensportale (*Enterprise Portals*, *Enterprise Information Portals*) oder Wissensportale (*Enterprise Knowledge Portals*).

An dieser Stelle treffen die bisher eher getrennt verlaufenen Entwicklungen der Dokumenten- und Wissensmanagement-Systeme mit denen der WWW-basierten Unternehmensportale und ihrer Abkömmlinge zusammen und bilden die neue Klasse der Wissensmanagement-Portale, oder auch der Portale für das Wissensmanagement, die das Thema dieser Arbeit sind.

1.2 Ziele der Arbeit

Gegen Zielsetzungen ist nichts einzuwenden, sofern man sich dadurch nicht von interessanten Umwegen abhalten läßt.
– MARK TWAIN

Portale entwickeln sich zunehmend zu Werkzeugen für das Wissensmanagement in Unternehmen [JBÖ00]. Aufgrund der historischen Entwicklung der Technologien des Dokumentenmanagements und des darauf aufbauenden Wissens- und Content-Managements ist eine enge Verzahnung sowohl auf technischer als auch auf konzeptueller Ebene festzustellen. Allerdings werden insbesondere im kommerziellen Umfeld die Begriffe uneinheitlich gebraucht. Die Grenzziehung geschieht meist willkürlich und ist oft durch den Wunsch geprägt, bestehende Produkte neu zu positionieren und an dem sich rasch entwickelnden Markt für Wissensmanagement zu partizipieren. Auf der anderen Seite ist die Durchdringung des Themas seitens der Forschung und Wissenschaft insbesondere auf der technischen Ebene (aus der informationstechnischen und besonders der softwaresystemtechnischen Perspektive), also der der systematischen Beschreibung von fachlichen Anforderungen, der Entwicklung von Konzepten und Modellen sowie der Aufstellung, Beschreibung und Umsetzung von Systemarchitekturen und -entwürfen, recht gering [Gen99, BVÖ99, WS98]. Hingegen ist die Abdeckung des Themas Wissensmanagement von der betriebswirtschaftlichen (auch der wirtschaftsinformatischen), sozialen und arbeitswissenschaftlichen Seite her bereits größer; eine Fülle von Publikationen belegt die Wichtigkeit des Themas [NT95, PRR99, DP98, Weg99, Sch96, Ame00, Nor98, Dru98].

In dieser interessanten Situation fokussiert sich diese Arbeit auf informationstechnische Fragestellungen. Die beiden wichtigsten Aspekte sind:

1. Die **konzeptuelle Fundierung** des Themas Portale für das Wissensmanagement soll konsolidiert werden. Dazu werden betriebswirtschaftliche, soziale und arbeitswissenschaftliche Erkenntnisse herangezogen, um zunächst rein fachliche Anforderungen allgemeiner Art an die benötigten Werkzeuge abzuleiten. Hierbei müssen ebenso organisationelle wie kulturelle Phänomene berücksichtigt werden. Die Untersuchung der Evolution von DMS-, CMS- und WMS-Systemen und OMIS liefert dabei wertvolle Erkenntnisse zur Eingrenzung der Begriffe. Auf dieser Basis kann dann ein breiter Katalog an Funktionalitäten erarbeitet werden, der Werkzeuge, Metaphern und Verfahrensweisen zur Unterstützung von Wissensmanagement durch Internet-Informationssysteme bietet. Dazu werden sowohl bestehende Lösungen verschiedener Software-Anbieter analysiert und klassifiziert als auch die neusten Entwicklungen untersucht.

2. **Architektur und Entwurf** eines auf Grundlage der gefundenen Anforderungen entwickelten Systems sollen dargestellt und begründet werden. Neben der breiten Abdeckung sowohl der fachlichen Anforderungen als auch der Systemanforderungen stehen dabei besonders die Ziele der Verwendung von modernen Technologien für Inter-, Extra- und Intranet im Vordergrund. Die Beiträge, die das Gebiet der kooperativen Informationssysteme [DDJ+97] für das Verständnis und den Bau von offenen, integrativen und verteilten Lösungen geleistet hat, sollen hier einfließen. Zwar ist Portalsoftware per se als klassische Client/Server-Architektur zu sehen, tatsächlich kann man sie aber auch als offene und verteilte Systeme sehen, sobald nämlich Aspekte wie Skalierung an Anbindung von weiteren Systemen und Komponenten relevant werden. Die Frage nach geeigneten objektorientierten Entwürfen und Entwurfsmustern [GHJV95] für diese Szenarien steht dabei im Vordergrund. Neben moderner Technik sollen zudem auch moderne Notationen wie die *Unified Modeling Language* (UML) für den objektorientierten Entwurf zum Einsatz kommen [BRJ99, RJB99, JBR99].

Um diese Aspekte auszuschärfen und die Fragestellungen, die in der Arbeit untersucht werden sollen, weiter zu präzisieren, werden nun eine Reihe von gebündelten Leitfragen formuliert, die den Fortgang und die Struktur der Untersuchungen bestimmen sollen. Die Leitfragen werden getrennt für die beiden Aspekte aufgestellt.

Für den ersten Aspekt:

- Wie läßt sich der Begriff Wissen näher beschreiben und welche Ziele, Aufgaben und Prozesse sind im Wissensmanagement wichtig? Welche davon lassen sich durch Informationssysteme gut unterstützen?
- Welche Arten von Systemen sind anzutreffen und wie lassen sie sich grundsätzlich klassifizieren? Was leisten Informationssysteme im Bereich des Wissensmanagements bisher?
- Wie lassen sich die eng miteinander verwandten Dokumenten-, Content- und Wissensmanagement-Systeme sowie Portale generell anhand der Konzepte und Funktionalitäten klassifizieren und was leisten Systeme, die bereits im Kontext der Internet-basierten Portalsoftware für das Wissensmanagement anzusiedeln sind?
- Welche Modellierungskonzepte werden häufig verwendet und welche Konzepte bedürfen weiterer Untersuchung bzw. wurden noch nicht hinreichend beachtet oder fehlen?
- Welche fachlichen Anforderungen an die Funktionalität und welche Systemanforderungen lassen sich für Portalsoftware für Wissensmanagement-Systeme stellen?

Für den zweiten Aspekt:

- Wie läßt sich ein problemadäquates Dokumentenmodell ausformulieren, das auf den Einsichten des ersten Aspekts beruht? Welche Bezüge zu bestehenden Modellen lassen sich dabei herstellen?
- Wie läßt sich ein Prozeßmodell für ein Wissensportal unter besonderer Berücksichtigung bestehender Ansätze zur Modellierung kooperativer Informationssysteme formulieren?
- Welche Entwurfsentscheidungen sind zu treffen und wie werden sie begründet? Welche Aufgaben des Systems sollten dabei durch bestehende Produkte (Komponenten) erfüllt werden und welche Funktionalität (Komponenten) sollten oder müssen neu entworfen und implementiert werden (*make vs. buy*)?
- Welche System-Architekturen werden von Portalsystemen für das Wissensmanagement häufig eingesetzt und wie sieht eine gute Architektur für ein Portalsystem unter Berücksichtigung des Wissensstandes über Client/Server-Systeme und Mehrschichtenarchitekturen aus? Wie erhält man größtmögliche Freiheit für Skalierbarkeit, neue Anforderungen, Wartbarkeit und Erweiterbarkeit?
- Wie sieht der softwaretechnische Entwurf aller Schichten und Komponenten auf Grundlage der Antworten auf die obigen Fragen aus? Welche Abstraktionen und Entwurfsmuster sind zu verwenden?

Insgesamt gesehen soll der Fokus der Arbeit auf dem Bereich der Informationssysteme liegen, hier speziell auf Konzepten, Systemen und Architekturen für Portalsoftware, und weniger auf den – für Wissensmanagement im allgemeinen ebenso wichtigen – Bereichen der betriebswirtschaftlichen Organisation und arbeitswissenschaftlichen Fragestellungen.

Abschließend soll kurz die Frage diskutiert werden, wie diese Arbeit im Spannungsfeld zwischen klassischer Informatik als wissenschaftlicher (naturwissenschaftlicher?) Disziplin und Ingenieursdisziplin [End99] positioniert werden soll. Im ersten Fall stünde nach [BS99] primär die wissenschaftliche Erkenntnis im Zusammenhang mit Fragen der maschinellen Informationsverarbeitung im Vordergrund, im zweiten Fall die Lösung konkreter Aufgaben der Informationsverarbeitung unter kommerziellen und technischen Randbedingungen. Die theoretischen (wirtschafts-, arbeits- und sozialwissenschaftlichen) Grundlagen dieser Arbeit aus dem Wissensmanagement werden nicht wesentlich weiter entwickelt. Kern aller (tatsächlich pragmatischen) Fragestellungen ist die Entwicklung konkreter Konzepte und Systemarchitekturen für den gewählten Problemkreis; kommerzielle (allerdings grundsätzliche) Überlegungen bestimmen den *thematischen* Rahmen dabei stark, während Überlegungen zum Vorgehen, der Klassifizierung und Modellierung den ständigen *methodischen* Rahmen bilden. So liegt der Schwerpunkt im Gebiet der Werkzeuge und Methoden der Softwaretechnik (bzw. *Systems & Software Engineering*) und damit sicherlich in dem Bereich, in dem sich Informatik als Grundlagenwissenschaft und als Ingenieursdisziplin weit überschneiden. Festzuhalten bleibt also, daß diese Arbeit als wissenschaftliche Arbeit mit ausgeprägtem ingenieurswissenschaftlichen Charakter gesehen wird, wobei aber manche Überlegungen bereits ganz stark in den Bereich der Wirtschaftsinformatik fallen. Wie dicht der Schwerpunkt an der jeweiligen Seite gesehen wird, hängt nicht zuletzt vom Standpunkt des Betrachters bezüglich seiner Einordnung der Informatik ab; versucht wurde aber, den Schwerpunkt möglichst in der Mitte zu halten.

1.3 Vorgehen und Gliederung

Ja, mach nur einen Plan
sei nur ein großes Licht
und mach dann noch 'nen zweiten Plan
gehn tun sie beide nicht.
– BERTOLT BRECHT, Dreigroschenoper

In Kapitel 2 werden nach einem Exkurs zur Begriffsbestimmung für das Wissensmanagement und einer Einordnung der Begriffe Wissen, Information und Daten die Grundlagen und Aufgaben des Wissensmanagements besprochen. Dazu werden verschiedene Dimensionen des Wissens erläutert und einige Sichten der Teilaufgaben und -prozesse des Wissensmanagements im sogenannten Wissenszyklus dargestellt und miteinander verglichen. Der Fokus liegt dabei auf der Darstellung der jeweiligen Ziele, Aufgaben und Probleme sowie auf einer Skizzierung der Beiträge, die die Informations- und Kommunikationstechnik[2] bisher zur Bewältigung dieser Aufgaben zu leisten vermag.

Im darauf folgenden Kapitel 3 wird in Form einer Bestandsaufnahme diskutiert, welche verschiedenen Arten von Informationssystemen für das Wissensmanagement überhaupt Verwendung finden und wie diese Systeme zu klassifizieren sind. Es findet dabei insbesondere eine Untersuchung der Evolution der besonders interessanten Klassen der Dokumentenmanagement-Systeme (DMS), Content-Management-Systeme (CMS) und Wissensmanagement-Systeme (WMS) sowie von Portal-Software (Unternehmensportale, EIP) statt. Der Schwerpunkt liegt dabei auf einer deutlichen Klassifikation und Abgrenzung der verwendeten Konzepte, Begriffe und Funktionalitäten der untersuchten Systeme. Ergänzt und illustriert wird dieser Abschnitt durch die Analyse einiger ausgewählter kommerzieller Systeme unter den genannten Gesichtspunkten. Hier

[2] Im folgenden wird der besseren Lesbarkeit wegen abkürzenderweise nur von *Informationstechnik* statt *Informations- und Kommunikationstechnik* gesprochen.

stehen in erster Linie Funktionen und Konzepte und in zweiter Linie Software-Architekturen im Mittelpunkt des Interesses.

In Kapitel 4 wird auf Grundlage der bisherigen informationstechnischen, betriebswirtschaftlichen, sozialen und arbeitswissenschaftlichen Erkenntnisse sowie der erfolgten Klassifikation von Systemen und Funktionen ein allgemeiner Anforderungskatalog an Kernfunktionalität erarbeitet, der Werkzeuge, Metaphern und Verfahrensweisen zur Unterstützung von Wissensmanagement durch ein Portal fordert. Dieser Katalog deckt primär fachliche Anforderungen ab, enthält jedoch auch Anforderungen an die Benutzungsoberfläche sowie wichtige Systemanforderungen.

In Kapitel 5 werden zunächst die Modelle und Konzepte entwickelt, die als Grundlage für den Entwurf dienen. Dabei werden erstens die aus den Untersuchungen konkreter Systemklassen und bestehender Systeme gewonnenen Einsichten über die Abstraktionsmechanismen von Informationsartefakten aufgegriffen und zu einem neuen, für diese Problemdomäne adäquaten Konzept geführt. Zweitens werden die Ergebnisse der Forschungen über Prozess- und Dokumentenmodelle in kooperativen Informationssystemen dargestellt und die nutzbringenden Aspekte auf den Bereich der Internet-Informationssysteme, hier speziell von Portalsystemen, übertragen.

Das Kapitel 6 ist der Darstellung des softwaretechnischen Entwurfs eines auf den beschriebenen Anforderungen und Modellen basierenden Architektur für ein Wissensportal gewidmet. Es werden die Entwurfsentscheidungen, die generelle Systemarchitektur, der Strukturentwurf und der Verhaltensentwurf des resultierenden Systems in technisch präziser Form beschrieben. Der Aufbau und die Funktionsweise der einzelnen Schichten und Komponenten werden zusammen mit den entwickelten Schnittstellen durch die Verwendung der *Unified Modeling Language* (UML) dokumentiert und die einzelnen Entwurfsentscheidungen diskutiert und begründet.

Anschließend wird in Kapitel 7 anhand mehrerer Einsatzbeispiele die Verwendung, Anpaßbarkeit und die Leistungsfähigkeit des entstandenes Systems gezeigt. Eine Zusammenfassung des Erreichten und ein Ausblick auf neu entstandene Fragestellungen bildet den Schluß in Kapitel 8.

Kapitel 2

Ziele, Aufgaben und Prozesse des Wissensmanagements

There is much pleasure to be gained from useless knowledge.
– BERTRAND RUSSELL

Zu Beginn dieser Arbeit wird der fachliche Hintergrund des Themas, nämlich das Wissensmanagement in Form einer Bestandsaufnahme untersucht. Hauptaufgabe dieses Kapitels ist es, näher zu klassifizieren und zu strukturieren, welche Ziele, Aufgaben und Prozesse das Wissensmanagement besitzt. Dazu werden im wesentlichen anerkannte Quellen für diese Thematik herangezogen und analysiert. Ziel dieses Kapitels ist es ausdrücklich nicht, eine neue Sichtweise des Wissensmanagements einzuführen, neue Aspekte zu identifizieren oder neue Vorgehensweisen zu entwickeln. Vielmehr wird die vorgenommene Analyse der Ziele und Aufgaben die Untersuchungen des folgenden Kapitels über Informationssystemunterstützung zum Wissensmanagement vorbereiten, indem eine strukturiertes Basis von Aufgaben beschrieben wird, anhand derer die zu untersuchenden Systemklassen und Systeme eingeordnet und bewertet werden können.

Zur Analyse wird zunächst durch Untersuchung einiger Definitionen festgestellt, worum es bei dem Thema Wissensmanagement überhaupt geht, und es werden daraus Fragestellungen für den Rest des Kapitels entwickelt. Als erste ergibt sich die Frage nach einer Präzisierung der drei ständig vorkommenden Begriffe Daten, Informationen und Wissen. Da es sich dabei um diejenigen Artefakte handelt, mit denen sich das Wissensmanagement inhärent befaßt, sind diese für den Geltungsbereich dieser Arbeit zunächst zu klären, um so eine einheitlich verwendbare Terminologie zu schaffen. Diese Diskussion ist auch deswegen nötig, weil die einzelnen Fachrichtungen (Informatik, Betriebswirtschaftslehre, Soziologie, Arbeitswissenschaft etc.) durchaus unterschiedlich an diese Frage herangehen. Insbesondere der Begriff des Wissens weist zahlreiche Aspekte auf, die als Dimensionen des Wissens beschrieben werden. Diese unterschiedlichen Arten haben große Auswirkungen für das Verständnis der von Informationssystemen zu unterstützenden Aufgaben und Prozesse und werden deshalb zur Vorbereitung der Untersuchungen von Systemen im nächsten Kapitel ausführlich diskutiert. Eine Besprechung der einzelnen Teilaufgaben – nach der dieser Einteilung zugrundeliegenden Sicht aus [PRR99] Bausteine genannt – des Wissensmanagements bildet den Kern dieses Kapitels. Anhand dieses Klassifikationsrasters lassen sich im folgenden Kapitel die einzelnen Beiträge von Informationssystemen zum Wissensmanagement erst einordnen. Es werden dabei bereits in diesem Kapitel erste Hinweise auf die jeweils für die einzelnen Bausteine verwendbaren Informationssysteme gegeben. Schließlich werden die Bausteine nach [PRR99] mit einigen anderen Schemata für Teilaufgaben verglichen,

um die Einteilung endgültig zu konsolidieren und diejenigen Aufgaben zu identifizieren, die überhaupt Relevanz für die folgende Untersuchung der Informationssysteme haben.

2.1 Definition von Wissensmanagement

Das Wissensmanagement ist eine interdisziplinäre Fachrichtung, die Aspekte aus Informationstechnik, Wirtschaftswissenschaften und Arbeits- und Sozialwissenschaften verbindet. Es gibt zahlreiche Definitionen für Wissensmanagement; einige ausgewählte sollen im folgenden wiedergegeben werden.

Eine OVUM-Studie [WS98] definiert *Knowledge Management* wie folgt:

> „[Knowledge management is] the task of developing and exploiting an organization's tangible and intangible knowledge resources. Knowledge management covers organizational and technological issues."

[IW99c] kommt zu der Definition:

> „Zum Wissens-Management zählt ein bestimmter Bereich von technischen und geschäftlichen Methoden, klassischerweise basierend auf der Überzeugung, daß Einzelpersonen 99 Prozent aller wertvollen Informationen in einem Unternehmen besitzen. Könnten diese Informationen erfaßt und verteilt werden, wäre eine effizientere Unterstützung der sogenannten Informationsökonomie möglich: Alle Firmen erzeugen im Prinzip Informationen und stehen auf der Informationsebene miteinander im Wettbewerb. Produkte sind nichts anderes als das Resultat dieser Informationen."

In [Sch96] findet man:

> „Das Wissensmanagement umfasst alle human- und technikorientierten Interventionen und Maßnahmenpakete, die dazu geeignet sind, die Wissensproduktion, Wissensreproduktion, Wissensdistribution, Wissensverwertung und Wissenslogistik in einer Organisation zu optimieren."

In [Hei00] wird definiert:

> „Das richtige Wissen zur richtigen Zeit am richtigen Ort verfügbar zu haben, um Fehler und Doppelarbeit zu vermeiden."

Diese oder eine ähnliche Formulierung liest man übrigens in Fachzeitschriften und weiteren Publikationen häufiger [Noh01]. Schließlich wird in [KLT00] definiert:

> „Aufgabe des Wissensmanagements ist es, den Mitarbeitern im Unternehmen entscheidungs- und handlungsrelevante Informationen bereitzustellen und die Mitarbeiter bei der intelligenten Verarbeitung dieser Informationen zu unterstützen."

Alle diese Definitionen haben viele Gemeinsamkeiten. Es soll hier an dieser Stelle nicht eine weitere Definition der obigen Art getroffen werden, vielmehr soll versucht werden, die zentralen Aspekte zu erfassen und zu hinterfragen, um daraus das weitere Vorgehen abzuleiten.

1. Alle Definitionen von Wissensmanagement sprechen von *Informationen* als den zu handhabenden Fakten, und nicht von *Wissen*. Hier muß also noch eine genauere Unterscheidung und Begriffsbestimmung getroffen werden.

2. Es gibt unterschiedliche Arten von Wissen bzw. Informationen, die es zu verwalten gilt.

3. Wissensmanagement findet in und für Organisationen statt (z.B. Unternehmen), also innerhalb von Gruppen von zusammenarbeitenden Menschen.

4. Es werden viele unterschiedliche Prozesse, um Wissen handzuhaben, angesprochen.

5. Die Mitarbeiter der Unternehmen stehen im Vordergrund des Interesses, von Informationssystemen ist erst in zweiter Linie die Rede.

Aus diesen Beobachtungen ergibt sich die folgende Untergliederung von näher zu diskutierenden Punkten: In den nachfolgenden Abschnitten sollen zunächst die Begriffe Daten, Information und Wissen definiert werden (Abschnitt 2.2), verschiedenen Dimensionen des Wissens dargestellt werden (Abschnitt 2.3), um dann einzelne Elemente des Wissensmanagements (Teil-Prozesse) zu benennen und zu erläutern (Abschnitt 2.4). Diese Ausführungen orientieren sich stark an dem etablierten Modell von [PRR99]. Abschließend wird die dargelegte Sichtweise auf die benötigten Bausteine mit mehreren anderen Klassifikationen in Beziehung gesetzt und zusammenfassend ein Kernbereich identifiziert, in dem die Unterstützung der Prozesse durch die Informationstechnik besonders gut möglich ist (Abschnitt 2.5). Eine Zusammenfassung (Abschnitt 2.6) beendet dieses Thema.

Von der an sich in diesem Zusammenhang folgerichtigen näheren Eingrenzung des Begriffes *Management* wollen wir absehen. Hier soll die [Gaß99] entnommene und dort eben diskutierte Definition von Management als „Gestaltung und Lenkung (einschließlich Entwicklung) eines zweckorientierten Systems“ genügen.

2.2 Abgrenzung von Daten, Information und Wissen

Da Wissensmanagement offensichtlich mit Wissen operiert, ist eine Abgrenzung der drei Begriffe *Daten*, *Informationen* und *Wissen* nötig. Dabei wird eine informatische Perspektive eingenommen, die die Begriffe aufeinander aufbauend als Interpretations- oder Abstraktions- bzw. Repräsentationsprozeß definiert (vgl. Abbildung 2.1).

Information stellt neben Energie und Materie einen der wichtigsten Grundbegriffe der Natur- und Ingenieurwissenschaften dar. Überraschenderweise gibt es gerade in der Informatik bisher wenige präzise Definitionen; in der Regel wird unter Bezugnahme auf SHANNONS Informationstheorie [Sha48] versucht, den Begriff einzugrenzen. Dabei stehen dann nachrichtentechnische Aspekte im Vordergrund und die *Nachricht* wird als Träger der Information eingeführt [Dud88] und sogar mit dem Begriff der *Daten* identifiziert [DIN78].

2.2.1 Daten

Wir sehen die Daten als physikalische Repräsentation (durch analoge oder digitale Signale wie Spannungen, elektrische Ladungen etc., die in Rechnersystemen häufig verwendet werden) von Informationen an. Die Daten werden durch eine geeignete Codierung auf Datenträgern gespeichert bzw. durch verschiedene Medien und Träger übertragen [Lag87]. Hier werden oft analoge

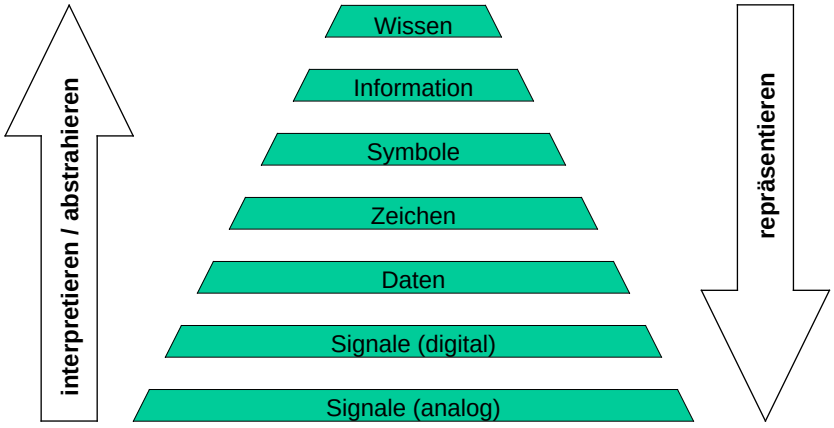

Abbildung 2.1: Daten, Information und Wissen

Codierungen benutzt. Nichttechnische Codierungen wie beispielsweise Kerben in Tontafeln oder Tinte auf Papier fallen streng betrachtet auch in diese Klasse, ebenso wie die Übertragung durch periodische Druckänderungen in der Luft (Sprache). In digitalen Rechnersystemen werden auf der Ebene der Rechnerstrukturen (Prozessor, Bus, Speicher etc.) Daten ausschließlich binär codiert, wobei noch verschiedene Interpretationen bezüglich der Wortlängen eine Rolle spielen. Zur Datendarstellung werden normalerweise verschiedene Alphabete verwendet, die jeweils geordnete Mengen von zu verwendenden Zeichen enthalten. Typische Beispiele von verwendeten Alphabeten sind:

- Die Menge der durch Zweierkomplement mit 32 Bit darstellbaren natürlichen Zahlen.
- Die Menge der Buchstaben und Ziffern nebst Satzzeichen und Sonderzeichen, die durch 8 Bit dargestellt wird wie es die Zeichenkodierung `ISO 8551 Latin 1` definiert.
- Die Wahrheitswerte `true` und `false`, die durch jeweils ein Bit dargestellt werden, das entweder gesetzt oder nicht gesetzt ist.

Die rechnerinterne Darstellung der üblichen Alphabete variiert sehr stark in Abhängigkeit von der verwendeten Rechnerarchitektur, der Betriebssystemsoftware, den Programmiersprachen und den Anwendungsprogrammen. Häufig sind sogar alle Freiheitsgrade miteinander kombinierbar.

> Beispiel: Die Folge von Bits `00001110` kann als Buchstabe einer speziellen Zeichencodierung betrachtet werden oder aber als die Ganzzahl `14`.

Auf einer etwas höheren Ebene kann von diesen technischen Darstellungsformen abstrahiert werden, so daß Daten durch elementare *Datentypen* beschrieben werden können wie *Ganzzahl*, *Zeichenkette* oder *Wahrheitswert*. Auf dieser Ebene wollen wir hier Daten definieren. Die Zeichen der verwendeten Alphabete werden *Symbole* genannt. Die erlaubten Symbole und ihre Anordnung (Syntax) bilden eine Sprache und werden durch Grammatiken formal beschrieben.

Im Bereich der Datenverwaltung (Netzwerke, Hardware, Rechnerarchitektur) interessieren in Bezug auf die Daten Fragen der Geschwindigkeit des Zugriffs, der Menge der Daten und der Sicherheit, etwa gegen Verfälschung und Verlust.

2.2.2 Information

Bisher wurde von jeder inhaltlichen Interpretation der Daten abgesehen. Eine Folge von Zeichen (Zeichenkette) verrät isoliert betrachtet noch nicht ihren Zweck oder die Tatsache, die durch sie beschrieben wird. Erst diese zusätzliche Kenntnis, die Semantik der Daten, erlaubt die Interpretation der Daten. Die Interpretation geschieht dabei durch einen Menschen oder zumindest durch ein Programm (*Software*) nach Regeln, die von Menschen erarbeitet wurden.

> Beispiel: Die Ganzzahl `14` ist an sich als Datum solange bedeutungslos, wie die Semantik nicht erklärt wurde. Ist hingegen bekannt, daß diese Zahl das aktuelle Datum (Tag des Monats) bezeichnet, so läßt sich die Zahl interpretieren.

Die Interpretation erfolgt durch einen der folgenden Prozesse (nach [DP98]):

- **Kontextualisierung:** Zweck und Zusammenhänge der Information erkennen
- **Kategorisierung:** Einordnen gegenüber anderen Informationen
- **Kalkulation:** Analysieren und Berechnen von neuen Informationen
- **Korrektur:** Korrigieren von Fehlern in den Daten
- **Kondensierung:** Zusammenfassen der Daten

Bereits die Interpretation kann nicht wirklich von Rechnern geleistet werden, sofern nicht wissensbasierte Systeme, die Konzepte, Regeln und Zusammenhänge explizit modellieren, verwendet werden. Systeme der künstlichen Intelligenz, die dazu in der Lage sind, und zu regelbasiertem Schließen etc. befähigt sind, sind nicht Untersuchungsgegenstand dieser Arbeit. Als Anwendungen solcher Systeme seien Textverstehen, automatisches Beweisen und Expertensysteme genannt. Die hier betrachtete Klasse von Informationssystemen kann lediglich Unterstützung für die Interpretation liefern. In Datenbanken und Informationssystemen geschieht dies üblicherweise durch die Bereitstellung von Datenwörterbüchern (*data dictionaries*) und in Programmiersprachen durch benutzerdefinierte Daten-Typen, Typkonstruktoren und in gewissem Maße durch Reflektion. Um eine klare Abgrenzung zum Begriff des Wissen zu erreichen, definieren wir daher für den Rahmen der Untersuchungen in dieser Arbeit Information als semantische Interpretation von Daten, die von Menschen durchgeführt wird.

Im folgenden wird der Informationsbegriff sehr häufig verwendet werden, um semantisch bedeutsame Einheiten zu charakterisieren, und er wird konsequenterweise auch dort verwendet, wo andere Autoren von Daten oder Wissen sprechen, aber Informationen gemäß der obigen Definition meinen. Dies spielt eine besonders große Rolle bei allen Modellen, die im weiteren Verlauf analysiert und entwickelt werden.

2.2.3 Wissen

> *Wissen kann man mitteilen, Weisheit aber nicht. Man kann sie finden, man kann sie leben, man kann von ihr getragen werden, man kann mit ihr Wunder tun, aber sagen und lehren kann man sie nicht.*
> – HERMANN HESSE, Siddhartha

Wissen geht deutlich über das bloße „Haben von Information“ hinaus; es erfordert beispielsweise Intuition, Verstehen, Erfahrung und ist die Grundlage für zielgerichtetes und kompetentes Handeln. In [DP98] wird etwa definiert:

> „Knowledge is a fluid mix of framed experience, values, contextual information, and expert insight that provides a framework for evaluating and incorporating new experiences and information. It originates and is applied in the minds of knowers. In organizations, it often becomes embedded not only in documents or repositories but also in organizational routines, processes, practices, and norms."

In [PRR99] wird Wissen folgendermaßen definiert:

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

Folgende Definition des Begriffs Wissen in Abgrenzung zu Daten findet sich ferner bei [Weg99]:

> „Wissen ist das, was Menschen die Fähigkeit gibt, bestimmte Aufgaben auszuführen, indem sie Daten aus verschiedenen externen Quellen kombinieren, die es ihnen ermöglichen, unter Verwendung eigener Informationen, Erfahrungen und Haltungen zu handeln. Dazu ist es notwendig, theoretisch oder praktisch mit der Domäne vertraut zu sein, in der die Aufgabe ihren Ursprung hat, oder vertraut mit den Vorgängen zu sein, die wichtig für die Ausführung dieser Aufgabe sind. Wissen anzuwenden, das auf diese Weise definiert ist, führt zu Ankündigungen, Voraussagen, kausalen Assoziationen oder Entscheidungen über das, was zu tun ist, und über das, wie es zu tun ist. Konsequenterweise sind Information und Wissen Konzepte, die nur auf persönlicher Ebene operationalisiert werden können."

Die Verwendung des Begriffs Daten widerspricht zwar unserer obigen Definition (stattdessen wäre Information einzusetzen), ansonsten charakterisiert diese Definition Wissen geeignet. Eine weitergehende Untersuchung über die definitorische Abgrenzung von Wissen mit zahlreichen unterschiedlichen Definitionen findet sich etwa in [Ame00, S. 39 ff.]. NONAKA und TAKEUCHI diskutieren die philosophische und geisteswissenschaftliche Entwicklung des Wissensbegriffs sowohl im westlichen als auch im östlichen Raum eingehend [NT95].

Eine ganz wesentliche Eigenschaft von Wissen ist, daß es ständig transformiert werden muß, um nutzbar gemacht werden zu können. Analog zu den Prozessen zur Informationsinterpretation können für Wissen die folgenden Prozesse zur Wissenstransformation identifiziert werden [DP98]:

- **Vergleich:** Wie verhält sich die Situation (die ja zunächst durch Informationen beschrieben ist) im Vergleich mit anderen bereits bekannten Situationen?
- **Konsequenzen:** Welche Folgen hat diese Information? Welche Entscheidungen und Maßnahmen müssen getroffen werden, um darauf angemessen zu reagieren?
- **Konversation:** Wie schätzen andere Personen die Information aufgrund ihres persönlichen Wissens ein?

Eine wesentliche Voraussetzung für die Transformationsprozesse ist das Vorhandensein von Erfahrungen, also einem Kontext, in dem die Informationen interpretiert und transformiert werden. Folgendes Beispiel aus [Lot01f] macht dies recht anschaulich deutlich:

„Context puts information into perspective. Or as the dictionary defines it, context is 'that which surrounds, and gives meaning to, something else.' For example, the concept of speed means something different depending on the context. The statement '10 miles an hour is fast' is meaningless by itself. But in the context of a person running, it is fast. For a bicycle, it is moderate. And for a car, it is painfully slow – unless you're driving in an ice storm at night with broken windshield wipers; then it might seem fast (more context)."

Abbildung 2.2: Wissenstreppe nach NORTH

Aus diesen Prozessen resultieren Können, Handeln und Kompetenz einzelner Personen oder Personengruppen und – in einem wirtschaftlichen Kontext betrachtet – letztendlich die Wettbewerbsfähigkeit einer Organisation. Abbildung 2.2 illustriert die Zusammenhänge zwischen Daten, Informationen und Wissen sowie den darauf aufbauenden Fähigkeiten anhand einer „Wissenstreppe" nach [Nor98]. Da der zentrale Begriff des Wissensmanagements eben der des Wissens ist, werden seine Dimensionen in dem folgenden Abschnitt weiter beleuchtet.

2.3 Dimensionen des Wissens

We know more than we can tell
and we can know nothing without relying upon those things
which we may not be able to tell.
– MICHAEL POLANYI

Wovon man nicht sprechen kann, darüber muß man schweigen.
– LUDWIG WITTGENSTEIN, Tractatus Logico-Philosophicus [Wit63]

Es gibt zahlreiche Klassifikationen von Wissen. Bereits in der Antike unterschied ARISTOTELES zwischen theoretischem und praktischem Wissen; eine Unterscheidung, die wir hier nicht weiter verfolgen wollen.

Eine der frühesten für das Wissensmanagement relevanten Klassifizierungen von Wissen geht auf POLANYI[1] zurück, der zwischen explizitem und implizitem (stillschweigendem) Wissen (*tacit knowledge*) unterschied [Pol58, Pol62, Sve97c]. [Weg99] betont außerdem die qualitativen Unterschiede zwischen explizitem Wissen (kodiertem Wissen) und implizitem Wissen:

Explizites Wissen

- besteht aus Informationen, die in Theorien, Formeln, Dokumentation, Modellen, Diagrammen usw. niedergelegt sind (Kenntnis),
- wird übertragen durch Unterweisung,
- wird Erworben durch Studieren und
- beinhaltet wenig Macht.

Implizites Wissen

- ist stillschweigendes Wissen (Alltagswissen),
- besteht aus Erfahrungen, Fertigkeiten und Einstellung (Können und Wollen),
- wird geteilt durch Demonstration,
- wird erworben durch Kopieren und Imitation im Sozialisierungsprozeß und
- kann als Macht angesehen werden.

Nach [Sve97c] basiert POLANYIS Wissenskonzept auf drei Annahmen: Wahre Erkenntnis wird nicht durch explizite Regeln oder Algorithmen gesteuert. Wissen ist öffentlich und zum größten Teil persönlich, da es von Menschen aufgebaut wird. Daher enthält es auch Emotionen. Das unter dem explizitem Wissen liegende Wissen ist grundsätzlicher und das gesamte Wissen ist deshalb entweder implizit oder basiert auf implizitem Wissen.

Das bedeutet, daß neues Wissen stets vor dem Hintergrund des bestehenden (impliziten) Wissens verarbeitet und interpretiert wird. Das meiste Wissen wird durch sprachliche Interaktion zwischen Menschen weitergegeben, da dies der effektivste Weg des Wissenstransfers ist. Die nur in der englischen Sprache) mögliche Unterscheidung zwischen *knowledge* als statischem Wissen und *knowing* als dynamischem Prozeß (des Anwendens, Verstehens und Lernens) macht die Dimensionen des Wissens deutlich. Wissen passiert demnach ständig. Wissen, auch implizites, kann durch Artikulation wieder explizit gemacht werden und gemäß der obigen Definitionen in Information verwandelt werden. Die Internalisierung und Externalisierung als Richtungen des Wissenstransfers sind daher von besonderem Interesse im Wissensmanagement.

Das explizite, externalisierte Wissen kann gut durch Informationen in Rechnersystemen repräsentiert werden. Als Informationsträger dafür werden Datenbanken, Dokumente (Textverarbeitung, Rechenblätter, Präsentationen), Zeichnungen, e-Mail-Notizen, Diskussionsforen etc. verwendet. Auf der nichttechnischen Ebene können z.B. gedruckte Informationen (Bücher, Manuskripte) benutzt werden.

Viele weitere Klassifikationen von Wissen sind möglich und wurden bereits vorgenommen, die hier nicht alle diskutiert werden können. Für den Bereich des Wissensmanagements, der hier von Interesse ist, sind folgende grundsätzliche Unterscheidungen relevant:

[1] Statt POLANYI trifft man öfters die (falsche) Schreibweise POLYANI an.

- Wissen kann *speziell* oder *allgemein* sein. Die Unterschiede sind graduell und die Skala reicht von allgemeinem Weltwissen (Alltagswissen) bis zu Expertenwissen in hochspezialisierten Domänen. Erstaunlicherweise ist gerade das Weltwissen besonders schwer zu formalisieren.
- Wissen kann *prozedural* oder *deklarativ* sind. Deklaratives Wissen wird durch Fakten und logische Regeln (Deduktionsregel, Inferenzregeln) beschrieben; prozedurales Wissen besteht aus Fertigkeiten und Können (Anweisungen und Ausdrücke eines Algorithmus, Rezepte, Anleitungen).
- Wissen kann *individuell* oder *kollektiv* sind, d.h. nur von Einzelpersonen getragen werden oder verteilt auf mehrere bis viele Personen sein.
- Wissen kann *explizit* oder *implizit* sein, wie oben bereits beschrieben.

In [NT95] werden zwei Dimensionen des Wissens beschrieben: neben der epistemologischen Dimension (explizites vs. implizites Wissen) existiert eine ontologische Dimension, die von individuellem Wissen über Gruppenwissen bis hin zu interorganisationalem Wissen reicht. Erweitert um die beiden oben genannten Dimensionen ergibt sich das in Abbildung 2.3 dargestellte Bild.

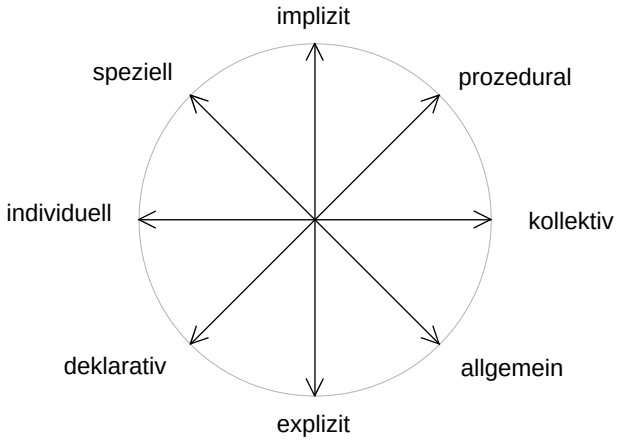

Abbildung 2.3: Dimensionen des Wissens

Ein weiterer Aspekt des Wissens sind die *Wissensträger*, d.h. diejenigen körperlichen Elemente, in denen sich Wissen manifestieren kann [Ame00]. Gemäß unserer obigen Definition muß statt von Wissensträgern dann auch von Informationsträgern gesprochen werden, wenn es sich nicht um Personen handelt. Es gibt neben personellen Wissensträgern eine Reihe von materiellen Informationsträgern wie Druckerzeugnisse, informationstechnische Artefakte (Datenbanken, Dokumente etc.) und industrielle Produkte, die sich zum Teil dazu zählen lassen können, wenn ihre Analyse das von den Herstellern hineingesteckte Wissen wieder rekonstruieren kann. Neben diesen isolierten Trägern lassen sich *kollektive* Wissensträger ausmachen, wobei wir als kollektives Wissen solches Wissen verstehen wollen, das über die bloße Aggregation oder Kumulation von individuellem Wissen hinausgeht, also Gesamtleistungen von Gruppen wie beispielsweise Orchestern, Sportmannschaften oder anderweitig „eingespielten Teams“ erklärt.

Die Summe der individuellen und kollektiven Wissensbestände wird als *organisationale Wissensbasis* bezeichnet [PRR99]. Sie subsumiert die Daten und Informationen, auf denen das jeweilige Wissen aufbaut. Die organisationale Wissensbasis unterliegt ständigen Änderungen durch eine Fülle von Ursachen: Produkte (eigene und fremde) werden verbessert, Prozesse, Aufgaben und

Verantwortlichkeiten werden verändert oder neu definiert, Organisationen werden (evtl. auch räumlich) umstrukturiert, Personen (Mitarbeiter, Kunden, Ansprechpartner) verlassen das Unternehmen, kommen neu hinzu oder bilden sich weiter, die Infrastruktur (Rechnerausstattung, Hard- und Software, Netzwerke, Kommunikationseinrichtungen, Bibliotheken) wird angepaßt, oder die rechtlichen und wirtschaftlichen Rahmenbedingungen ändern sich. Durch diese Veränderungen werden Lernprozesse induziert, die als *organisationales Lernen* bezeichnet werden. Die zielgerichtete Lenkung dieser Prozesse ist eine Managementaufgabe [PRR99, Ame00]. Die Lernprozesse selbst werden durch eine Vielzahl von Lerntheorien beschrieben, die in die Gebiete der Psychologie, Pädagogik (insbesondere der pädagogischen Psychologie) und Soziologie fallen [Ede96].

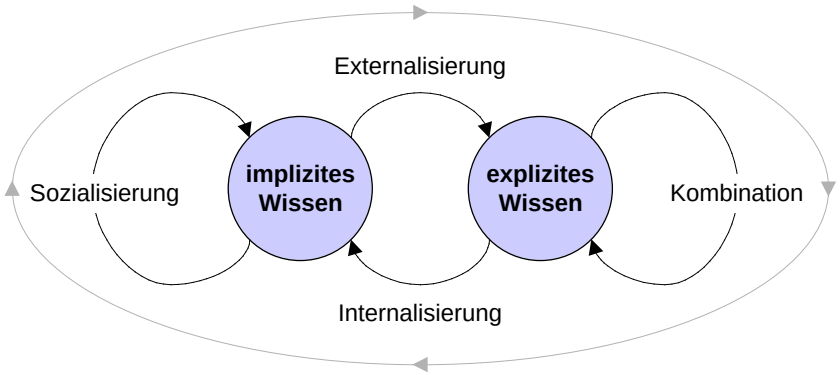

Abbildung 2.4: Arten der Wissenskonvertierung

Sehr aufschlußreich sind die vier Arten der Wissenskonvertierung, die [NT95] innerhalb der epistemologischen Dimension identifiziert haben. Es gibt demnach vier verschiedene Prozesse der Überführung von explizitem und implizitem Wissen in allen Kombinationen. Abbildung 2.4 stellt die Begriffe und die Richtungen in einer anderen graphischen Notation als in [NT95, S. 62] dar, nämlich als Zustandsübergänge. Der umgebende Kreis macht die Richtung der Wissensspirale [NT95, S. 71] deutlich, die beim fortwährenden Übergang entsteht. Als Schlüsselressource von Organsationen wird daher vielfach gerade das implizite Wissen angesehen; das Ziel vieler Bemühungen des Wissensmanagements ist daher die Nutzbarmachung – also Explizierung – dieser Art von Wissen. Dabei tritt paradoxerweise die Situation auf, daß ein Erfolg dabei das nun explizite Wissen angreifbar für Replikation macht und so die Gefahr schafft, daß eine Organisation dadurch ihren Wissensvorsprung verliert [Sch99b].

Diese Prozesse der Wissenskonvertierung sind auf das Individuum bzw. die Gruppe bezogen; weitere wirtschaftliche, organisationale und kulturelle Einflüsse sind dabei noch nicht systematisch in Betracht gezogen worden. Diese praktisch orientierten Überlegungen werden im nächsten Abschnitt weiter ausgebaut.

2.4 Der Wissenszyklus

Das organisationale Lernen als zentrale Aufgabe des (betriebswirtschaftlichen) Wissensmanagements muß durch praxistaugliche Methoden und Konzepte unterstützt werden. In [PRR99] wird ein pragmatisches Wissensmanagement-Konzept dargestellt, das mittlerweile zumindest im deutschsprachigen Raum weite Verbreitung gefunden hat (siehe etwa [BBM01, Gro01, BÖV00, RMS00, AS99, BVÖ99, Gaß99, Hab99, GFS98, BWP97]) und als anerkanntes Modell gilt.

Die in [PRR99] sogenannten *Bausteine* des Wissensmanagements bilden einen Regelkreis. Es gibt in dem Zyklus acht Elemente, die in Abbildung 2.5 grob der ursprünglichen Illustration folgend dargestellt sind.

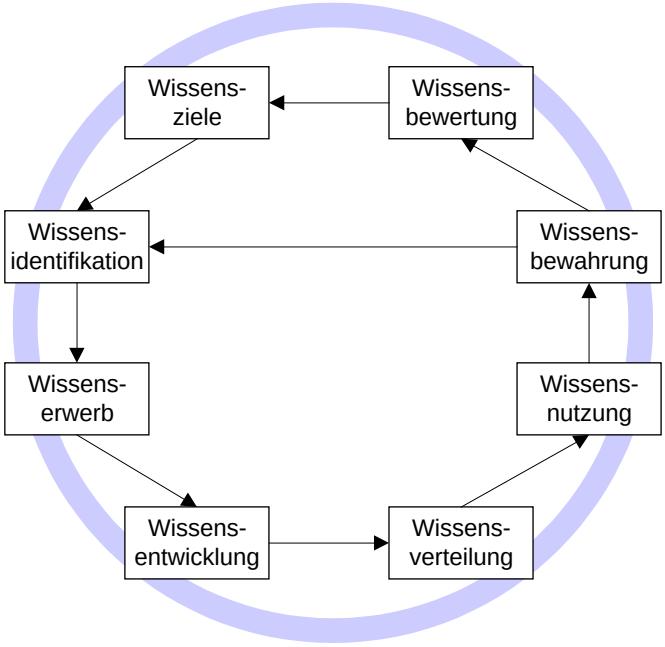

Abbildung 2.5: Bausteine des Wissensmanagements

Die im folgenden beschriebenen Aufgaben oder Kernprozesse stehen in enger Verbindung miteinander. Eine isolierte Betrachtung oder ein isoliertes Herangehen an einzelne Aufgaben ist im Rahmen eines unternehmensweiten Wissensmanagement-Konzeptes meistens nicht ratsam; alle Aufgaben müssen in einem gemeinsamen organisatorischen Rahmen seitens des Managements angegangen werden [JBÖ00]. Dies geht natürlich weit über die informationstechnischen Aspekte hinaus, und so auch über den Rahmen dieser Arbeit. Für die Fundierung der informationstechnischen Aufgaben und die von ihr anzubietenden Lösungsmöglichkeiten ist eine übersichtsartige Beschreibung der grundsätzlichen Ziele, Aufgaben und Prozesse dennoch unerläßlich, da die Informationstechnik wie immer stets nur Mittel zum Zweck ist.

Bei der Analyse der jeweiligen Aufgaben soll jeweils untersucht werden, inwieweit die Informationstechnik bereits geeignete Hilfsmittel, Werkzeuge und Verfahren liefert. Dies ist zunächst nur als begleitende Bestandsaufnahme zu begreifen; eine aus Sicht der Informationstechnik systematische Zusammenfassung der verwendeten Konzepte und Systeme wird im Anschluß in Kapitel 3 gegeben.

2.4.1 Wissensziele

Wenn man nicht weiß, wohin man geht, tut die Geschwindigkeit dabei nicht viel zur Sache.
– CEES NOOTEBOOM, Die folgende Geschichte

Eine Definition der Wissensziele bildet den Ausgangspunkt aller diesbezüglichen normativen, strategischen und operativen Managementziele. Es wird festgelegt, welches Kompetenzprofil

(Kernkompetenzen) das Unternehmen erhalten soll, welche Strukturen, Aktivitäten und welches Verhalten (z.B. Unternehmenskultur) erreicht werden soll.

Typische Hindernisse und Probleme beim Formulieren solcher Ziele sind z.B. auf der einen Seite alte Gewohnheiten, Machtinteressen und auf der anderen Seite z.B. unklar definierte Unternehmensvisionen (nicht oder schlecht meßbare Ziele), das Fehlen von Verantwortlichkeiten, mangelnde Einführung (z.B. durch *coaching*) oder nicht wahrgenommene Vorbildfunktionen. Ein weiterer Grund für das Scheitern von Wissensmanagementprojekten kann eine einseitige Technologieorientierung sein, die organisatorische und psychologische Faktoren außer acht läßt [SA99].

Festzuhalten bleibt, daß die Definition der Wissensziele in erster Linie eine Managementaufgabe ist und eng mit Fragen der Unternehmenskultur zusammenhängt. Diese Aspekte sind bisher durch informationstechnische Hilfsmittel nur äußerst begrenzt unterstützbar. Die jetzige Rolle der Informationstechnik liegt hierbei lediglich in der Bereitstellung der Kommunikationsinfrastruktur, die dabei hilft, die Visionen und Ziele den Adressaten zu vermitteln, als alleiniges Medium dafür ist sie aber sicher ungeeignet.

2.4.2 Wissensidentifikation

„Wenn ihr Unternehmen wüsste, was es alles weiß“ lautet der provokante Titel von [DP99] und macht so das Hauptproblem der Wissensidentifikation deutlich. In diesem Zusammenhang werden weitere Paradoxien sichtbar [PRR99]:

- Wir bilden unsere Mitarbeiter gründlich aus, aber lassen sie ihr Wissen nicht anwenden.
- Wir lernen am meisten in Projekten, aber geben die gemachten Erfahrungen nicht weiter.
- Wir wissen alles über unsere Konkurrenten, aber nur wenig über uns selbst.
- Wir dokumentieren alles gründlich, aber können nicht auf unsere Wissensspeicher zugreifen.
- Wir fordern jeden zur Wissensteilung auf, aber behalten Geheimnisse für uns.
- Wir haben für jede Frage einen Experten, aber die wenigsten wissen, wie man ihn findet.

Die Gründe dafür sind vielfältig; Dezentralisierung, Ausgliederung, globales Engagement und hohe Fluktuation sind nur einige davon, wesentlicher ist meist das Fehlen einer übergreifenden Wissensstrategie.

Die Frage der Wissensidentifikation ist also „Wer hat welches Wissen?“. Häufig ist es sinnvoll, in den Personalabteilungen mit der Analyse zu beginnen, da dort oft Aufgaben- und Stellenbeschreibungen sowie Qualifikationsnachweise zu finden sind [RMS00]. Sofern bereits ein gezieltes Qualifikationsmanagement besteht, kann dies genutzt werden. Zum Einsatz kommen weiterhin Techniken wie *benchmarking*, zunehmend mit dem Ziel, sogenannte *best practices* zu bestimmen [Töp97]. *Benchmarking* dient dem Leistungsvergleich mit der Konkurrenz, während die Ermittlung von *best practices* zum Ziel hat, innerhalb des Unternehmens Leistungsvergleiche zu ziehen und so die erfolgreichsten Methoden zu identifizieren. Schließlich muß bei der Wissensidentifikation transparent gemacht werden, wer welche Kompetenzen besitzt, Projekterfahrungen hat und welche informellen Strukturen bestehen. Besonders kritisch ist die Identifikation kollektiven Wissens. Zusätzlich sollte versucht werden, das externe Wissen, also das von Geschäftspartnern, Beratern, Kunden etc. zu identifizieren.

Die Rolle der Informationstechnik ist im Zuge der Identifikation zunächst nur eine unterstützende. Sie hilft bislang bei der Sicherung und Darstellung der bereits identifizierten Wissensquellen. Werkzeuge sind etwa Wissenslandkarten oder „gelbe Seiten“ mit Verzeichnissen von Experten. Auch bei der eigentlichen Identifikation bietet heute das Intranet neue Möglichkeiten, z.B. durch interne Web-Server und Diskussionsforen. Die Analyse der Nutzung und der Inhalte von Kommunikations- und Kollaborationswerkzeugen (*Groupware*) kann wichtige Aufschlüsse über Ressourcen liefern.

Eine andere Rolle spielt die Informationstechnik als externer Informationsanbieter: Durch die bereits anfangs (Abschnitt 1.1) beschriebene Entwicklung des WWW ist heute für Unternehmen der Zugriff auf eine unüberschaubare Menge von zusätzlicher Information möglich. Diese zunächst unstrukturierte Fülle gemäß den Informationsbedürfnissen des Unternehmens nutzbar zu machen stellt eine weitere Herausforderung für die Wissensidentifikation dar. Die Frage lautet dann nicht „Wer hat welches Wissen?“, sondern „Wo ist welches Wissen verfügbar?“.

2.4.3 Wissenserwerb

Nach dem in Abbildung 2.5 illustrierten Zyklus der einzelnen Aktivitäten kann nach erfolgter Definition der Wissensziele und der Bestandsaufnahme durch die Wissensidentifikation daran gegangen werden, neues Wissen gemäß den Zielvorgaben aufzubauen. Häufig entsteht die Situation, daß ein Teil des benötigten Wissens im Unternehmen einfach nicht vorhanden ist. Es bietet sich dann an, externes Wissen zu erwerben bzw. zu mieten. Dies kann auf verschiedene Arten geschehen:

- Beauftragung von Unternehmensberatungen. Dies ist ein gängiges Verfahren, um vergleichsweise schnell zu Ergebnissen zu kommen. Ein Problem besteht dabei in der Sicherung der gewonnenen Erkenntnisse, nachdem das Projekt abgeschlossen ist und die externen Berater das Haus wieder verlassen haben.
- Einstellung von neuen Mitarbeitern, sei es fest oder für einen beschränkten Zeitraum (z.B. als Freiberufler). Um an genügend hoch qualifizierte Mitarbeiter heranzukommen, bedarf es einer sorgfältigen Beobachtung und Pflege der jeweiligen Wissensmärkte, z.B. durch gezielte Kontakte und Kooperationen mit Hochschulen. Dieser Aspekt ist Aufgabe des Personalmarketings.
- Integration des Wissens der *Stakeholder*, also der im Umfeld des Unternehmens angesiedelten Personen und Gruppen wie Eigentümer, Banken, Kunden etc. Insbesondere das Wissen der Kunden aufzunehmen erweist sich als wichtig.
- Bilden von Kooperationen mit anderen Institutionen (Firmen, Universitäten etc.). Das mögliche Spektrum ist sehr breit und reicht von loser, informeller Zusammenarbeit über *Joint-Ventures* bis hin zu Übernahmen und Fusionen. Gerade durch letzteres versprechen sich in letzter Zeit multinationale Konzerne deutliche Wettbewerbsvorteile. Kritisch ist dabei insbesondere die Art und Weise der Übernahme, da dann verschiedene Unternehmenskulturen integriert werden müssen, und so auch evtl. völlig unterschiedliche Strategien für das Wissensmanagement.
- Einkauf von Informationsmaterial, also Informationsträgern im Sinne der Definitionen aus Abschnitt 2.2, wie z.B. Büchern, Lexika oder CDs.
- Fortbildung der Mitarbeiter durch interne und externe Schulungen und Kurse. Gerade der kommerzielle Schulungsmarkt erfreut sich großer Beliebtheit, was die Wichtigkeit dieser Art des Wissenserwerbs bestätigt.

Der Wissenserwerb ist nur schwer durch die Informationstechnik systematisch zu unterstützen, da es sich um viele unterschiedlich gelagerte und unregelmäßig stattfindende Maßnahmen handelt. Betrachtet man die Dokumentation des *externen* bzw. des *fehlenden* Wissens, das erworben werden muß, als Aufgabe der Wissensidentifikation, so kann die Informationstechnik dafür die gleichen Leistungen erbringen. Andererseits kann z.B. neu erworbenes Informationsmaterial meist effizient in bestehende Lösungen für das Wissensmanagement eingebunden werden. Alleine die Information, daß neues Wissen vorhanden ist, ist oft schon wichtig. Dies bekannt zu machen ist dann Aufgabe der Wissensverteilung.

Ein Aspekt der informationstechnischen Unterstützung, der in der letzten Zeit stark wachsende Aufmerksamkeit erfahren hat, ist der des elektronisch unterstützten Lernens (*e-Learning*) [Sch97b]. In kurzer Zeit sind viele neuartige Systeme entstanden, die mit Metaphern wie virtuellen Klassenzimmern oder Schulungszentren etc. arbeiten (z.B. [Hyp01b]), und dazu Funktionen wie interaktive Schulungsmaterialien, Diskussionsforen, Online-Tests und strukturierten Zugriff auf Lehrmaterialien über z.B. das Intranet bieten. Gerade die Fernuniversitäten haben naturgemäß großes Interesse an solchen Lernumgebungen [Sch01].

2.4.4 Wissensentwicklung

Die Wissensentwicklung ist eine der wichtigsten Aufgaben im Wissensmanagement. Sie dient dazu, systematisch aus dem bestehenden (oder gerade frisch erworbenen) Wissen noch nicht vorhandenes Wissen und Fähigkeiten zu produzieren. In großen Organisationen ist dies normalerweise die Aufgabe einer dedizierten Forschungs- und Entwicklungsabteilung. Wesentlicher Erfolgsfaktor ist die systematische Vorgehensweise zum Lösen von Problemen gepaart mit Kreativität. Beides in Hinblick auf die vorgegebenen Wissensziele in Balance zu halten ist dabei essentiell. Damit ist die Wissensentwicklung insbesondere eine organisatorische und kulturelle Aufgabe: Die Schaffung eines Klimas, das Kreativität, Freiräume, Handlungsentlastung, Eigeninitiative, persönliche Interessen, Wissensaustausch, Fehlertoleranz und Vertrauen fördert, ist unerläßlich.

Auf die individuelle Ebene bezogen, ist die Wissensentwicklung der eigentliche kreative Anteil der Arbeit, der im Kopf von Personen abläuft und direkt Wissen verarbeitet. Diese Leistung kann nicht direkt von Informationssystemen erbracht oder unterstützt werden. Informationssysteme spielen daher nur eine untergeordnete Rolle, wichtig ist vor allem das Festhalten des Gelernten, z.B. der gemachten Erfahrungen (*lessons learned*) und der besten Vorgehensweisen (*best practices*). Die konkrete Durchführung dieses Festhaltens ist Aufgabe der Wissensverteilung.

2.4.5 Wissens(ver)teilung

Ebenso wichtig wie die Wissensentwicklung sind die Wissensverteilung (als die – evtl. informationstechnisch unterstützte – Distribution von Informationen) und Wissensteilung (als die persönlichen Bereitschaft einzelner, Wissen weiterzugeben); sie wird vielfach sogar als eine der wichtigsten Aufgaben des Wissensmanagements angesehen. Ihrem guten Funktionieren kommt eine „Hebelfunktion“ für das Wissensmanagement zu [PRR99]. Die Wissensverteilung umfasst die Multiplikation von Wissen, d.h. die möglichst schnelle Verteilung an viele Personen, die Sicherung und Teilung bereits gemachter Erfahrungen (*lessons learned* und *best practices*) sowie den permanenten Wissensaustausch (Wissenstransfer) zum Zwecke der erneuten Wissensentwicklung. In diesem Zusammenhang wird vielfach der Begriff der *Communities of practice* als informelles Instrument zur Wissensverteilung genannt [NRP00, Sch00a, Bor00].

Bei der Verteilung werden oft Barrieren sichtbar, die in der Regel auf Macht- und Vertrauensfragen hinauslaufen. So sind Mitarbeiter oftmals nicht bereit, ihr Wissen zu teilen, da sie fürchten, Machtverlust zu erleiden, ihren Kompetenzvorsprung zu verlieren oder sogar überflüssig zu werden [DH99]. Neben diesen individuellen Hürden sind weitere organisationale (funktionale) und hierarchische Barrieren auszumachen, die die Wissensverteilung grundsätzlich erschweren: Hierarchische Barrieren schotten die einzelnen Führungsebenen einer Organisation voneinander ab, während funktionale Barrieren Abteilungen auf gleicher Ebene voneinander isolieren. Das Auftreten von beiden Arten zusammen kann zu der Entstehung von relativ unverbundenen „Wissensinseln“ führen (vgl. Abbildung 2.6).

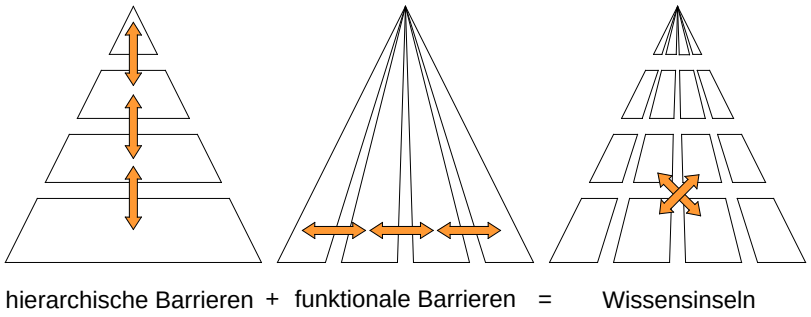

Abbildung 2.6: Wissensbarrieren

Um die Wissens(ver)teilung zu fördern sind ständige Interaktion und Kommunikation der Betroffenen nötig. Besonders wichtig ist der Wissenstransfer: die Explizierung impliziten Wissens spielt dafür eine herausragende Rolle. Die Schaffung eines Klimas, das das Entstehen von Barrieren verhindert und das den Wissensaustausch auf persönlicher Ebene sowie die Nutzung von unterstützenden Informationssystemen für das Wissensmanagement fördert, ist genauso wie für die Wissensentwicklung (vgl. Abschnitt 2.4.4) unerläßlich. Materielle oder ideelle Anreize, die extrinsische oder intrinsische Motive der Mitarbeiter berücksichtigen, können die Bereitschaft zur Teilnahme nachhaltig erhöhen.

Die Informationstechnik spielt für die Unterstützung dieser Aufgabe eine herausragende Rolle. Es gibt bereits zahlreiche Systeme für Kommunikation und Kollaboration, die die Verteilung von Informationen unterstützen. Ein oft verwendetes Informations-Paradigma ist das der *push*-Technik, d.h. der aktiven Zustellung von neuen Informationen durch ein System an einen Benutzer. Dies ist im Gegensatz zur *pull*-Technik zu sehen, bei der der Benutzer aktiv werden muß, um neue Informationen zu holen. Werkzeuge wie LOTUS NOTES, e-Mail-Systeme und firmeneigene Intranets stellen Beispiele für beide Arten dar.

Die Limitierungen, die gerade beim Einsatz von Informations- und Kommunikationssystemen erreicht werden, sind der begrenzte Nutzen von Informationen, wenn Informationsüberflutung einsetzt. Nicht jeder Nutzer benötigt jede Information, da sonst ohnehin das Prinzip der Arbeitsteiligkeit konterkariert würde (Effizienz). Auch aus Gründen der Vertraulichkeit darf nicht jede Information jede Person erreichen, und Machtfragen stellen sich wieder [Sch99b]. Eine angemessene Filterung und aufgabenbezogene Verteilung ist also Voraussetzung erfolgreicher Wissensverteilung. Fortschrittlichere Systeme – möglicherweise rein organisatorischer Art – sind in der Lage, Nutzerinteressen zu berücksichtigen und Inhalte zu klassifizieren, indem sie angemessen gepflegt und aufbereitet werden. Eine breitere Darstellung von Systemen und Anforderungen findet in Kapitel 3 statt.

2.4.6 Wissensnutzung

Nach Wissensidentifikation, -entwicklung und -teilung muß das Wissen auch genutzt werden. Dazu gehört einerseits, daß das Wissen in einer nutzbaren Form angeboten wird, andererseits muß die Nutzungsbereitschaft der angesprochenen Personen hergestellt werden. Beides ist wiederum eine Führungsaufgabe. Die Nutzungsbereitschaft wird ähnlich wie bei der Verteilung potentiell durch Nutzungsbarrieren erschwert. Betriebsblindheit, Angst vor Bloßstellung eigener Unwissenheit und Mißtrauen gegenüber fremdem Wissen können Gründe dafür sein. Die Bereitschaft zur Teilnahme kann durch nutzungsorientierte Gestaltung von Arbeitssituationen (etwa entsprechende Arbeitsplatzgestaltung und Präsentation der Informationen) und die Einbettung der Maßnahmen des Wissensmanagements in den Handlungszusammenhang der Mitarbeiter erreicht bzw. verbessert werden. Dies gilt im übrigen für alle Bausteine des Wissensmanagements gleichermaßen.

Insbesondere die nutzungsgerechte Präsentation der Informationen ist eine Aufgabe der Informationstechnik. Um Benutzbarkeit von Informationssystemen (z.B. Verständlichkeit, Erlernbarkeit und Bedienbarkeit) zu erreichen, müssen Erkenntnisse der Softwareergonomie berücksichtigt werden [EOO94, Bal01]. Die Oberfläche der Systeme muß demnach so gestaltet werden, daß eine hohe Akzeptanz durch die Benutzer erreicht wird, indem die Standardaufgaben einfach und effizient unterstützt werden.

2.4.7 Wissensbewahrung

Wenn Wissen erfolgreich erworben und genutzt wird, wird die Sicherung oder Bewahrung dieses Wissens zu einer wichtigen Aufgabe. Die wesentliche Aufgabe der Wissensbewahrung besteht darin, das Unternehmensgedächtnis (*organizational memory*) aufzubauen, zu pflegen und davor zu bewahren, daß Erinnerungen verloren gehen. Normalerweise hat jede Organisation ihr kollektives Gedächtnis, das sich aus dem Wissen der einzelnen Mitarbeiter zusammensetzt. Die gemeinschaftliche Leistung wird häufig durch die Gruppe erst ermöglicht. Das Unterschätzen dieser Bedeutung und des Wissens altgedienter Mitarbeiter, die dafür nicht einmal eine Schlüsselfunktion wahrnehmen müssen, führt bei Reorganisationsprozessen oft zu überraschenden Schwierigkeiten. Die Förderung der systematischen Weitergabe von Kenntnissen und Fähigkeiten (etwa durch *Communities of practice* etc.) ist deshalb neben der Errichtung von Austrittsbarrieren (etwa durch materielle Anreizsysteme) eine wichtige organisatorische Aufgabe.

Wichtigstes Werkzeug, Informationsträger und Medium zugleich für kollektive Firmengedächtnisse als solche sind bestimmte Informationssysteme, nämlich die sogenannten *Organizational Memory Information Systems* (OMIS). Ihre Aufgabe ist es, alle relevanten Dokumente, z.B. Protokolle, *lessons learned*, *best practices* etc. zentral zu speichern und den Zugriff darauf sicherzustellen [ABH+98]. Die Klasse der Dokumentenmanagement-Systeme (DMS), die in Abschnitt 3.3.1 ausführlich untersucht werden wird, spielt ebenfalls eine dafür wichtige Rolle, wenn auch die Intention eine etwas andere ist (vgl. Abschnitt 3.3.1).

Bei der Nutzung von OMIS ergeben sich wichtige Anforderungen. So muß neben organisatorischen und kulturellen Voraussetzungen (Nutzungsbereitschaft) sichergestellt werden, daß ein gemeinsames Vokabular benutzt wird und daß die Informationen regelmäßig aktualisiert werden, um den Kampf gegen das organisationale Vergessen zu führen. Ein ebenso wichtiger Punkt ist auf der anderen Seite die Sorge dafür, daß veraltete Informationen ausgesondert werden. Viele Untersuchungen fordern daher die Einführung von expliziten Rollen für Informationsmakler und Multiplikatoren [KLT00]. Informationsmakler sollen das Systems fachinhaltlich betreuen und kommunikative Prozesse moderieren, während Multiplikatoren die gruppenorientierte Ver-

teilung der Informationen sicherstellen sollen. Auch diejenigen Personen, die den Zugang zu bestimmten Informationsbeständen regeln (*gatekeeper*), finden sich durch die Einführung von OMIS und Intranet-Technik in neuen Rollen, z.B. als Informationsmakler [Zac00].

2.4.8 Wissensbewertung

> *When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind: it may be the beginning of knowledge, but you have scarcely in your thoughts advanced to the stage of science.*
> – LORD KELVIN

> *Was man nicht messen kann, das kann man auch nicht managen!*
> – anonyme Managerweisheit [PRR99]

Wie bereits in der Einleitung (Abschnitt 1.1) ausgeführt, wird das Wissenskapital einer Organisation zum wettbewerbsentscheidenden Faktor [EHB+00]. Da Wissensmanagement eben als Managementaufgabe begriffen wird, muß der Erfolg für das Controlling auch meßbar gemacht werden. Die Schwierigkeit besteht darin, die nur schwer greifbare Ressource Wissen bzw. den Wissenszuwachs objektiv zu messen. Dies ist gegenwärtig ein besonders offenes Gebiet, und gesicherte Ansätze gibt es erst wenige. Die Hauptprobleme bei der Messung sind: das Übersehen wichtiger Ressourcen, Messen der falschen Ressourcen, Messen mit falschen Maßstäben oder Messen ohne klare Zielvorstellungen.

Ansätze für das Messen von Wissen sind etwa mit den *Balanced Scorecards* [KN96] vorhanden, dem *Intangible Assets Monitor* [Sve97a] oder dem *Skandia Navigator* [Sve97b, Sve98]. Insbesondere der schwedische Finanzdienstleister SKANDIA hat sowohl im Wissensmanagement überhaupt als auch speziell in der Wissensbewertung ein richtungsweisendes Vorgehen gezeigt, das seitdem als die klassische Erfolgsgeschichte [PRR99] des Wissensmanagements gilt.

Die Informationstechnik dient hier eher der Informationsbereitstellung für das Controlling (*Management Support Systeme* und *Business Intelligence*). Aktuelle Schlagworte sind technische Unterstützung für *Balanced Scorecards* [Möl00, KN01, Kap01] und besonders *Data Warehousing* und *Online Analytical Processing* (OLAP) [Glu97, PN00].

2.5 Vergleich von Bausteinen des Wissensmanagements

Nach der vorangegangenen Betrachtung der Bausteine des Wissensmanagements, die an der Einteilung von [PRR99] orientiert ist, soll diese Sicht zu einigen anderen in Beziehung gesetzt und mit ihnen verglichen werden. Interessante Ansätze dafür liefern:

- [Kap99b] mit einer Darstellung des Wissenstransferzyklus, der als wichtiges Mittel zur Externalisierung und Internalisierung von Wissen bzw. Information betrachtet wird. Interessant ist die Einordnung der einzelnen Schritte innerhalb der Hierarchie von Daten, Information und Wissen (vgl. Abbildung 2.7). Auffällig ist dabei, daß der Schritt der Organisation des Wissens (Strukturierung, Annotation etc.) auf der Ebene der Daten angesiedelt ist. Insgesamt liefert diese Betrachtung ein stark auf den Bereich Informationstechnik konzentriertes Bild, das aber die Abgrenzung zu Fragen der Organisation deutlich macht.

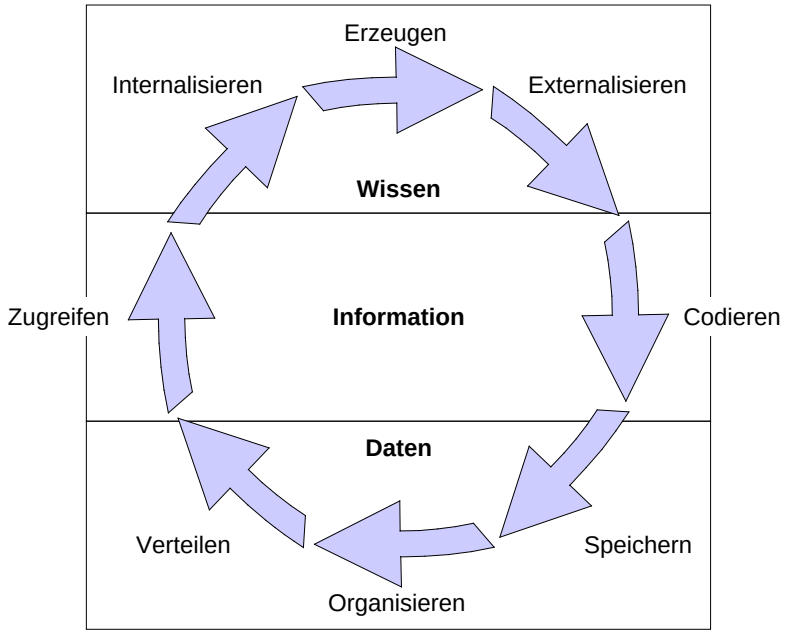

Abbildung 2.7: Wissenstransferzyklus nach [Kap99b]

- [SAD99] stellen die Hauptaufgaben eines *Organizational Memory Information Systems* (OMIS) als Wissenserfassung und Wissenspflege, Wissensaufbereitung und Wissensintegration sowie Wissenssuche und Wissensnutzung dar und gehen dabei explizit auf Informatikmethoden ein, die zur Unterstützung verwendbar sind. Diese reichen von Textanalyse, *Knowledge Discovery*, *Data Warehousing* und OLAP für Erfassung und Pflege über Ontologien, Wissensanalyse, Klassifikation, Metadatenmodellierung und Visualisierung für Aufbereitung und Integration bis hin zu kooperativen Informationssystemen, persönlichen Assistenten, Expertensystemen, Problemlösungsmethoden und fallbasiertem Schließen für Suche und Nutzung. Diese Methoden bzw. Systeme werden im folgenden Kapitel näher diskutiert.

- [BWP98] stellt eine [PRR99] recht ähnliche Sichtweise dar, die etwas konkretere Teilaufgaben beschreibt und weniger Managementaspekte (Zieldefinition und Wissensbewertung). So entfallen etwa Wissenserwerb und Wissensbewahrung; die Wissensverteilung gliedert sich nochmals in Wissensspeicherung und Wissensverteilung.

Ein weitergehender Vergleich von Strukturierungen ist etwa in [Gaß99, S. 19 ff.] zu finden. Eine Gegenüberstellung der verwendeten Begriffe ist in Tabelle 2.1 vorgenommen worden. Dabei wurde versucht, die korrespondierenden Bausteine zu finden und einander zuzuordnen. Etwas außerhalb der Vergleichsmöglichkeiten stehen hier die Zieldefinition, Wissensidentifikation und der Wissenserwerb, sowie schließlich die Wissensbewahrung und Wissensbewertung, da diese nicht in allen Modellen vorkommen und zudem eher managementorientiert als informationstechnisch orientiert sind und somit im Rahmen dieser Arbeit keine große Rolle spielen. Ein Bereich guter Übereinstimmung ist in Wissensentwicklung, Wissensverteilung und Wissensnutzung zu sehen (den Begriffen von [PRR99] folgend). In allen Modellen ist der kreative Teil des Wissenserwerbs derjenige Teil, der prinzipiell in den Köpfen der Personen abläuft und daher nicht direkt durch Informationstechnik unterstützbar ist. Dazu gehören insbesondere die Internalisierung expliziten Wissens, der eigentliche intellektuelle Prozeß der Erzeugung neuen

<table>
<tr><th>[PRR99]</th><th>[BWP98]</th><th>[SAD99]</th><th>[Kap99b]</th></tr>
<tr><td>Zieldefinition</td><td>Zieldefinition</td><td rowspan="3"></td><td rowspan="3"></td></tr>
<tr><td>Identifikation</td><td>Identifikation</td></tr>
<tr><td>Erwerb</td><td></td></tr>
<tr><td rowspan="3">Entwicklung</td><td rowspan="3">Erzeugung</td><td rowspan="3"></td><td>Internalisierung</td></tr>
<tr><td>Erzeugung</td></tr>
<tr><td>Externalisierung</td></tr>
<tr><td rowspan="5">Verteilung</td><td rowspan="2">Speicherung</td><td rowspan="2">Erfassung</td><td>Codierung</td></tr>
<tr><td>Speicherung</td></tr>
<tr><td rowspan="4">Verteilung</td><td>Pflege</td><td rowspan="3">Organisation</td></tr>
<tr><td>Aufbereitung</td></tr>
<tr><td>Integration</td></tr>
<tr><td></td><td>Verteilung</td></tr>
<tr><td rowspan="2">Nutzung</td><td rowspan="2">Anwendung</td><td>Suche</td><td></td></tr>
<tr><td>Nutzung</td><td>Zugriff</td></tr>
<tr><td>Bewahrung</td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>Bewertung</td></tr>
</table>

Tabelle 2.1: Vergleich der Bausteine des Wissensmanagements

Wissens und schließlich die Externalisierung der Ergebnisse zurück in explizites Wissen. Dieser Teil wird gut durch die entsprechenden Wissenskonvertierungen in 2.4 beschrieben. Die Externalisierung und Codierung (in Begriffen von [Kap99b]) hängt dabei natürlich eng zusammen. Codierung und Speicherung, auch Erfassung in [SAD99] genannt, zählen in den Bausteinen nach [PRR99] bereits zur Verteilung. Dazu gehört dann auch die gesamte Organisation in Form von Pflege, Aufbereitung und Integration der Informationen. Hier ist – wiederum dem Schema aus Abbildung 2.7 folgend – bewußt von Informationen die Rede, da dies durch Informationssysteme gut unterstützbar ist. Gleiches gilt für die Nutzung, die Suche und Zugriff enthält. Gemäß des Wissenstransferzyklus von [Kap99b] finden alle Prozesse außer der Wissensentwicklung auf der Ebene der Informationen bzw. Daten statt. Damit ist der Bereich identifiziert, der angemessen durch die Informationstechnik unterstützbar ist, nämlich genau die Wissensverteilung und Wissensnutzung. Dieser Bereich ist in der Tabelle 2.1 grau hinterlegt.

Die groben Funktionen der Informationstechnik sind bereits in Abschnitt 2.4 erläutert worden, eine feinere Analyse und Darstellung folgt in Kapitel 3.

2.6 Zusammenfassung

In den vorangegangenen Abschnitten wurde deutlich, daß das Wissensmanagement einen mittlerweile festen Platz innerhalb der Wissenschaftsdisziplinen erreicht hat. Es wurden zahlreiche Modelle entwickelt, die die Aufgaben in unterschiedlicher Breite und mit unterschiedlichen Schwerpunkten betrachten, wobei bestimmte Kernaufgaben – zwar mit verschiedenen Abgrenzungen und verschiedenen Namen und Beschreibungen – stets in ähnlicher Form wiederkehren. Der gemeinsame Nenner der Untersuchungen ist häufig die Unterscheidung expliziten und impliziten Wissens. Eine präzise Abgrenzung der drei Begriffe Daten, Information und Wissen

voneinander findet oft nicht statt; wenn sie dennoch vorgenommen wird, ergeben sich meist sehr unterschiedliche Resultate.

Allen Herangehensweisen gemeinsam ist ist die Feststellung, daß erfolgreiches Wissensmanagement sich auf keinen Fall auf isolierte Aspekte, z.B. ausschließlich auf die Informationstechnik, konzentrieren darf, sondern die organisationalen und kulturellen Aufgaben zusammen mit den technischen angehen muß [Due01]. Wissensmanagement bleibt so immer eine Managementaufgabe im eigentlichen Sinne. Die Informationstechnik bzw. die Informatik bleibt damit in der klassischen Rolle, unter Berücksichtigung aller dieser Bedingungen adäquate Werkzeuge zusammen mit Metaphern, Prozessen und Systemen zu entwickeln, die die Gesamtaufgabe des Wissensmanagements möglichst gut unterstützen. Nicht alle Teilprozesse und Aufgaben des Wissensmanagements lassen sich gleichermaßen gut durch Informationssysteme unterstützen, insbesondere die organisationalen Aufgaben (Unternehmensorganisation, Personalmanagement etc.) sind hier grundsätzlich problematisch.

Als Grundlage für Wissensmanagement existieren verschiedene Modelle, die die Ziele, Aufgaben und Prozesse des Wissensmanagements strukturieren. Allen gemeinsam ist eine zyklische Struktur, die zumindest – unterschiedlich strukturiert – Phasen wie Entwicklung, Verteilung und Nutzung vorsehen. Diese Phasen, und insbesondere die Bausteine nach [PRR99], sind dabei nicht strikt zyklisch und als scharf voneinander getrennt zu verstehen, sondern decken teilweise etwas überlappende Aufgabenbereiche ab. Einige der Kernaufgaben lassen sich weitreichend unterstützen. Dazu zählen insbesondere die Wissensverteilung und Wissensnutzung; dies trägt der Einschätzung der Wissensverteilung als „Hebelfunktion“ Rechnung. Die Wissensbewahrung geschieht bei Nutzung von Informationssystemen implizit.

Es existiert schon eine Fülle von Modellen und Werkzeugen unterschiedlicher Art für solche Systeme. Diese näher zu untersuchen und zu klassifizieren ist Aufgabe des folgenden Kapitels.

Kapitel 3

Informationssysteme zur Unterstützung des Wissensmanagements

Im vorangegangenen Kapitel 2 wurden die wesentlichen Aufgaben im Wissensmanagement in Form von diversen Bausteinen beschrieben. Als Bausteine, die besonders gut durch die Informationstechnik unterstützbar sind, wurden die Wissensverteilung und Wissensnutzung identifiziert. In diesem Kapitel wird die Unterstützung des Wissensmanagements durch Informationssysteme weiter untersucht, wobei hier der Fokus auf der Frage liegt, welche Systeme welche Aufgaben in welcher Weise unterstützen. Dazu werden sowohl ganze Systemklassen als auch ausgewählte konkrete Systeme analysiert. Ziel dieses Kapitels ist es, diesbezüglich zunächst eine an den Aufgaben des Wissensmanagements orientierte Bestandsaufnahme zu machen; und zwar sowohl auf der Ebene der Abdeckung der fachlichen Anforderungen als auch auf der Ebene der konzeptuellen Modelle. Aus den Ergebnissen dieser Untersuchungen wird die Kernthese dieser Arbeit entwickelt werden, daß für beide Ebenen ein problemadäquates, integriertes Modell noch nicht existiert. Auf dieser These fußen die in den folgenden Kapiteln vorgenommene Entwicklung eines an den Bausteinen des Wissensmanagements orientierten integrierten fachlichen Anforderungskatalogs und der Entwurf eines problemadäquaten Dokumentenmodells, die beide zusammen die Grundlage des softwaretechnischen objektorientierten Entwurfs bilden werden.

Zur Untersuchung der Informationssysteme wird zunächst in Abschnitt 3.1 ein kurzer historischer Rückblick auf die Art und Weise des Einsatzes von Informationssystemen in den letzten 30 Jahren gegeben, um die derzeitige Situation zu illustrieren, die durch den immensen Erfolg von Internet-Informationssystemen geprägt ist. Das hier entstehende neue Paradigma bildet später die Grundlage der Kernthese der Arbeit. Zur weiteren Strukturierung des Untersuchungsgegenstandes wird in Abschnitt 3.2 eine Klassifikation von Informationssystemen vorgenommen. Es wird dabei die produktorientierte von der prozeßorientierten Art unterschieden, sowie eine Aufteilung anhand des Strukturierungsgrades der verwalteten Informationen vorgenommen. Es wird zu erkennen sein, daß klassische Informationssysteme jeweils nur eingeschränkte Bereiche unterstützen, aber für die Aufgaben des Wissensmanagements vielfältige Arten von Prozessen und Informationen benötigt werden. Nach der Betrachtung der Informationssysteme von der strukturellen Seite her folgt in Abschnitt 3.3 eine ausführliche Analyse der Systemklassen der Dokumenten-Management-Systeme, der Content-Management-Systeme und der Portale, wobei

zunächst jeweils eine genauere Definition und Abgrenzung von den anderen Systemen vorgenommen wird. Diese ist unerläßlich, da die Begriffsbildung dort bisher äußerst uneinheitlich ist. Es wird dann jeweils untersucht, welche Funktionen diese Systeme besitzen, welche Art von Informationen sie verwalten und wie das zugrundeliegende konzeptuelle Modell ihrer Informationsartefakte aufgebaut ist, um herauszufinden, inwieweit sie jeweils Beiträge für Wissensmanagementsysteme liefern. Abschließend wird ein integriertes Szenario für Wissensportale formuliert, das die Beiträge der zuvor untersuchten Systeme aufnimmt. Dieselben Untersuchungen werden in Abschnitt 3.4 anhand dreier kommerzieller Portalsysteme geführt, die explizit als Wissensmanagementsysteme positioniert sind. Hier stehen also wieder die Funktionalität und insbesondere die verwendeten Modelle im Vordergrund des Interesses. Abschließend werden in Abschnitt 3.5 die Erkenntnisse zusammengefaßt und die Kernthese der Arbeit formuliert, daß internetbasierte Wissensmanagementsysteme bislang sowohl eine Abdeckung der Funktionalität bieten, die geringer als benötigt ist, als auch integrierte und problemadäquate Modelle für die Informationsartefakte vermissen lassen.

3.1 Historische Entwicklung der Informationssysteme

Bereits in der Einleitung (Abschnitt 1.1) wurde die Entwicklung des Internets und der dort zum Einsatz kommenden Technik als neuer Impulsgeber für das Wissensmanagement angerissen. Die Unternehmensstrukturen und die Benutzerkreise, Aufgaben und Arbeitsweisen der betrieblichen Informationssysteme haben sich in den letzten 30 Jahren stark verändert. Zusammengefaßt sind die Änderungen in der Tabelle 3.1 nach [Pou97, MW00]. Es lassen sich vier Phasen voneinander abgrenzen:

1. Die Phase der Großrechner gestattete nur eingeschränkten Benutzerkreisen Zugang zu Informationssystemen, die dann hauptsächlich für den Hintergrundbetrieb ausgelegt waren. Als Haupt-Anwendungen kamen Lohn- und Finanzbuchhaltung zum Einsatz.
2. Mit dem Erscheinen des *Personal Computer* (PC) kamen erstmals größere Benutzerkreise in Kontakt mit Anwendungen; Büroautomatisierung war die Hauptaufgabe.
3. Auf der Grundlage des Client/Server-Paradigmas wurden die bisherigen isolierten Anwendungen vernetzt und allen Mitarbeitern Zugriff von ihrem PC auf die zentralen Systeme gewährt, um Geschäftsprozesse möglichst gut zu unterstützen.
4. Mit der Entwicklung des Internets zum Massenmedium hielten dessen Prinzipien und Techniken auch in den Unternehmen Einzug. Insbesondere Kundenorientierung und konstanter Wandel bestimmen die Unternehmensstrukturen. Charakteristisch für die Aufgaben und Arbeitsweise der Informationstechnik ist die unternehmensübergreifende Integration von Systemen. Dabei ist die hohe Lebensdauer der gespeicherten Informationen zu berücksichtigen (oft über 10 Jahre und mehr) und eine daraus resultierende heterogene Systemlandschaft. Das Paradigma des *Network-Computing* wurde damit geschaffen [Att99].

Der Übergang von der dritten zur vierten Phase bestimmt gegenwärtig die Systemlandschaft (auch) im Bereich des Wissensmanagements. Eine große Anzahl von klassischen Client/Server-orientierten Systemen, beispielsweise betriebliche Informationssysteme, aber auch Dokumentenmanagement-Systeme werden noch eingesetzt (und werden auch noch lange Zeit eingesetzt werden), während gleichzeitig alle Arten von bestehenden Systemen einerseits web-fähig gemacht werden und sich andererseits parallel zu den bestehenden neue, innovative Systeme wie Content-Management-Systeme und viele mehr oder weniger dedizierte Wissensmanagement-Systeme in Form von Portalen oder zumindest Intranet-basierten Web-Anwendungen etablieren.

Ära	Organisation	Hauptaufgabe der IT	Arbeitsweise der IT	Benutzerkreis
1970 ... Mainframe	zentralisiert, hierarchisch, statisch	Back-Office-Unterstützung (Lohn- & Finanzbuchhaltung, Kunden-Datenbank)	serielle Hintergrund-verarbeitung, zentrale Datenhaltung und -zugriff	eingeschränkt: Systemadministrator, Datentypisten
1980 ... Personal Computer	strategische Geschäftseinheiten werden in Funktionen organisiert, Strukturen werden verschmolzen	Automatisierung der Büroarbeit, Entscheidungs-unterstützung	komplexe, interaktive Anwendungen, einzelstehende PC-Anwendungen	ausgeweitet: Sekretariat, Management, ...
1990 ... Client/Server	Übergang zu flexiblen flachen, sich selbst organisierenden Strukturen, basierend auf Geschäftsprozessen	Unterstützung aller Geschäftsprozesse, technischer und strategischer Entscheidungen, fundamentale Rolle digitalisierter Dokumente	verteilte Verarbeitung, Integration der PC-Anwendungen	unternehmensweit: einfache Angestellte bis Manager, überall und jederzeit, besonders mobil und außerhalb der Firma
2000 ... Internet / Intranet	kundenorientiert, vernetzt, in konstantem Wandel befindlich (Übernahmen, Ausgründungen, Outsourcing)	Front-Office-Unterstützung, Überwindung von Medienbrüchen und Systemgrenzen	generische, dokumentenzentrierte Klienten, verteilte, spezialisierte Server, über Softwarebrücken miteinander kooperierend	unternehmens-übergreifend: Kunden, Makler, Subunternehmer, Zulieferer

Tabelle 3.1: Wandel von Unternehmensstrukturen und Informationstechnik

Das oben angesprochene Paradigma des *Network-Computing* verlangt nach teilweise unternehmensübergreifenden Lösungen, während gleichzeitig innerhalb von Organisationen erst (dazu) Medienbrüche und Systemgrenzen überwunden werden müssen. Dies ist zugleich die typische Situation, vor der Projekte zur Einführung von Wissensmanagementsystemen stehen. Als wichtigste Grundlage für moderne Systeme zum Wissensmanagement muß also die Nutzung von Internet-Techniken und ein hoch integrativer Charakter gefordert werden.

3.2 Klassifikation von Wissensinfrastrukturen

Nach diesem historischen Überblick werden nun verschiedene Klassen von Informationssystemen identifiziert, die im Kontext des Wissensmanagements eine Rolle spielen. Unglücklicherweise herrscht heutzutage große Unsicherheit darüber, welche Arten von Systemen als reine Wissensmanagement-Systeme anzusehen sind, ob es solche überhaupt gibt oder ob es sich letztlich nur um Weiterentwicklungen bestehender Informationssysteme mit neuer Etikettierung handelt. Analysiert man etwa die in [Gen99, BVÖ99] aufgeführten Arten von Informationssystemen, so finden sich sogar fast ausschließlich altbekannte wieder, die früher nicht unbedingt als WMS firmierten. Dies macht die Schwierigkeit der systematische Beschreibung von Wissensmanage-

ment-Systemen recht deutlich. Rekapituliert man die Erkenntnisse des letzten Kapitels 2 vor dem Hintergrund der dort geführten Diskussion über die Definition der Begriffe Daten, Information und Wissen, so wird eines klar: Die Begriffsbildung ist deshalb so uneinheitlich, weil grundsätzlich alle Informationssysteme – die ja unzweifelhaft mit Daten und Informationen umgehen – von Menschen benutzt werden und so in bestimmter Art und Weise zur Wissensverteilung oder -nutzung verwendet werden, also mit einer gewissen Berechtigung als Wissensmanagement-System betrachtet werden können.

Wenn heute Wissensmanagement betrieben wird, so werden auf der systemtechnischen Seite eine Vielzahl von Informationssystemen eingesetzt. Je nach spezieller Aufgabe und Ausprägung der Ziele werden unterschiedliche Systeme in verschiedenen Zusammenstellungen verwendet. Ein „Standard"-Wissensmanagement-System oder eine Standardkonfiguration von Teilsystemen ist nicht auszumachen. Eine Reihe von Studien haben allerdings im Rahmen von Befragungen u.a. Untersuchungen darüber angestellt, welche Technologien für das Wissensmanagement zum Einsatz kommen bzw. welche geplant seien [ADI+00, BWP97, Noh00c, HV98]. Die Anzahl der befragten Unternehmen ([ADI+00] mit 143, [BWP97] mit 311, [Noh00c] mit 22), die Region und die Systematik (Art der Fragen) variieren dabei relativ stark. Die Ergebnisse aller vier Studien sind in Tabelle 3.2 zusammengefaßt; die Nennungen der Systeme bzw. Technologien sind in der Reihenfolge ihrer Häufigkeiten gemäß der jeweiligen Studie aufgeführt. Auf die Angabe der relativen oder absoluten Häufigkeit wurde der Übersichtlichkeit halber und der schwer vergleichbaren Methodik wegen verzichtet; auch waren besonders auffällige Spitzenreiter oder Ausreißer nicht zu verzeichnen und die Häufigkeit fällt bei allen recht gleichmäßig ab.

[ADI+00]	[BWP97]	[Noh00c]	[HV98]
• Internet • Intranet • DMS • Groupware • Suchmaschinen • Diskussionsforen • WFM • Portale • Data Warehouses • andere • Push-Technologie	• Telefon • Kunden-DB • Telefax • Erfahrungs-DB • e-Mail • Intranet • DMS • Engineering-DB • Expertensysteme	• Intranet • Datenbanken • DMS • Groupware • Textretrieval • Data Warehouses • Mind-Mapping • WFM	• Wissens-DB • Intranet, DB-Zugriff • Expertensysteme, CBR • intelligente Agenten

Tabelle 3.2: Technologien des Wissensmanagements

Es läßt sich aus den Studien ersehen, daß Intranet- bzw. Internet-Technologien zusammen mit Dokumentenmanagement-Systemen (DMS) und *Groupware* (worunter auch e-Mail fallen dürfte, vgl. Abschnitt 3.2.2) sowie teilweise recht unspezifische Datenbank-Anwendungen klar dominieren. Eine genauere Reihenfolge läßt sich aus diesen Daten allerdings nicht unbedingt ableiten. Interessant ist die Dominanz der klassischen Werkzeuge Telefon und Telefax in [BWP97], die evtl. auf die Art der Fragestellung (Bedeutung der Technologien für das WM) zurückzuführen ist. Dennoch macht dies die Wichtigkeit von persönlicher Kommunikation für den Wissenstransfer deutlich. Wie in [ADI+00] beobachtet wird, ist die geringe Resonanz auf Portale und *push*-Technologie bemerkenswert; die Gründe werden jedoch darin gesehen, daß einerseits die Einführung häufig erst geplant ist und daß sich andererseits eine gemeinsame Terminologie noch nicht überall durchgesetzt hat. Das zeigt, daß deutlicher Untersuchungsbedarf bezüglich Terminologie, Konzepten und Modellen besteht. Überraschend ist, daß die Ergebnisse großer Gebiete der wissenschaftlichen Forschung, nämlich Expertensysteme, künstliche Intelligenz und Agen-

tensysteme, nur die letzten Plätze belegen. Dies könnte man so interpretieren, daß innovative Entwicklungen nur zögerlich in der Praxis Verwendung finden und eher Forschungsprototypen bleiben (siehe zu dieser Thematik [AMB01]). Grundsätzlich problematisch bei allen Untersuchungen ist, daß keine genaue Begriffsbestimmung der Technologien stattfindet. Hier wären präzisere Abgrenzungen, z.B. von Intra- und Internet gegenüber Portalen, bei der Analyse sicherlich hilfreich.

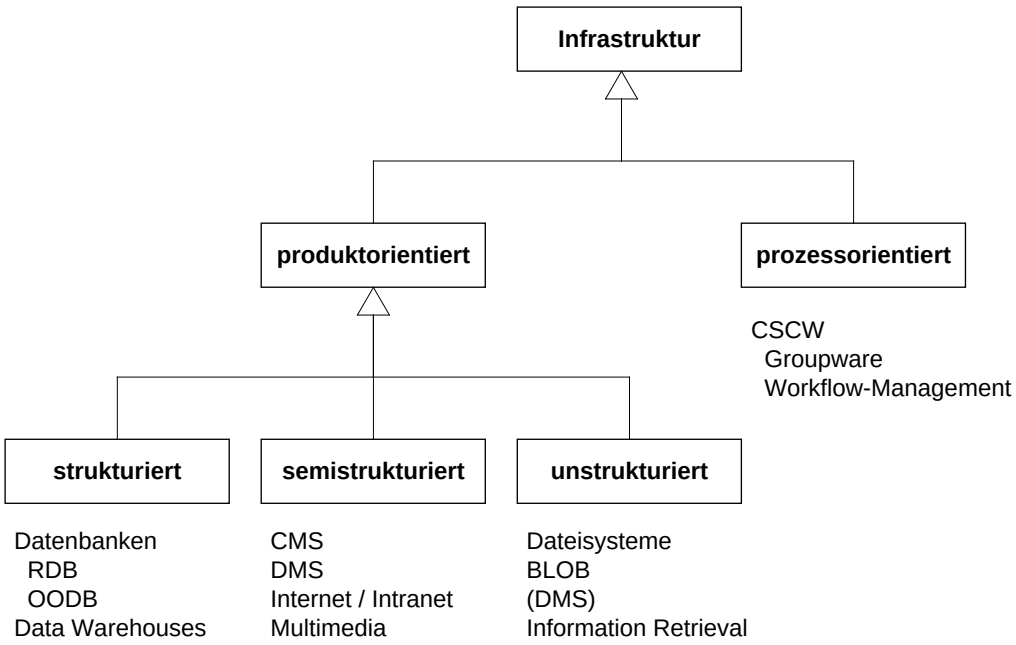

Abbildung 3.1: Klassifizierung von Wissensinfrastrukturen

Nachdem die besonders häufig eingesetzten Technologien und Systeme identifiziert wurden, werden diese nun zunächst genauer klassifiziert, um die Arten der verwalteten Informationen und der unterstützten Prozesse besser gegeneinander abgrenzen zu können. Eine erste grobe Klassifikation der benutzten Systeme ist durch die Perspektive entweder aus der *produktorientierten* Sicht oder aus der *prozessorientierte* Sicht [SAD99, ADK98] möglich. Es läßt sich so zwischen produktorientierten Informationssystemen einerseits und prozessorientierten andererseits unterscheiden (siehe Abbildung 3.1). Die produktorientierten Systeme lassen sich weiterhin anhand des Strukturierungsgrades der abgelegten Informationen[1] unterteilen, und die prozessorientierten Systeme anhand der Strukturierungsstärke der modellierten Prozesse (in der Abbildung nicht dargestellt).

3.2.1 Produktorientierte Informationssysteme

Im Mittelpunkt der Modelle, derer sich die produktorientierten Informationssystemen bedienen, stehen die verwalteten Produkte, also Informationen. Innerhalb der produktorientierten Informationssysteme läßt sich genau genommen zwischen *strukturierten*, *semistrukturierten* und *unstrukturierten* Informationen (bzw. Modellen) differenzieren, die von den jeweiligen Systemen verwaltet werden können.

[1] Statt von Informationen wird in diesem Zusammenhang häufig von Daten gesprochen; wir folgen aber den Definitionen aus Abschnitt 2.2.

Strukturierte Informationen: Zu den Informationssystemen, die mit strukturierten Informationen umgehen, gehören natürlich in erster Linie die klassischen Datenbanksysteme [LS87]. Dazu zählen diejenigen Systeme, die den historischen Datenbankmodellen wie dem Netzwerk-Datenbankmodell und dem hierarchischen Modell folgen, und die immer noch einen beachtlichen Anteil an den produktiv im Einsatz befindlichen Datenbanksystemen stellen, insbesondere bei Banken und Versicherungen. Bei beiden ist die Modellierungsmächtigkeit in wichtigen Punkten beschränkt [SM98].

Weiterhin sind als die beiden wichtigsten Vertreter der Datenbanken die relationalen Datenbankmodelle (RDBM) [Dat95, Cod70] sowie die neueren objektorientierten Datenbankmodelle (OODBM) [Cat00, Cat97, Cat94, Heu92, SRL+90] zu nennen. Beide unterstützen die starke Strukturierung von Informationen in hohem Maße durch große Modellierungsmächtigkeit. Die jüngste Entwicklung ist eine Art Synthese beider Modelle durch Erweiterung relationaler Datenbankkonzepte um objektorientierte in den Objekt-Relationalen Datenbankmodellen (ORDBM). Durch das Anbieten von mächtigeren Abstraktionsmechanismen soll der sogenannte *impedance mismatch* zwischen Anwendungssprachen und Datenbankmodell verringert werden [Luf99, Sto96, Sku98, SM98, Heu92, Cat94].

Die Rolle der Datenbanken im Wissensmanagement liegt in der Bereitstellung von speziellen, strukturierten Informationen, die etwa Unternehmenszahlen wie Umsätze oder Personal und viele weitere Unternehmensdaten (Kunden, Adressen, Mitarbeiter, Aufträge, Lager) betreffen. Der direkte Umgang mit Datenbanken gehört allerdings eher zur Ausnahme; in der Regel werden Datenbanksysteme dazu verwendet, innerhalb einer Mehrschichten-Architektur die Grundlage für weitere Anwendungssysteme zu bilden. Grundsätzlich sind diese Anwendungssysteme dafür ausgelegt, mit strukturierten Informationen umzugehen.

Eine weitere Klasse von Informationssystemen, die mit stark strukturierten Daten umgehen, sind die der analytischen Systeme wie *Data Warehouses* [Glu97, PN00], die auch als *Management Information Systems* (MIS) bezeichnet werden. Sie haben die Aufgabe, entscheidungsrelevante Informationen zu erzeugen und zu speichern. Sie erhalten ihre Daten zumeist aus den zuvor genannten operativen Systemen wie Auftrags- und Bestellsystemen etc. und werten diese dann ggf. zeitnah (*OnLine Analytical Processing*, OLAP) aus. Diese Systeme haben natürlich eine hohe Bedeutung für das Wissensmanagement, da sie äußerst konzentrierte Informationen bereitstellen, die dicht an der Generierung neuen Wissens angesiedelt sind.

Semistrukturierte Informationen: Für den Begriff der semistrukturierten Daten bzw. Informationen existiert heute noch keine ganz einheitliche und anerkannte Definition, allerdings hat sich praktisch die Sichtweise von Objektsammlungen mit heterogener Struktur durchgesetzt, die im Gegensatz zu der homogenen Struktur der durch die etablierten Datenbankmodelle (s.o.) geprägten Sichtweise steht. Weitere Merkmale sind tiefe, geschachtelte Strukturen, die unregelmäßig auftreten können und Auslassungen und Wiederholungen erfahren. Oft ist die Struktur durch die Daten selbst erklärt und besitzt keine externe, explizite Definition [Bun97, BKOS01]. Dieser pragmatische Ansatz ist stark durch die Sicht der Datenbeschreibungssprache XML (*Extensible Markup Language*) geprägt [Far99, WWW98a]; mitunter scheinen die Begriffe semistrukturierte Daten und XML gleichbedeutend verwendet zu werden.

Neben den durch XML beschriebenen oder beschreibbaren Informationen existieren zahlreiche weitere Arten von Informationen, die eine nur schwach ausgeprägte Struktur aufweisen, etwa Textdateien, e-Mails und HTML-Seiten. Viele weisen zwar eine gewisse Binnenstruktur auf, die aber beispielsweise bei HTML wegen der fehlenden Trennung von Inhalt und Layout häufig nicht semantisch relevant ist. Abhilfe gegen die Vermischung von Inhalt und Layout verspricht erst XML.

Informationssysteme, die mit semistrukturierten Daten umgehen, sind zum Beispiel Dokumentenmanagement-Systeme und insbesondere moderne Contentmanagement-Systeme. Beiden gemeinsam ist das Dokument als zentrale Abstraktion, dessen Aufbau nicht oder nur zum Teil formal strukturiert sein kann. Eine genauere Untersuchung der Dokumentenmodelle findet in Abschnitt 3.3 statt.

Genauso lassen sich z.T. die Anwendungsserver (*Application-Server*), die für das Betreiben von Internet- oder Intranet-Systemen verwendet werden, als Informationssysteme sehen, die mit semistrukturiertem Inhalt umgehen. Schließlich fallen alle Arten von Multimedia-Informationssystemen (etwa Bilddatenbanken, *Streaming-Server* etc.) in diese Klasse. Der Begriff des *Media Asset Managements* fällt öfters [Neu01] im Zusammenhang mit Multimedia-Datenbanken und ist in enger Beziehung zum Dokumenten- und Content-Management zu sehen.

unstrukturierte Informationen: Die Mehrzahl der heute noch in der Praxis verwendeten Dokumentenformate (beispielsweise *Office*-Dokumente von Textverarbeitungen und Tabellenkalkulationen, Bilder, Multimediadateien) weist eine monolithische, dem Benutzer und anderen als den zur Erstellung verwendeten Werkzeugen unzugängliche – weil unbekannte – Struktur auf, so daß sie praktisch als unstrukturiert betrachtet werden müssen. Allenfalls existieren einige wenige Metadaten (eigentlich Meta-Informationen), die Aufschluß über Titel, Autor, Erstellungsdatum etc. liefern.

Unstrukturierte Informationen dieser Art werden üblicherweise in Dateisystemen abgelegt. Abhängig vom Grad der Abstraktion des Dokumentenmodells werden unstrukturierte Informationen auch von Dokumenten-Management-Systemen sowie Content-Management-Systemen verwaltet, wobei sie oft als Typ „unstrukturierte Daten“ innerhalb stark oder semistrukturierter Dokumentenmodelle auftreten. Die neueren Datenbanken besitzen häufig Datentypen, die große binäre Objekte ohne (bekannte) Struktur aufnehmen können, die sogenannten *Binary Large OBjects* (BLOBs).

Eine typische Anwendung, die sich mit unstrukturierten Informationen beschäftigt, ist die Volltextsuche im Rahmen des *Information Retrieval*. Hier wird der Versuch gemacht, aus den proprietären, monolithischen Dateiformaten zumindest den reinen Text wieder zu extrahieren, und wenn möglich, auch weitergehende Metainformationen und Strukturinformationen.

3.2.2 Prozeßorientierte Informationssysteme

Im Unterschied zur produktorientierten Sichtweise stehen hier Zusammenarbeit (Kooperation, Koordination, Kommunikation) und arbeitsteilige Prozesse im Vordergrund. Gerade die Wissensverteilung (vgl. Abschnitt 2.4.5) kann sinnvoll durch prozessorientierte Informationssysteme unterstützt werden. Das interdisziplinäre Gebiet des *Computer Supported Cooperative Work* (CSCW) beschäftigt sich schon lange – unabhängig vom Wissensmanagement – mit Fragen der kooperativen Arbeit von Menschen und den Möglichkeiten der technischen Unterstützung durch Informationssysteme.

Diese Unterstützung läßt sich einerseits in den zwei Dimensionen Raum und Zeit klassifizieren, wobei jeweils unterschieden wird, ob die Teilnehmer der Kooperation sich am gleichen Ort befinden oder nicht, und ob die Kooperation synchron (also zur gleichen Zeit) oder asynchron (also nicht gleichzeitig) stattfindet [Eng90]. Die Matrix in Tabelle 3.3 zeigt die Dimensionen und welche Systeme typischerweise für welche Arten der Kooperation eingesetzt werden. Andererseits kann der Strukturierungsgrad der Prozesse zur Klassifizierung herangezogen werden. Zwei Arten von Systemen werden üblicherweise im Rahmen des CSCW unterschieden:

	gleiche Zeit	unterschiedliche Zeit
gleicher Ort	• Entscheidungsunterstützung • Computerunterstützte Sitzung • Präsentationssoftware	• Terminkalender • Projektmanagement • Workflow-Systeme
unterschiedlicher Ort	• Telefon • Audio / Videokonferenzen • Chat / Instant Messages • elektronisches Whiteboard	• FAX • e-Mail • Diskussionsforen • File-Sharing • Versionsmanagement

Tabelle 3.3: Arten der Unterstützung durch CSCW

Sogenannte *Groupware* und Workflow-Management-Systeme (WFMS). Sie unterscheiden sich hauptsächlich in der Modellierung bzw. Unterstützung der Strukturierungsstärke der Prozesse: Während *Groupware*-Systeme weniger Wert auf streng definierte Abläufe, die Einhaltung von Geschäftsprozessen legen und so eher ad-hoc-Vorgehen und lose Teamarbeit unterstützen, liegt der Schwerpunkt von WFMS eher auf der anderen Seite der starren Prozesse mit wenigen Ausnahmen. In dem Spannungsfeld dieser Extreme sind real existierende Systeme angesiedelt. Die Dimensionen Raum und Zeit finden sich in dieser Einteilung nicht unbedingt wieder.

Groupware-Systeme erlauben es den Benutzern, auf elektronischem Wege Dokumente und Informationen auszutauschen, um die Gruppenarbeit produktiver und effizienter zu gestalten. Wesentliche Bestandteile sind *Messaging*-Dienste, also der Austausch von Nachrichten (e-Mail) und die Teilnahme an Diskussionsforen, Terminkalender für Mitarbeiter und Gruppen, die gemeinsam gepflegt und eingesehen werden können, häufig Verwaltung von Aufgabenlisten sowie Erstellen, zentrale Ablage und Austausch von Dokumenten. Eine gute Definition für *Groupware* ist die folgende an [FPU98] angelehnte:

> „*Groupware* sind Software-Systeme, die Arbeitsgruppen bei der Lösung von wenig strukturierten Aufgabenstellungen unterstützen. Die Basis für diese Unterstützung sind drei Hauptfunktionen:
>
> **Kommunikation:** Die Übermittlung von Nachrichten in elektronischen Formaten.
>
> **Kooperation:** Für die Arbeitsgruppe steht ein gemeinsam genutzter, virtueller Arbeitsplatz zur Verfügung.
>
> **Koordination:** Es existieren Möglichkeiten, die realen Abläufe auf den virtuellen Arbeitsplatz abzubilden.“

Eine systematische Klassifikation von Werkzeugen im Spannungsfeld zwischen Kommunikation, Kooperation und Koordination bietet die in [BVÖ99, Gen99] gefundene Einordnung (siehe Abbildung 3.2). Sie deckt viele Systeme ab, die als Werkzeuge zur Unterstützung von Wissensmanagement gelten. Zusätzlich zu der ursprünglichen Darstellung wurden die Systemklassen der Dokumentenmanagement-Systeme (DMS) und Content-Management-Systeme (CMS) mit eingeordnet.

Eine der populärsten Plattformen für Dienste dieser Art bildet LOTUS NOTES [Car99, Lot98], auf dessen Grundlage zahlreiche *Groupware*-Anwendungen basieren. Besonders erfolgreich ist das

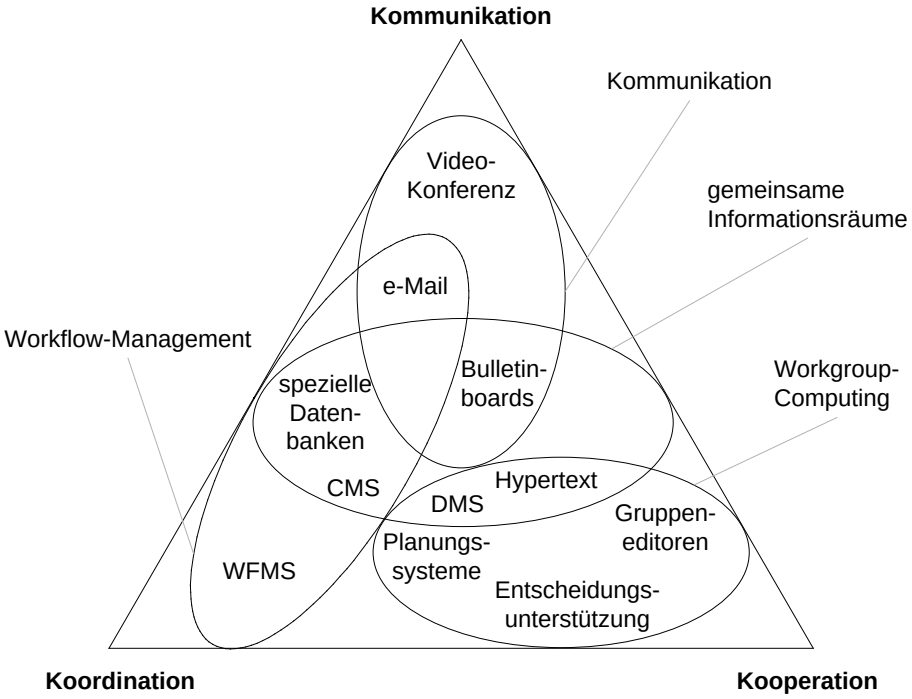

Abbildung 3.2: Kommunikation, Kooperation und Koordination

Konzept von LOTUS NOTES wegen seines dokumentenorientierten Konzepts in Verbindung mit ausgeprägten Fähigkeiten zur feingranularen Verteilung und Replikation der Dokumentenbestände.

Die Intention von Workflow-Management-Systemen ist es, die sich mittlerweile immer schneller ändernden Geschäftsprozesse technisch adäquat zu unterstützen; insbesondere hohe Flexibilität bei der Änderung von Prozeßbeschreibungen ist hierbei wichtig [Zie98]. Hilfreich ist die Definition von Workflow, die [JBS97] entnommen ist:

> „Ein Workflow (WF) ist eine zum Teil automatisiert (algorithmisch) von einem Workflow-Management-System (WFMS) gesteuert ablaufende Gesamtheit von Aktivitäten, die sich auf Teile eines Geschäftsprozesses oder andere organisationelle Vorgänge beziehen. Ein Workflow besteht aus Abschnitten (Subworkflows), die weiter zerlegt werden können. Workflows haben einen definierten Anfang, einen organisierten Ablauf und ein definiertes Ende."

Die Rolle von WFMS innerhalb des Wissensmanagements liegt einerseits darin, daß das *Prozeßwissen*, also das Wissen darüber, welche Geschäftsprozesse wie zu erledigen sind, seinerseits ebenfalls wichtiges Wissen darstellt, das entwickelt, verteilt und genutzt werden muß, andererseits aber umgekehrt durch Workflow-Management-Systeme Wissensprozesse wie die Wissensverteilung oder Wissensnutzung unterstützt oder durchgeführt werden können.

3.2.3 Bewertung

Die vorgenommene Unterscheidung zwischen strukturierten, semistrukturierten und unstrukturierten Informationen eröffnet bereits eine wichtige Klassifikationsdimension für Informationssysteme. In diesem Zusammenhang ist die häufig vorgebrachte Einschätzung interessant, daß in Organisationen lediglich 20 bis 30 Prozent der Informationen in strukturierter Form vorlägen, der Rest von 70 bis 80 Prozent hingegen in unstrukturierter bzw. semistrukturierter Form [RR01, Dit00, Noh00a]. Dies spiegelt sich in der Nutzung der Wissensinfrastruktur wider und erklärt den hohen Stellenwert von Internet, Intranet und Dokumentenmanagementsystemen in den Untersuchungen (vgl. Anfang von Abschnitt 3.2). Diese Klassen von Systemen sind offenbar dafür prädestiniert, tendenziell unstrukturierte und semistrukturierte Informationen zu verwalten, geeignet ausgebaut aber auch strukturierte Informationen mit einzubeziehen.

Es zeigt sich hier bereits, daß die konkret zur Verfügung stehenden Werkzeuge und Systeme jeweils nur Teilbereiche der Anforderungen abdecken. Sowohl die produktorientierten System sind in der Regel auf die Handhabung genau einer Art von Information (strukturiert, unstrukturiert oder semistrukturiert) spezialisiert, als auch die prozessorientierten, die jeweils unterschiedliche Schwerpunkte (Kommunikation, Koordination oder Kooperation) setzen. Systeme, die eine größere Abdeckung der jeweiligen Bereiche versprechen, sind nicht auszumachen.

Als zentrale Anforderung an integrierte Wissensmanagement-Systeme ergibt sich also die gleichzeitige Benutzung vorhandener strukturierter und unstrukturierter Informationen neben einer ebenso breiten Abdeckung der prozeßorientierten Aspekte. Weil diese Anforderungen besonders wichtig für das Wissensmanagement sind, werden Dokumentenmanagement-Systeme und die damit eng verwandten Systemklassen der Content-Management-Systeme sowie Unternehmens-Portale im nächsten Abschnitt eingehend diskutiert, um zu untersuchen, inwieweit sie Ansätze für eine Integration aller Aspekte aufweisen.

3.3 Klassifikation relevanter Systeme

Nach der Diskussion verschiedener grundlegender Klassen von Informationssystemen von der strukturellen Seite her werden nun drei relevante Klassen von Informationssystemen von der funktionellen Seite und der der Modelle her untersucht und voneinander abgegrenzt. Relevante Systeme sind:

- Dokumenten-Management-Systeme (DMS), weil diese Systeme einerseits eine traditionell starke Rolle im Wissensmanagement zur Verwaltung von expliziten Informationen spielen (vgl. Abschnitt 3.2) und andererseits z.T. als historische Vorgänger der Content-Management-Systeme viele Gemeinsamkeiten mit diesen aufweisen, die es ebenfalls zu untersuchen gilt;
- Content-Management-Systeme (CMS), weil diese Systeme normalerweise zum Betrieb von Web-Portalen erforderlich sind und wie eben erwähnt z.T. aus den Dokumentenmanagement-Systemen hervorgegangen sind. Sie weisen ferner bereits interessante Eigenschaften zur Abwicklung von Prozessen auf, die wichtig für das Wissensmanagement sind. Weiterhin werden CMS mittlerweile häufig explizit als Wissensmanagement-Werkzeug verstanden. Zudem gibt es bisher nur wenige systematische Ansätze zur Beschreibung und Abgrenzung der CMS.
- Unternehmens-Portale (EIP) bzw. Wissensportale, weil diese Systeme neuerdings öfters für das Wissensmanagement verwendet werden und das Kernthema dieser Arbeit sind, aber

die bestehenden Ansätze zur Beschreibung noch äußerst spärlich sind. Hier ist insbesondere zu klären, welche Arten von Portalen existieren und wo die Abgrenzungen untereinander verlaufen.

In den folgenden Abschnitten 3.3.1 bis 3.3.3 werden für jede dieser drei Klassen die historische Entwicklung beleuchtet, Definitionen gegeben, ihre wichtigsten und charakteristischen Funktionen identifiziert, diese Funktionen von den anderen Klassen abgegrenzt und einige Beispiele von kommerziellen Anwendungssystemen genannt. Der Schwerpunkt dieser Untersuchung ist aber, welche Art von Informationen im Sinne der Unterscheidungen des Abschnitts 3.2 sie jeweils benutzen und wie das zugrundeliegende konzeptuelle Modell ihrer Informationsartefakte aufgebaut ist. Insgesamt ergibt sich dann daraus ein Bild, das beschreibt, welche Aufgaben und Prozesse des Wissensmanagements nach Kapitel 2 sie abdecken.

Abschließend wird in Abschnitt 3.3.4 nach einer Diskussion der beiden Begriffe des Wissensmanagement-Systems (WMS) und des *Organizational Memory Information Systems* (OMIS) ein integriertes Szenario zum Verständnis des Zusammenwirkens von Dokumentenmanagement-Systemen, Content-Management-Systemen und Wissensportalen aufgebaut, das die Beiträge der zuvor untersuchten Systeme aufnimmt.

3.3.1 Dokumenten-Management-Systeme

Dokumentenmanagement-Systeme (DMS) existieren schon sehr lange (über 20 Jahre) als eigenständige Systemklasse [BL94]; das Schlagwort vom „papierlosen Büro“ stellt die ursprüngliche Intention dieser Systeme recht gut dar [Pfa95]. Dabei hat sich diese Klasse bisher nur relativ geringer Aufmerksamkeit seitens der Informatik-Forschung erfreut, insbesondere die technischen und softwaretechnischen Fragestellungen nehmen in der Diskussion keinen breiten Raum ein. Der Fokus der meisten Berichte und Untersuchungen liegt eher auf der organisatorischen Einbettung; ein großer Teil der technischen bzw. wissenschaftlichen Betrachtungen konzentriert sich auf Fragen der mit Dokumentenmanagement eng verknüpften automatischen optischen Erkennung der Inhalte von Papierdokumenten und fällt so mit in das Gebiet der Muster- und Bilderkennung. Immerhin spielt das Dokumentenmanagement im betrieblichen Umfeld eine wichtige Rolle und wird von der Betriebswirtschaft bzw. der Wirtschaftsinformatik dergestalt wahrgenommen, daß Dokumentenmanagement ein Schlüsselfaktor ist, um Geschäftsprozesse und die Wertschöpfungsketten, die schon durch Informationstechnik unterstützt werden, ohne weitere Medienbrüche (elektronisches vs. Papierdokument) durchzuführen [Dan99]. Eine gute, hierzu passende Definition von Dokumenten-Management findet sich in [BKM92, S. 15]:

> „Das Dokumenten-Management hat die Aufgabe Dokumente zu archivieren, zu speichern, zu drucken, einzulesen, wiederzufinden und den Menschen bei der Bearbeitung, der Verwaltung, der Weitergabe und Ablage von Dokumenten zu unterstützen. Es hat zum Ziel, die Produktivität durch eine Verkürzung der Dokumentendurchlaufzeit und eine sofortige Bereitstellung notwendiger Informationen zu erhöhen.“

Zudem stellt sich die Frage der Definition des Begriffs Dokument. In Anlehnung an [TB99] kann man als Dokument definieren:

> „Ein Dokument ist jede Art von unstrukturierter Information, die als geschlossene Einheit in einem Informationssystem als Datei vorliegt.“

Diese beiden Definitionen beschreiben also eher eine Fortschreibung der klassischen Arbeitsweise in Organisationen mit klassischen Dokumenten, die jetzt aber lediglich auf einem anderen Medium als Papier verfügbar sind. Im Kontrast dazu steht die Definition aus [IW99c], die Dokument folgendermaßen definiert:

> „Der Begriff ‚Dokument' beschreibt [...] einen Container, der eine beliebige Anzahl von Komponenten enthalten kann: zum Beispiel Text, Bilder, Grafiken, Tabellen, oder *Applets*."

Interessant ist hier zudem die uneinheitliche Sichtweise des Begriffes, die von unstrukturierter Information bis hin zu einer offenbar strukturierten Sammlung von Einzelkomponenten reicht. Eine ähnliche Sichtweise liegt dem dokumentenorientierte Ansatz von LOTUS NOTES zugrunde, der als zentrales Artefakt ebenfalls das explizit so genannte Dokument besitzt, das aus zahlreichen system- und benutzerdefinierten Feldern (Attributen) besteht (siehe [Car99] für eine detaillierte Diskussion der Modellierungskonzepte unter einer relationalen und objektorientierten Sicht). Diese Disparität scheint eine der Hauptursachen für die fehlende begriffliche Transparenz und Abgrenzung der Systemklassen zu sein; tatsächlich führt so gut wie jeder Anbieter mit seinem System eine neue implizite oder explizite Definition ein. Um dieser unbefriedigenden Situation zu entkommen, wollen wir für den Bereich des Dokumentenmanagements die erste Definition zugrunde legen, also von einer unstrukturierten Einheit in Form einer Datei als Dokument ausgehen. Dafür spricht, daß dies die lange gepflegte traditionelle Sichtweise der Mehrzahl der Dokumentenmanagement-Systeme ist, die erst in jüngster Zeit von den gleich noch zu beschreibenden Entwicklungen eingeholt und verändert wurde. Als Vorgriff auf Abschnitt 3.3.2 sei angemerkt, daß dort ein anderer, strukturierterer Dokumentenbegriff als der passendere betrachtet wird, und daß die Anforderung, strukturierte Dokumente zu verwalten, für Wissensmanagement-Systeme besonders relevant ist.

Bei Dokumenten in dem jetzt für Dokumentenmanagement verwendeten Sinne kann es sich also beispielsweise um alle möglichen Arten von Dateien, die von Textverarbeitungen, Tabellenkalkulationen, Präsentationssoftware (Folien) oder anderer Büro-Software erzeugt werden, Bilder oder Textdateien handeln. Dies spiegelt die gängige Praxis und die pragmatischen Kernanforderungen realistischer Alltagsszenarien gut wider.

Aufgrund des in Organisationen ständig wachsenden Umfangs der Datenbestände entstand die Notwendigkeit, Dienste und Werkzeuge zur Verwaltung dieser sonst nicht mehr handhabbaren Mengen von Dokumenten zur Verfügung zu stellen. Die Kernaufgaben solcher Dokumentenmanagementsysteme sind:

- Die zentrale Ablage und Sicherung von Dokumenten. Dazu wird sich in der Regel des Client/Server-Prinzips bedient. Hier sind äußerst verschiedene Arten der Anbindung von Client und Server realisiert worden; die Vielfalt reicht von in bestimmte Anwendungen integrierten Clients über dedizierte Server bis hin zu transparent als Dateisystem erscheinenden Dokumentenspeichern.

- Die Strukturierung und Organisation der abgelegten Dokumente. Dies kann von dem simplen Angebot einer Verzeichnisstruktur bis hin zu multidimensionalen Leitsystemen reichen, in denen die Dokumenten nach Art, Schlagworten, Stichworten, Autoren etc. erschlossen und klassifiziert werden.

- Die Indexierung der abgelegten Dokumente nach Stich- und Schlagworten, Metadaten und Organisation sowie entsprechende Recherchefunktionen dafür. Die Volltextindizierung der

Dokumente wird dabei häufig zusätzlich unterstützt, erfordert allerdings eine große Anzahl für Dokumentenformate spezifische Textfilter, um alle halbwegs gängigen Dokumentenarten erfassen zu können.

- Die Verwaltung von Benutzern und Berechtigungen für die abgelegten und abzulegenden Dokumente. Dabei sind verschiedene Berechtigungskonzepte (etwa dokumentweise, verzeichnisweise etc.) und Operationen möglich.
- Der kontrollierte Zugriff auf die Dokumente durch Versionierung, bei der alle alten Zwischenstände eines Dokuments gespeichert werden, und Sperrmechanismen, die den gleichzeitigen Zugriff auf Dokumente einschränken (*Check in/out*).

Neben diesen als Kernfunktionalität zu bezeichnenden Eigenschaften von Dokumentenmanagement-Systemen gehören häufig weitere Funktionen hinzu:

- Die automatische Erfassung von Papiervorlagen (*Scannen*), etwa der eingehenden Post (Schreiben, Rechnungen, Belege etc.), um diese sofort im elektronischen System weiter zu bearbeiten. Dies wird in der Branche gewöhnlich als *Imaging* bezeichnet [IW00f, Teil 1]. Solche Lösungen erfordern normalerweise beträchtlichen Hardwareaufwand besonders für die Scanner, da der Einsatz erst ab einem gewissen Volumen an täglich anfallender Post lohnt. Zusätzlich zum reinen optischen Erfassen der Vorlagen und Speichern als Bilddokument mehren sich die erfolgreichen Ansätze, per optischer Zeichenerkennung (OCR) die Inhalte[2] der gescannten Dokumente zu erfassen und in der elektronisch wesentlich besser weiterverarbeitbaren Form zu nutzen. Gerade die Auswertung von eher schematischen Vorlagen wie Formularen etwa in Banken und Versicherungen spart sehr viel Arbeit. Die automatische Erfassung von Überweisungsträgern ist ein gutes Beispiel für die bereits erreichte Leistungsfähigkeit [IW00e]. Als konsequente Fortsetzung erfolgt dann die direkte Anbindung an entsprechende betriebliche Software wie beispielsweise an SAP R/3-Systeme [IW00d].
- Die automatische und periodische Sicherung bzw. Archivierung von Dokumenten auf besonders sichere Speichermedien wie optische Datenträger mithilfe von leistungsfähigen Archivsystemen. Neben der natürlich nicht zu vernachlässigenden Aufgabe der Datensicherung, um den Betrieb gegen Ausfälle von Hardware etc. zu sichern, ist die Motivation hier die der *revisionssicheren* Ablage von Dokumenten [Pfa95]. Der Hintergrund ist die rechtliche Lage, die Unternehmen für bestimmte Bereiche zwingt, Dokumente in rechtssicherer, beweiskräftiger Form langjährig aufzubewahren: So müssen z.B. in der Pharma-Industrie zur Zulassung neuer Medikamente alle klinischen Vorstudien und chemischen Analysen lückenlos dokumentiert und nachgewiesen werden; dasselbe gilt für die Produkthaftung in der Fertigungsindustrie, die im Streitfall etwa revisionssichere Konstruktionsbeschreibungen beiliefern können muß [IW99b]. Dabei ist besonders die Manipulierbarkeit der Medien entscheidend, so daß zur Zeit oft genau einmal beschreibbare Medien wie CD-ROM verwendet werden; ein Stichwort hierzu ist *Computer Output on Laserdisk* (COLD). Dennoch bleibt bis heute der elektronische Status dem Papierdokument nicht gleichgestellt [IW00f].
- Die Unterstützung der reinen Ablage durch Workflow-Funktionen (bzw. des Workflows durch die Dokumenten-Dienste) ist eine oft genutzte, sinnvolle Erweiterungsmöglichkeit [BM93, IW99b]. Dies gestattet die gemeinsame, arbeitsteilige Bearbeitung von Dokumenten in einer Reihe von Prozeßschritten. Die Anbindung von *Groupware*-Lösungen ist eine gängige Erweiterung. In sehr großen Unternehmen (Versicherungen, Banken) beruht häufig die gesamte Vorgangsbearbeitung auf der Integration von optischer Erfassung, elektronischem Zugriff, Workflows, Ablage und Archivierung.

[2] also die Information in der eingeführten Terminologie.

Die über lange Zeit geschehene Entwicklung des Dokumentenmanagements hat zu ausgereiften Systemen geführt. Für den zentralistischen Ansatz wurden unterschiedlichste Arten der Anbindung von Client und Server realisiert, die von speziellen Anwendungen bis hin zu transparent integrierten Sichten auf den Dokumentenspeicher reichen. Zugleich wurden eine Reihe von Standards erarbeitet, die verschiedene System interoperabel machen sollen bzw. offene Schnittstellen für Anwendungen bereitstellen. Dazu zählen insbesondere das ODMA (*Open Document Management API*), das von recht vielen Anbietern unterstützt wird [ODM97], sowie in die letzter Zeit aufgekommenen Schnittstellen WEBDAV [GWF+99, Web01a] und MICROSOFTs WEBPOST API. Erstere basiert auf einer Erweiterung von HTTP, die zweite stellt eine neue Programmier-Schnittstelle unter dem Betriebssystem WINDOWS dar.

In jüngster Zeit sind eine Reihe von Entwicklungen im Dokumentenmanagement zu beobachten [IW99a, IW00a, IW00b], die den Beginn einer starken Umorientierung andeuten, und die durch die rasante Entwicklung des Internet als treibende Kraft zumindest mit ausgelöst wurden. Die Entwicklungen lassen sich in zwei Bereiche aufteilen: einen, der die Positionierung der Systeme insgesamt betrifft, und einen, der sich auf die Funktionalität beschränkt.

1. Die großen Trends bezüglich der Positionierung von Dokumentenmanagementsystemen insgesamt im Markt der Informationssysteme sind, daß deren Anbieter mit ihren Produkten zunehmend in das Gebiet des elektronischen Handels, der betrieblichen Standardsoftware sowie das sich besonders rasch entwickelnde Gebiet des Content-Management eindringen [IW00a].

   - Am naheliegendsten ist die Ausweitung des Dokumentenmanagements in Richtung Content-Management, da hier zunächst die gleiche Basisfunktionalität (Ablage, Zugriff, Sicherheit, Versionierung etc.) erforderlich ist. Der Ausbau um die typischen Web-Content-Management-Funktionen stellt demgegenüber einen nur relativ geringen Schritt dar. Welche das im einzelnen sind und wo die Abgrenzung verläuft, wird im nächsten Abschnitt (Abschnitt 3.3.2) noch genauer diskutiert werden. Jedenfalls drängen einige klassische DMS-Anbieter wie DOCUMENTUM oder FILENET in letzter Zeit nachhaltig in den CMS-Markt [IW00b].
   - Nahtlos an Content-Management sowie an Dokumentenmanagement knüpft ferner die Unterstützung des elektronischen Handels (*e-Commerce*) an. Zum Betrieb von elektronischen Einkaufsmöglichkeiten (*Shops*) ist die Verwaltung einer großen Menge von Artikeln, Artikelgruppen etc. nötig, die sich hervorragend als Dokumente in Dokumentenmanagement- oder Content-Management-Systemen abbilden lassen. Wieder ist von da aus der Schritt in Richtung *Shop* o.ä. nur relativ klein, da prinzipiell nur die eigentlichen zur Abwicklung des Bestellens und Bezahlens nötigen Dienste ergänzt werden müssen.
   - Unmittelbar an *e-Commerce* schließt sich die Verbindung zu betrieblicher Standardsoftware wie beispielsweise SAP R/3 an [MZ98, Wit00, Zie97]. Kaum ein Anbieter, der bereits vor Einführung von internet-basierten Verkaufsmöglichkeiten ein konventionelles Buchungs- und Abrechnungssystem oder ein Warenwirtschaftssystem verwendet hat, wird danach ein Parallelsystem in einem *Shop* betreiben, so daß die Anbindung der *e-Commerce*-Funktionalität an bestehende ERP-Lösungen (*Enterprise Ressource Planning*) meist die zwingende Konsequenz ist. Auch hier ist der Schritt wiederum relativ kurz, da die meisten ERP-Lösungen über gut ausgebaute und dokumentierte Schnittstellen verfügen, z.B. SAP R/3 über die BAPI (*Business Application Programming Interface*) [BEG97, LL99, Car99, Lut97, SAP98].
   - Die Ausweitung des Begriffs Wissensmanagement auf Dokumentenmanagement bzw. die umgekehrte Erweiterung der Dokumentenmanagement-Systeme um bestimmte

Funktionalität (s.u.) in Richtung Wissensmanagement stellt einen weiteren aktuellen Trend dar [IW01c]. Hierbei ist eine enge Verbindung insbesondere mit dem Content-Management zu beobachten, da viele der aus bereits vorhandenen Dokumentenmanagement-Systemen entstehende Wissensmanagement-Systeme gleich als Internet bzw. Intranet-Systeme oder Portale konzipiert werden. Gerade das Thema Portale erfreut sich in diesem Zusammenhang hohen Interesses [KLT00, IW00a].

Andersherum ist neben diesen Entwicklungen ein in gewisser Weise gegenläufiger Trend sichtbar: Die klassischen DMS-Anbieter erhalten zugleich aus den Branchen, in die sie einzudringen versuchen, in ihrem angestammten Gebiet Konkurrenz. So statten einige ERP-Anbieter, *e-Commerce*-Spezialisten, natürlich besonders die CMS-Produzenten und sogar die Hersteller von Workflow-Management-Systemen ihre Produkte mit Dokumentenmanagement-Funktionalität aus [IW99a].

2. Die andere, technische Dimension der Trends im Dokumentenmanagement führt zur Aufwertung der Funktionalität von bestehenden DMS-Lösungen. Diese Weiterentwicklung ist als Grundlage für die zuvor beschriebenen Positionierungstrends zu sehen.

   - Eine Funktion, die heute für Dokumentenmanagement immer wichtiger wird, ist die automatische, intelligente Klassifikation von Dokumenten [IW99a]. Aufgrund des noch immer stark zunehmenden Datenumfangs gewinnt die Unterstützung von Indexierung und Recherche zunehmende Bedeutung. Gerade im Falle der Neueinführung von Dokumentenmanagement-Systemen und der Erschließung von umfangreichen Beständen an Dokumenten ist eine manuelle Nacherfassung von Stich- oder Schlagworten und Kategorien meistens nicht mehr praktikabel. Die kontinuierliche Klassifizierung beispielsweise von Nachrichtenströmen ist ein dafür prädestiniertes Gebiet. Aufgrund der Humanbarrieren bei der Wissensnutzung kann hier automatische Vorklassifizierung oder zumindest ein guter Vorschlag eine willkommene Unterstützung sein. Zudem ist es oft außerordentlich schwierig, die Konsistenz vollständig manuell gepflegter Systematiken, Klassifikationen und Thesaurii sicherzustellen. Hier versprechen Verfahren, die auf Methoden und Ergebnissen der Wissensrepräsentation, des *Information Retrieval* und der Textanalyse beruhen, erhebliche Gewinne [Die01, Büc01, Noh99, Noh00a, Hau00, Fuh96].
   - Neben der inhaltlichen Erweiterung der Funktionalität findet auch eine Verbesserung der Benutzbarkeit statt, indem bestimmte Funktionalität von Dokumentenmanagement-Systemen auf die Ebene der Betriebssysteme und der *Middleware* herunter verlagert wird [IW99a]. So wird heute eine viel weitergehende nahtlose Integration von zentralen Dokumentenmanagement-Diensten in die Arbeitsumgebung der Anwender, also des Clients, erreicht, indem beispielsweise Anwendungen direkt mit dem DMS-Server kommunizieren und Dokumente transparent in den Dokumentenspeicher einstellen, oder indem der Dokumentenspeicher als Netzlaufwerk (*File Server*) in die Arbeitsplatzumgebung der Benutzungsoberfläche eingebunden wird. Die zusätzlichen DMS-Funktionen (z.B. zur Versionskontrolle) stehen wie gewohnt als Eigenschaften der so sichtbaren Dokumente zur Verfügung.

Die wichtigsten Anbieter von Dokumentenmanagement-Systemen weltweit sind FILENET, IBM, DOCUMENTUM und OPEN TEXT, in Deutschland IXOS, SER, CE und EASY, weiterhin etabliert haben sich AIS, ACS, COI, DAA, DOCUNET, MATERNA und WIN!DMS [IW00c, IW00a]. Auch die großen betrieblichen Informationssysteme wie SAP R/3 besitzen mittlerweile Funktionalität in diesen Bereichen.

3.3.2 Content-Management-Systeme

Jeder wird ein Content Management Problem haben,
wenn nicht heute, dann morgen.
– META GROUP

Der Begriff des Content-Management und damit verbunden der des Content-Management-Systems (CMS), häufig Web-Content-Management (WCM) und Web-Content-Management-System (WCMS) genannt, ist relativ neu. Er kam ab ca. 1997 auf, als die Betreiber von umfangreichen *Web-Sites* die immer dringender werdende Notwendigkeit erkannten, die dort angebotenen Inhalte (den *Content*) systematisch zu verwalten. Eine Beschreibung der Evolution von WCMS ist in [BZTZ00] zu finden. Die bereits in Abschnitt 1.1 angedeutete Entwicklung des Internet hatte zu der Ausbildung einer Reihe von verschiedenen Geschäftsmodellen und Anwendungen wie horizontaler und vertikaler Portale und diversen Mehrwertdiensten geführt, die neue, wesentlich höhere Anforderungen sowohl an die *Quantität* als auch an die *Qualität* der angebotenen Informationen stellten. Die gestiegenen quantitativen Anforderungen betreffen einerseits eine höhere Anzahl von Dokumenten, eine stärkere Vernetzung der Inhalte und mehr vorhandene und zu unterstützende Formate, andererseits einen immer schnelleren Bedarf an aktuellen Informationen, beispielsweise von Nachrichten, Hintergrundinformationen, Bildern und Artikeln. Qualitativ schlägt sich die Entwicklung in steigender Konkurrenz mit dem Zwang zum Rationalisieren und effektivem Management der Informationen nieder, gekoppelt mit der notwendigen Steigerung der inhaltlichen Qualität. Früher eher verzeihbare Fehler wie falsch gesetzte Hyperreferenzen (tote *Links*) und veraltete Informationen sind im anbrechenden Zeitalter des elektronischen Handels nicht mehr so leicht hinzunehmen und gefährden den Erfolg der Dienste ernsthaft. Zudem erfordern neue Geschäftsmodelle, die weit über das passive Anbieten von Informationen hinausgehen, die Integration einer Vielzahl von Diensten und Systemen, beispielsweise von e-Mail für Benachrichtigungen (*newsletter*), elektronischen Geschäften (*Shops*) oder der Unterstützung von virtuellen Gemeinschaften (*Communities*) durch Diskussionsforen und *Chat*-Systeme. Die Erschließung, Integration, Vermarktung (etwa über Broker) und mehrfache Publikation von Informationen über verschiedene Medien (etwa Zeitung vs. *Online*-Ausgabe der Zeitung) stellt zusätzliche, komplexe Anforderungen an die verwendeten Systeme. Der sich in jüngster Zeit entwickelnde elektronische Handel über das Internet (*e-Commerce*) bringt wieder ganz neue Anforderungen mit sich. Insbesondere die massenhafte Verteilung, Aufbereitung und der Handel mit digitalen Gütern prägt ein neues Bild, das der *Content Economy*: Zusätzlich zum klassischen Content-Management (also der reinen Erstellung und Wartung einer *Web-Site*) muß in der *Content Economy* der gesamte Wertschöpfungsprozeß für verschiedene Medien abgedeckt werden, der aus Produktion, Management, Auslieferung (*Delivery*), *Syndication* etc. besteht [Sta01]. Die Anforderungen an die Produktionssysteme für digitale Dienste gehen weit über die klassischen Content-Management-Systeme hinaus und umfassen Skalierbarkeit, hohe Leistungsfähigkeit für große Lasten, dynamische, lokalisierte und personalisierte Dienste sowie die mehrstufige Verarbeitung einer Vielzahl von Diensten.

Historisch gesehen haben sich die Content-Management-Systeme folgendermaßen entwickelt: Nachdem der Inhalt von *Web-Sites* (HTML-Seiten) anfänglich von wenigen Experten einzeln, per Hand und direkt im HTML-Quelltext geschrieben wurden, kamen rasch leistungsfähige HTML-Editoren auf den Markt, die das Editieren direkt in der Vorschau ermöglichten (WYSIWYG). Bald darauf konnten diese Werkzeuge ganze *Web-Sites*, also die Verzeichnisstrukturen und die darin enthaltenen HTML-Dateien, verwalten und die Referenzen zwischen den einzelnen Seiten automatisch überwachen und anpassen. Die so erstellten *Web-Sites* werden als statische *Web-Sites* bezeichnet. Einige ganz wesentliche Probleme waren damit aber noch nicht gelöst, deren Existenz die Entwicklung der Content-Management-Systeme ausgelöst hat: Zum einen sind die o.g. Werkzeuge inhärent nur für die Benutzung durch Einzelpersonen konzipiert; die verteilte Arbeit

in einer Gruppe an demselben Projekt (nämlich einer *Web-Site*) wird nicht oder nur sehr rudimentär ermöglicht, und redaktionelle Publikationsprozesse wie etwa die Freigabe von Artikeln, die sehr wichtig für Zeitungs-Redaktionen sind, werden nicht unterstützt. Zum anderen findet eine im Sinne der einheitlichen graphischen Präsentation und der typischen Aufgabenteilung in Redaktionen eigentlich wünschenswerte Trennung von Inhalten und Layout nicht statt, insbesondere da HTML als Medium des WWW dies mangels passender Konzepte nicht direkt zuläßt. Zudem stößt eine rein auf HTML-Design ausgerichtete Strategie bei der Einbindung von weiteren Diensten und Systemen rasch auf unüberwindbare Hürden. Das Erstellen und Betreiben von dynamischen *Web-Sites*, also solchen, bei denen große Teile des Inhalts erst zum Zeitpunkt der Auslieferung generiert werden, erfordert eine neue Klasse von Systemen.

Entsprechend dieser Anforderungen haben sich zwei Dimensionen von Content-Management herausgebildet [SW00b]: Die Anforderungen aus dem Betrieb rein informationspräsentierender *Web-Sites* mit keiner oder minimaler zusätzlicher Funktionalität, und die aus dem Betrieb von Geschäftsplattformen mit einer hohen Integrationsdichte weiterer, bestehender Informationssysteme, z.B. von betrieblichen Informationssystemen wie Warenwirtschaftssystemen (ERP), Kundenbeziehungsmanagement (CRM) etc. In beiden Fällen kommen Anforderungen wie Personalisierung, Sicherheit und Datenaustausch hinzu. Auch wenn alle diese Anforderungen zunächst relativ einheitlich erscheinen mögen, gibt es doch eine Reihe von Content-Management-Systemen, die sich den Problemen mit unterschiedlichen Ansätzen und Schwerpunkten aus verschiedenen Richtungen nähern, je nach dem, ob die Systeme historisch durch ihre Entwicklungsgeschichte bedingt ihre Wurzeln eher in Dokumentenmanagement-Systemen, Redaktionssystemen, Workflow-Management-Systemen oder Datenbank-Systemen haben [SW00a].

Trotz (oder wegen?) der Aktualität des Themas und der rasanten Entwicklung der Systeme wurden Content-Management-Systeme als Informationssystemklasse und die softwaretechnischen Fragestellungen bisher von der akademischen Forschung (insbesondere der Softwaretechnik) nur begrenzt wahrgenommen.[3] Zur Zeit dominieren Beiträge aus dem wirtschaftlichen Umfeld (Unternehmensberatungen, Fachpresse) sowie aus der wirtschaftswissenschaftlichen Fakultät, z.B. [SW00b, SW00a, Kes00]. Die Literaturangaben etwa in [SW00b, Noh00b] belegen dies recht deutlich. Hier wird nun versucht, nach einer Definition des Begriffs CMS die wesentlichen Konzepte und Funktionen in Abgrenzung zu Dokumentenmanagement, Portalen und Wissensmanagement-Systemen zu identifizieren.

Definitionen von Content-Management finden sich viele, genannt seien hier zunächst drei aus [Kes00] entnommene:

> „[Content Management ist die] Kombination klar definierter Rollen, formaler Prozesse und unterstützender Systemarchitektur, die Firmen zur Erstellung, Kollaboration, Kontrolle und Veröffentlichung von Internet-Seiten nutzen."
> (FORRESTER RESEARCH)

> „[Content Management ist die] vordefinierte Anzahl von Aufgaben und Prozessen beginnend mit der Kreation bis hin zur Archivierung von Content mit der Zielsetzung, diesen im Web zu publizieren."
> (OVUM)

> „[...] Content-Management-Systeme als die zentrale Internet-Infrastruktur-Software, die es ermöglicht, die Geschäftsprozesse zur Content-Erstellung, -verwaltung und

[3] Der Begriff „Content-Management" findet sich beispielsweise in bisher (August 2001) in keiner Publikation der *Lecture Notes in Computer Science* (LNCS).

> -Kollaboration sowie die Bereitstellung des Content softwareseitig abzubilden und damit erfolgreich die Inhalte für Web-Seiten und andere web-bezogene Endgeräte (z.B. WAP-*Cellphone*, PDA) zu managen."
> (WESTLB PANMURE)

Diese Definitionen sind insoweit hilfreich, als daß sie alle drei die Wichtigkeit der Einzelaufgaben und arbeitsteiliger (Geschäfts-)Prozesse herausheben, z.B. von Erstellung, Kontrolle, Publikation oder Archivierung von Inhalten. In [RR01] finden sich zwei weitere Definitionen von Content-Management, eine dort als Definition von Content-Management im engeren Sinne bezeichnete, und eine als Definition im weiteren Sinne. Die erste lautet:

> „Software-basiertes Content Management befasst sich mit der systematischen Sammlung und Verwaltung von Informationsbausteinen in einem einzigen (logischen) Bestand. Es stellt Anfragemethoden und Mechanismen für die sichere Arbeit ganzer Nutzergruppen mit diesem Inhaltsbestand (‚Content Base') bereit."

Unter dieser sehr allgemeinen Definition lassen sich allerdings auch Dokumentenmanagement-Systeme und bei entsprechender Auslegung alle Datenbanken und Informationssysteme subsumieren. Die zweite Definition lautet:

> „Software-basiertes Content Management befasst sich mit der systematischen Sammlung, Erstellung, Speicherung und Veredelung von strukturierten Inhalten und Mediendaten aller Art in einem einzigen, fein granulierten (logischen) Bestand. Es unterstützt gezielt die die sichere Aggregation, Veredelung, Verarbeitung, Auswertung und Wiederverwendung dieser Content Base durch ganze Benutzergruppen."

Es werden also gegenüber dem Dokumentenmanagement neue Schwerpunkte gesetzt. Gerade die letzte Definition beschreibt alle Tätigkeiten, die zu Wertschöpfungsprozessen in der Informationsverarbeitung gehören (insbesondere Erstellung und Veredelung). Die Zielrichtung liegt dabei stark auf dem kooperativen Publikationsprozess. Zusammenfassend wollen wir in enger Anlehnung an die letzte Definition Content-Management als die systematische Verwaltung von beliebig strukturierten Inhalten in einem zentralen Bestand zum Zwecke der Erstellung, Veredelung, Fortschreibung und Publikation verstehen. Der entscheidende Unterschied zum reinen Dokumenten-Management ist der Zweck, nämlich die Publikation.[4] Unbefriedigend ist in allen Definitionen die Wahrnehmung des Content-Begriffes selbst; hier ist noch eine beträchtliche Schärfung zu leisten.

Aufgrund der mittlerweile relativ hohen Zahl von unterschiedlichen Content-Management-Systemen und der bislang unscharfen Definitionen des Content-Begriffs selbst ist eine nähere Diskussion der häufigsten Anforderungen und Funktionen solcher Systeme nötig. Betrachtet man den Begriff des Content-Management-Systems, so ergeben sich – [SW00b] folgend – von alleine die drei Aspekte des Content, des Managements und der Systeme, die im folgenden jeweils systematisch beleuchtet werden.

1. Zum **Content**. Wie bereits bei den Dokumentenmanagement-Systemen stellt sich die Frage nach der Definition der Informationsartefakte, hier also des *Contents* oder des Inhalts. Der Begriff des Dokuments bei Dokumentenmanagement-Systemen entspricht dem des

[4] Der in letzter Zeit häufig genannte Trend der Einführung von Content-Management für interne Inhalte einer Organisation (etwa in einem Intranet) kann man so als Mißverständnis auslegen, da von der Intention her eigentlich Dokumentenmanagement gemeint ist und überhaupt kein Publikationsprozess stattfindet.

Inhaltsobjektes (Content-Objektes) hier. Häufig wird im Content-Management von Dokumenten gesprochen, so daß für diesen Kontext eine neue Definition für Content bzw. Dokument nötig wird.

Die traditionelle Aufgabe des Publizierens von Dokumenten betrachtet das Dokument – als Erzeugnis des Prozesses – als strikt sequentielle und nach der Fertigstellung unveränderliche Inhaltsmenge [RR01]. Diese Sicht der Dinge ändert sich mit den hypermedialen Eigenschaften der neuen Content-Objekte dramatisch: Die Erzeugnisse bzw. die verwendeten und gleichfalls vom CMS verwalteten Vorprodukte sind nicht mehr notwendigerweise sequentiell geordnet, sondern vielfältig vernetzt; und sie sind nicht mehr unbedingt dauerhaft und unveränderbar, sondern hoch dynamisch, ständig veränderbar oder werden sogar ad hoc erzeugt. Verkompliziert wird die Lage weiter durch die häufig aus juristischen Gründen notwendige Versionierung, also dem dauerhaften Speichern aller jemals publizierten Inhalte. Dieses völlig neue Umfeld hat zu einer gegenüber der Sichtweise des Dokumentenmanagements stark erweiterten Definition geführt: Nachdem der Dokumentenbegriff bei Dokumentenmanagement-Systemen von einer unstrukturierten Einheit in Form einer Datei ausgeht, stellt sich das Dokument hier wesentlich strukturierter dar, und es wird unter verschiedenen Aspekten betrachtet. Dazu trennen alle modernen Content-Management-Systeme zwischen den grundsätzlichen (expliziten oder impliziten) Bestandteilen eines Dokuments: Struktur, Inhalt und Darstellung. Als vierte Komponente wird in [BZTZ00] die der Funktionalität genannt, die häufig logischer Bestandteil von aktiven Dokumenten ist.

Struktur: Die Strukturdefinition eines Dokumentes legt den Aufbau und die Einzelinformationen eines Dokuments fest. Dies entspricht dem Typ einer Variablen in Programmiersprachen oder dem Schema eines Datenbankobjektes. Es wird deshalb oft von Dokumenttyp gesprochen. Die wesentliche Veränderung gegenüber dem in Abschnitt 3.3.1 eingeführten Dokumentbegriff ist, daß das Dokument überhaupt eine wohldefinierte Struktur aufweist und so identifizierbare, semantisch relevante Teile oder Attribute (Informationen im Sinne der gebrauchten Terminologie) existieren. Eine besonders wichtige Art von Informationselement ist die Verknüpfung (Link) eines Dokumentes mit einem anderen, wobei die Strukturdefinition bereits die verknüpfbaren Dokumenttypen und die Kardinalitäten festlegen kann; dies wird häufig später zur Erhaltung der referentiellen Integrität des Inhalts, d.h. zum Vermeiden von gebrochenen Verknüpfungen (*broken links*) genutzt. Ein Beispiel wäre die Struktur einer Preisliste; diese bestünde etwa aus einer Liste von strukturierten Einträgen aus Name, Produktnummer und Preis.

Inhalt: Der Inhalt eines Dokuments füllt gemäß der Strukturdefinition die Elemente des Dokuments mit Informationen. In dem obigen Beispiel wäre das also die Liste von Werten, bestehend aus Zeichenketten und Zahlen. Wichtig ist dabei, daß der Zusammenhang von Inhalt und Struktur stets bestehen bleibt, so daß jedem Wert das korrespondierende Strukturelement eindeutig zuzuordnen bleibt.

Darstellung: Getrennt von Struktur und Inhalt läßt sich für jedes Dokument die Darstellung oder die Darstellungen (es sind mehrere, alternativ je nach Kontext oder Zweck verwendbare denkbar) definieren. Häufig wird dies als Layout bezeichnet. Das eigentliche Dokument wird also nicht mit den Formatierungsanweisungen vermischt; diese werden separat gepflegt. Im Dokument wird lediglich (entweder explizit oder implizit durch an anderer Stelle definierte Regeln) festgehalten, welche Formatierung zu verwenden ist. Diese Formatierungen stellen dann ggf. kontextabhängige Regeln bereit, um einzelne Werte der verschiedenen Strukturelemente gemäß eines einheitlichen Erscheinungsbildes in ein gewünschtes Ausgabeformat (z.B. HTML) zu transformieren.

Funktionen: Die klassische Sichtweise von statischen, unveränderbaren Inhalten wird gerade beim Publizieren von Web-Inhalten stark durch das Aufkommen von dynamischen oder semidynamischen Anteilen abgelöst [BZTZ00]. Die Einbettung von dynamisch zur Ansichtszeit generierten Informationen stellt eine weitere Dimension des Dokuments dar, die entsprechend getrennt von Inhalt, Layout und Darstellung spezifiziert werden muß.

Eine wichtige konzeptuelle Erweiterung der Vierteilung eines Dokuments in Struktur, Inhalt, Layout und Funktionalität stellen die Metainformationen[5] dar. Metainformationen – also Informationen über die Informationen selbst – sind beispielsweise die verwendete Strukturdefinition (der Dokumententyp), das zu verwendende Layout, Autoren, die Lebensdauer, das Erstellungs- und letzte Änderungsdatum, und ggf. eine Historie davon. Viele weitere Metainformationen sind denkbar (Klassifikationen, Schlagworte, Urheberrechte etc.). Technisch realisiert werden können sie zum Teil auf die gleiche Art und Weise wie normale Informationselemente eines Dokuments, konzeptionell stellen sie aber eine andere Klasse dar [DH98].

Die vorgenommene Vierteilung (berücksichtigt man Metadaten separat, sogar Fünfteilung) ist stark durch die Existenz und die Funktionsweise der *Extensible Markup Language* XML [WWW98a], XML-Schemata [WWW99] sowie *Stylesheets* und Transformationen [Dea99, Cla99] motiviert. Diese setzen genau die Dreiteilung in Struktur (XML-Schema bzw. DTD), Inhalt (XML) und Layout (XSL, XSLT) um [RR01, Tol99], wobei die Funktionalität in XML noch nicht berücksichtigt wird. Dennoch sind ganz andere Möglichkeiten zur Umsetzung der Trennung vorhanden; XML-basierte Lösungen sind nicht zwingend dafür nötig. Grundsätzlich wird – häufig im Zuge der *Content Syndication* – durch die Trennung von Inhalt und Layout angestrebt, aus denselben Informationen mehrere unterschiedliche Repräsentationen kontextabhängig für verschiedene Zwecke (unterschiedliche Endgeräte, Medien oder Benutzer) generieren zu können. Das Prinzip lautet dabei, daß stets nur eine einzige Quelle vorhanden sein darf (*single source*), von der dann mehrere Ausgabeformate abhängen. Das Schlagwort des *Cross media publishing* beschreibt diesen Ansatz [Kam01].

2. Zum **Management**. Wie aus den obigen Definitionen bereits gut herauszulesen ist, umfaßt Content-Management zahlreiche arbeitsteilige Prozesse. Die wichtigsten Prozesse und dazugehörigen Funktionen, die ein System unterstützen muß, sind nach [Kes00, SW00b, SW00a]:

   - Benutzerverwaltung für Personen, Gruppen, Rollen mit den jeweiligen Rechten;
   - Erstellung und Pflege (Administration) der Struktur des Informationsangebots; dazu gehören die Definition des strukturellen Aufbaus, der Dokumenttypen und die Festlegung des Erscheinungsbildes (des Layouts);
   - Unterstützung des Lebenszyklus des Inhaltes: Erstellung, Pflege und Wiederverwendung der redaktionellen Inhalte und Dokumente (*Authoring*) gemäß des wirksamen Rechte- und Rollenkonzeptes, Versionierung der Inhalte und Archivierung alter, nicht mehr aktueller Inhalte. Insbesondere die automatische Beibehaltung konsistenter Verknüpfungsstrukturen (*Link-Management*) und Unterstützung bei der automatischen Generierung von Navigationsstrukturen zählt hier zu den essentiellen Anforderungen. Zusätzlich sind mächtige Recherchefunktionen über Metadaten und Volltexte von Vorteil;
   - Abbildung redaktioneller Prozesse (Workflows) wie Qualitätskontrolle, Freigabe und Publikation;

[5] Häufiger Metadaten genannt.

- Unterstützung für automatischen und manuellen Import und Export von Informationen, etwa zur Einbindung weiterer Informationsquellen wie Nachrichtenströme, zum Import von Dateien in Fremdformaten oder zum Erstellen von *Offline*-Kopien des gesamten Informationsangebotes oder Teilen darauf auf CD-ROM (*Cross media publishing*).

Wichtig für diese Funktionen ist, daß sie auch von Benutzern ohne technische Qualifikation zu leisten sind, eventuell mit Ausnahme der oben als Administrationsaufgaben gekennzeichneten Teile.

3. Zu **Systemen**. Praktisch eingesetzte Systeme haben in Abhängigkeit von ihrer Historie sehr verschiedene Architekturen und Funktionsweisen. In allen Fällen sind sie jedoch Client/Server-Lösungen, wobei der Server die Kernfunktionalität der Verwaltung und Ablage von Struktur, Inhalt und Layout sowie die (technische) Publikation der Erzeugnisse leistet. Im Client stehen Werkzeuge zur Erstellung und Bearbeitung des Inhalts gemäß der oben aufgeführten Funktionen bereit.

   Zur Realisierung der Komponenten werden unterschiedlichste Technologien eingesetzt, für die Datenhaltung des Servers werden beispielsweise relationale Datenbanken, objektorientierte Datenbanken oder spezielle XML-Datenbanken wie TAMINO von der SOFTWARE AG eingesetzt [Sof01, Tam99]. Letztere unterstützt die Trennung von Struktur, Inhalt und Layout naturgemäß besonders gut. Für die eigentliche Anwendungslogik der Server kommt eine Vielzahl von Sprachen zum Einsatz; die Spanne reicht von Skriptsprachen wie PERL und TCL über die Verwendung von Produkten wie LOTUS NOTES und Technologien wie ASP und JSP. Einige moderne Systeme sind vollständig in der plattformunabhängigen Sprache JAVA entwickelt [SW00b]. Auf weitere technische Architekturmerkmale wird später bei konkreten Produkten in Abschnitt 3.4 noch genauer eingegangen.

   Eine essentielle Komponente aller Systeme ist der eigentliche Web-Server, der letztendlich die fertig publizierten Inhalte an die Web-Browser der anfragenden Nutzer ausliefert und die Generierung der dynamisch erzeugten Inhalte erledigt bzw. auslöst. Trotz der unterschiedlichen Technologien und Implementierung lassen sich in allen Systemen dafür mehrere verschiedene immer verwendete Prinzipien für die Arbeitsweise des Publikationsprozesses bzw. für die Architektur des Systems identifizieren. Es kommen im wesentlichen entweder das Prinzip des *Staging*-Servers oder des *Live*-Servers zum Einsatz (siehe Abbildung 3.3).

   Im Falle des *Live*-Servers in Abbildung 3.3 a) sind der Content-Server, mit dem die internen redaktionellen Aufgaben unter Nutzung spezieller Redaktions-Clients abgewickelt werden, und der Web-Server, der die Auslieferung an die externen Nutzer im Internet leistet, identisch. Das Freischalten bzw. Publizieren der Dokumente ist dabei unter Umständen kein separater Schritt und Änderungen werden sofort sichtbar; alternativ dazu sind Strategien denkbar, bei denen die Dokumente erst explizit publiziert werden müssen, um extern sichtbar zu werden. Der Server legt alle Daten in einer gemeinsamen Datenbank ab. Ein Problem, das bei einer solchen Konfiguration entstehen kann, ist das der Skalierbarkeit und der Last, da beispielsweise größere Publikationsfreigaben die Leistung des externen Web-Servers beeinträchtigen können, und andersherum kann hohe externe Last die Redaktionsaufgaben behindern [Kes00].

   Das Konzept des *Staging*-Servers in Abbildung 3.3 b) sieht eine Zweiteilung in einen Produktionsserver, der alleine für die Unterstützung aller redaktionellen Aufgaben verantwortlich ist, und einen Publikationsserver vor, der ausschließlich die externen Nutzer bedient. Die Freigabe (Publikation) von Dokumenten geschieht explizit und stößt die Übertragung der geänderten oder neu hinzugekommenen Inhalte vom Produktionsserver auf den Publikationsserver an bzw. bewirkt das Löschen alter, nicht mehr benötigter Inhalte

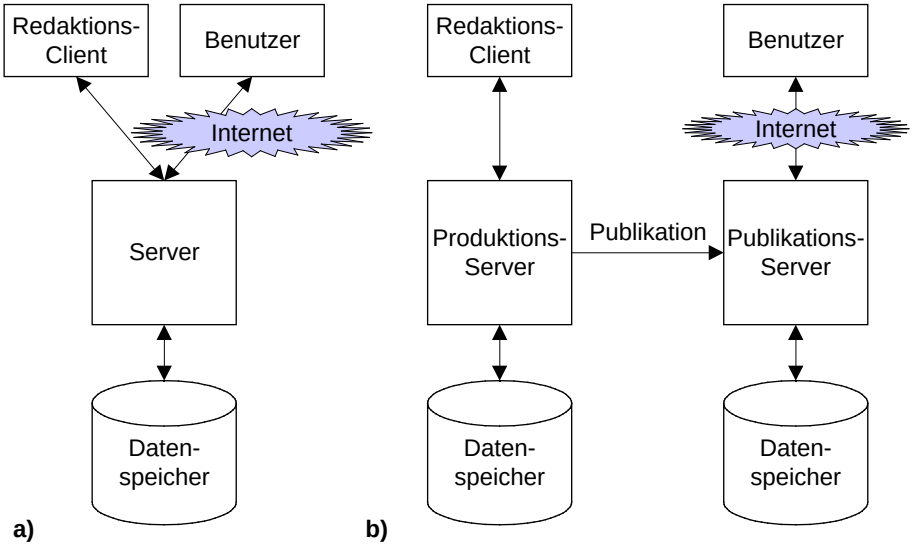

Abbildung 3.3: Architekturen von Content-Management-Systemen

auf dem Publikationsserver. Idealerweise sind die beiden Server auf verschiedenen Rechnern lokalisiert, so daß das Lastverhalten beider Komponenten sich nicht gegenseitig beeinflußt. Lediglich der Publikationsprozeß stellt hier noch eine potentielle Gefahr dar. Um diese Probleme wirksam zu bekämpfen, ist die Architektur der Publikationsserver selber in Hochleistungssystemen nochmals mehrstufig und evtl. verteilt auf mehrere Rechner aufgebaut, um eine optimale Lastbalancierung zu erhalten [Cor00, Sta01].

Eine weitere Dimension der Komplexität des Publizierens entsteht aus der Tatsache, daß entweder beim Publizieren der Inhalte oder zumindest beim ersten Abruf des Inhaltes von externer Seite das engültige Ausgabeformat für das Zielgerät (also HTML für Web-Browser, WML für WAP-Geräte etc.) aus den bislang getrennt vorliegenden Inhalten, Strukturen und Layouts generiert werden muß, was durchaus kostenintensiv bezüglich Rechenzeit und Speicherplatz sein kann. Besonders kritisch wird dieser Schritt bei dynamischem Inhalt, der bei jedem Abruf neu berechnet werden muß, etwa für personalisierte Dienste oder weitere integrierte Funktionalität. Hier sind sowohl intelligente, mehrstufige Cache-Strategien innerhalb der Systeme als auch wieder gute Verteilungsmöglichkeiten und redundante Auslegung der einzelnen Systembestandteile erforderlich, um bei hoher Last keine gravierenden Leistungseinbußen zu erleiden.

Angesichts der Vielzahl von unterschiedlichen Systemen und der verschieden ausgelegten Funktionalität gehört eine gründliche Planung zur Auswahl eines Content-Management-Systems, ein Aspekt, der hier nicht weiter verfolgt wird. In [SW00b] findet sich beispielsweise eine umfangreiche Methodik zur Auswahl mit einer gut vorbereiteten Aufstellung von detaillierten Einzelanforderungen, oder in [Noh00b] ein konzentrierterer Überblick darüber. Die allgemeine Planung von Web-CMS-Projekten beschreibt [BZTZ00].

Wichtige internationale Anbieter sind nach [IW00b] ALLAIRE, BROAD VISION, DOCUMENTUM, FILENET, GAUSS INTERPRISE, INFOPARK, INTERWOVEN, PIRONET, TRIDION und VIGNETTE. Ein umfassenderer Einblick ist in [SW00b] zu finden, wo über 50 Anbieter und Produkte tabellarisch mit den wichtigsten Funktionen aufgeführt sind.

3.3.3 Portale

Das vorherrschende Thema des frühen Internets war der sogenannte „Browser-Krieg“ zwischen NETSCAPE und MICROSOFT. Nachdem er inzwischen weitgehend zu Gunsten des letzteren Unternehmens entschieden wurde, hat sich das Hauptinteresse auf die Inhalte, Geschäftsmodelle, Konzepte und Techniken der Angebote verlagert. Das Thema *Portale*, das zusammen mit dem Begriff etwa ab 1998 aufkam [SS99], steht dabei für diese Entwicklung; in [MAB+99] ist sogar recht treffend die Rede von „Portalmania“. Historisch gesehen sind drei Phasen bis zur Entwicklung der Portale zu verzeichnen, die im folgenden kurz skizziert werden. Diese Schritte sind zugleich als Versuche zu verstehen, die wachsende Menge des über das WWW verfügbaren Wissens besser zu erschließen.

Durch das Entstehen des *World-Wide-Web* ist es wie nie zuvor möglich, gleichzeitig auf große Bestände an Wissen zuzugreifen und selber große Wissensbestände für andere verfügbar zu machen. Die damit einhergehende Problematik, in der unstrukturierten Fülle von Informationen die benötigte zu finden, bildet die Kehrseite der Medaille. Diesem Phänomen der Informationsüberflutung (*information overload*) wurde im WWW in mehreren Schritten Rechnung getragen. In der Anfangsphase des WWW war die Benutzung zum größten Teil auf Studenten und wissenschaftliche Mitarbeiter an Forschungseinrichtungen und Universitäten beschränkt, die Zugang zum Internet hatten. Als Informationsressourcen dominierten persönliche *homepages* und per Hand gepflegte Listen von Verweisen, als Werkzeuge kamen die *Bookmark*-Verzeichnisse der Web-Browser und als Kommunikationsmedium e-Mail und Diskussionsgruppen (*newsgroups*) zum Einsatz. Die Inhalte waren meist zielgruppenspezifisch eingeschränkt.

Bedingt durch das schnelle Wachstum der Anzahl der Seiten wurden neue Lösungen in Form von Suchmaschinen gefunden, die als Teil einer *Web-Site* in den Web-Server eingebunden wurden [MV98]. Als erster Vertreter ist hier ALTAVISTA zu nennen. Der Grundgedanke der Suchmaschinen, alle vorhandenen, d.h. transitiv erreichbaren, Web-Seiten durch sogenannte Web-Roboter herunterzuladen und zu indexieren [Tur98], scheiterte wiederum an dem anhaltenden Wachstum der Quellen. Ergebnisse von Anfragen waren (und sind häufig noch immer) riesige Treffermengen mit zweifelhafter Relevanz. Um dieser Entwicklung entgegenzuwirken, wurden die ersten *horizontale Portale* aufgebaut [KLT00], die den einfachen, strukturierten und kategorisierten Zugang zu Inhalten und Verweisen zu allen Themen durch manuelle Pflege der Betreiber ermöglichten. Der erste Vertreter war hier YAHOO. Als Portal wollen wir hier zunächst bis zu einer eingehenderen Definition eine *Web-Site* verstehen, die Zugang zu Informationen strukturiert nach Branchen, Sachgebieten oder Unternehmensbereichen anbietet. Mittlerweile sind allerdings auch beträchtliche Fortschritte auf der Seite der Suchmaschinen gemacht worden: Durch Ausnutzung von Kontextinformationen, der *Link*-Struktur und anderer Hinweise ist die Trefferrelevanz deutlich gestiegen; ein viel beachtetes Beispiel ist die Suchmaschine GOOGLE. Die dort verwendeten Techniken [BP98b] gehen bereits stark in die Richtung der automatischen Klassifikation und des *Information Retrieval*.

Ergänzt wurde dieses Konzept in der Folge durch das Angebot von eigenen oder zu diesem Zweck eingekauften Inhalten, also Informationen aller Art wie Artikeln, Bildern, Musikstücken etc. Beispiele für solche Anbieter sind AOL, MSN, T-ONLINE etc. Vielfach sind diese Anbieter ursprünglich reine Zugangsanbieter (*Provider*) gewesen, die ihren Endbenutzern zuvor Netzzugang über proprietäre Systeme, also nicht das Internet, ermöglicht hatten. Dem Zwang, die proprietäre Zugangssoftware und die Protokolle den herrschenden (de-facto) Standards anzugleichen, haben sich mittlerweile alle diese Anbieter gebeugt. Als Beispiel sei COMPUSERVE genannt.

Der nächste Schritt der Entwicklung war das Entstehen einer großen Anzahl von thematisch spezialisierten Diensten, die mit für das jeweilige Themengebiet erhöhter Informationsqualität und

speziellen Dienstangeboten die sogenannten *vertikalen Portale* schufen. Durch den Zukauf von redaktionell gepflegtem Inhalt von klassischen Informationsanbietern wie Zeitungen, Fachverlagen, Bildagenturen und Nachrichtenagenturen und durch den sich entwickelnden elektronischen Handel (*e-Commerce*) wurden völlig neue Geschäftsmodelle entwickelt (Information bzw. Wissen als Wirtschaftsgut!), bei denen zum Teil allerdings noch auf den Erfolg gewartet werden muß. Die Integration von weitergehenden Anwendungen, die bisher nicht über das WWW verfügbar waren, für die Kunden dieser Dienste bildet die derzeit jüngste Entwicklung.

Neben den eben beschriebenen Informationsanbietern, deren Kerngeschäft das Betreiben eines Informationsdienstes im WWW ist, haben inzwischen sehr viele andere Unternehmen das WWW als Medium zur Information der Kunden, zur Abwicklung von Geschäften (*e-Commerce*), zur Kundenbindung und zum internen Einsatz im Unternehmen entdeckt, und so entwickelte sich zusätzlich der Begriff des *Unternehmensportals*. Insbesondere darauf ist das exponentielle Wachstum des WWW zurückzuführen. Genauso wie bei den Informationsanbietern ist dabei eine Entwicklung von einfachen, statischen Strukturen zu komplexen Portalen, die personalisierte Mehrwertdienste für die Benutzer zur Verfügung stellen, zu verzeichnen.

Die anwendungsorientierte Entwicklung der WWW-basierten Systeme wurde auf der Seite der Systemtechnik natürlich parallel nachvollzogen. Mit den wachsenden und teilweise völlig neuen Anforderungen an die zu verwendenden Systeme wurden einerseits immer leistungsstärkere und vor allen Dingen funktionsreichere Systeme zum Betreiben der Dienste entworfen und implementiert, andererseits fand die Entwicklung einer ganz neuen Klasse von Anwendungen, eben den bereits untersuchten Content-Management-Systemen (CMS) statt, die es gestatten, die ständig wachsende Komplexität der redaktionellen Verwaltungsaufgaben zu beherrschen. Sowohl konzeptuell als auch technisch gesehen ist dabei zu beobachten, daß viele Anleihen bei der Klasse der Dokumenten-Management-Systeme (DMS) gemacht werden (vgl. Abschnitt 3.3.1).

Nicht zuletzt aufgrund der schnellen Entwicklung kann derzeit nur von einer äußerst uneinheitlichen bis chaotischen Terminologie in diesem Bereich gesprochen werden. Beinahe jede umfangreichere *Web-Site* erhält mittlerweile von ihren Betreibern das Prädikat Portal, eine inflationäre Menge von einander unterschiedlich stark überlappenden bis völlig disjunkten neuen Unterbegriffen und eine Fülle von Software-Produkten zur Erstellung, Pflege und dem Betrieb von bestimmten Arten von Portalen sind auf dem Markt anzutreffen. Dies belegt natürlich die hohe Relevanz des Themas und fordert zur systematischen Untersuchung dieser Phänomene auf. Da der Portalbegriff eine zentrale Grundlage für diese Arbeit darstellt, ist eine Untersuchung dieses Begriffs und eine Klassifikation seiner verschiedenen Erscheinungsformen die Aufgabe des restlichen Abschnitts.

Im allgemeinen Sprachgebrauch versteht man unter einem Portal einen „baulich hervorgehobenen, repräsentativ gestalteten größeren Eingang an einem Gebäude“ [Dud89], „die oft bedeutende Ausmaße annehmende Außentür eines Baues“ [Bro89] oder einen „durch architektonische Gliederung und plastischen Schmuck hervorgehobenen Eingang von Tempeln, Kirchen und Palästen“ [Lex91]. Eine Sammlung von weiteren ähnlichen Definitionen, insbesondere aus Online-Nachschlagewerken, findet sich in [Wal01b].

Die Zugangsfunktion eines Portals steht damit außer Frage: Ein Internet-Portal bietet den interessierten Benutzern einen – häufig ausgeschmückten, d.h. mit diversen Zusatzfunktionen versehenen – Überblick und Einstieg in für sie relevante Informationen unter Nutzung des Internet[6]. Wesentliches Charakteristikum des Portals ist, daß es den gemeinsamen Zugang zu einer Fülle von weiteren Informationen unter einem Dach (hier greift die Metapher des Gebäudes wieder) bildet, was manchmal als *single point of access* bezeichnet wird. An den Einstieg schließt sich na-

[6]Tatsächlich ist hier *Internet* gemeint, da sich die Nutzung nicht auf das WWW beschränken muß, sondern Portale auch Dienste wie e-Mail, FTP etc. nutzen können. Siehe dazu die Fußnote auf Seite 2.

türlich meist reichhaltige Funktionalität an. Grundsätzlich beschreibt der Begriff des Portals also nach außen hin für Benutzer sichtbare Informationen und Funktionalität. Dies steht im Kontrast zu den Sichtweisen in den letzten beiden Abschnitten 3.3.1 und 3.3.2, die eine Sicht auf die Systeme von innen besitzen und die sich insbesondere im Falle der Content-Management-Systeme mit der zielgerichteten, effizienten Erstellung und dem Betrieb von *Web-Sites*, also auch Portalen, beschäftigen. Zusammengefaßt kann man also behaupten, daß Portale (nicht ausschließlich) die Produkte von Content-Management-Systemen sind. Natürlich bleibt die Erstellung von Portalen ohne Content-Management möglich.

Die für eine Sichtweise von außen wichtigen Fragen bei der Analyse von Portalen sind die nach Zielgruppen und Funktionalität, sowie die im Rahmen dieser Arbeit nicht verfolgten Fragen der Finanzierbarkeit, von Geschäftsmodellen und Märkten. Gerade zu diesen Aspekten gibt es von vielen international tätigen Unternehmensberatungen zahlreiche Studien, die als Adressaten hauptsächlich Investoren haben. Einige gehen dennoch in gewissem Rahmen terminologischen Fragen und Fragen der Funktionalität nach, so etwa [Kes00, Gol99, GR01, Ver00b, MAB+99]. Ansonsten gilt wiederum das schon Gesagte bezüglich der Rezeption seitens der Informatik: auch hier tut sich die Wirtschaftsinformatik deutlicher hervor; zur Zeit eher noch mit Diplomarbeiten (z.B. [Mos00, Fau00, Sch99a]) und Arbeitspapieren (z.B. [SS99]). Im folgenden werden zunächst verschiedenen Arten von Portalen definiert und voneinander abgegrenzt werden, um darauf aufbauend die Zielgruppen und die wichtigsten Funktionen zu beschreiben.

Eine häufig anzutreffende Unterscheidung differenziert zwischen dem *Consumer-Portal* und dem *Enterprise-Portal* [SS99, MAB+99]. Statt *Consumer-Portal* liest man auch Web-Portal oder einfach nur Portal. Andererseits ist die Unterteilung in *horizontale* und *vertikale* Portale gängig [KLT00], die meistens nicht in Verbindung mit den obigen Definition gebraucht wird. Als Ausgangsbasis für alle weiteren Portaldefinitionen wollen wir uns der allgemein gehaltenen Definition aus [KLT00] bedienen:

> „Ein Web-Portal ist eine *Website* im *World-Wide-Web*, die Informationen aus verschiedenen, ausgewählten Quellen zusammenfasst und ihren Nutzern über einen Standard-Web-Browser einen (personalisierten) Zugang mittels Suche und/oder Navigation von Verzeichnisstrukturen bietet, gegebenenfalls ergänzt um redaktionellen Inhalt, Funktionalität zur Kommunikation und/oder Informationsverarbeitung."

Wir wollen im folgenden die Begriffe Portal und Web-Portal als synonym betrachten. Zugleich wird – ebenfalls sehr allgemein – in [KLT00] definiert, was horizontale und vertikale Portale unterscheidet:

> „Ein horizontales Web-Portal deckt einen weiten Bereich von Themen ab, während ein vertikales Web-Portal auf einen eingeschränkten Themenbereich in größerer Detailliertheit fokussiert."

Die Wortschöpfung „Vortal" als Abkürzung für vertikale Portale findet sich teilweise in der Literatur [Ver00b, Wal01b]. Eines der bekanntesten vertikalen Portale ist AMAZON, das auf den Verkauf von Büchern etc. spezialisiert ist. Die Zielgruppe von horizontalen Portalen ist im Gegensatz zu vertikalen Portalen nicht speziell und das Angebot soll alle Nutzer gleichermaßen ansprechen. Klassische Beispiele für horizontale Portale sind die Portale von AOL, YAHOO, LYCOS, ALTAVISTA, INFOSEEK, EXCITE, MSN oder T-ONLINE. Die großen horizontalen Portale, entstanden als reine Informationsanbieter, die strukturierten und kategorisierten Zugang zu verschiedensten weiteren Quellen anboten, mußten bald einerseits ihr Angebot um eigene oder zu diesem Zweck eingekaufte Inhalte, also Informationen aller Art wie Artikel, Bilder, Musikstücke

etc. erweitern, und andererseits neue, zusätzliche Dienste integrieren, die erhöhten Nutzen für die Kunden versprachen, wie etwa e-Mail und Diskussionsforen. Um gleichermaßen den Mehrwert für die Kunden und die Kundenbindung an den Anbieter zu steigern, sind mittlerweile alle Dienste dazu übergegangen, größere Teile des Angebots zu personalisieren. Schlüsselelemente der Strategien sind der Aufbau von Gemeinschaften (*Communities*) zur Kundenbindung sowie von neuen Diensten wie *e-Commerce* und länderspezifische Anpassungen (z.B. Lokalisierung der Sprache) zur Steigerung der Kundenzufriedenheit [Par00]. Typische Dienste und Angebote, die heute von horizontalen Portalen bereitgehalten werden, sind folgende [MPR00, SS99]:

- Web-Suchmaschinen über die Volltexte aller indizierten Seiten und strukturierte, manuell gepflegte Verzeichnisse von Informationsquellen (*Directories*); manche Suchmaschinen bieten zugleich Übersetzungsdienste an;
- e-Mail; dabei werden häufig eine web-basierte Benutzungsoberfläche und zusätzlich der Zugriff über die Protokolle POP und SMTP sowie manchmal IMAP angeboten;
- aktuelle Agentur-Nachrichten (*Newsticker*), Berichte und Reportagen sowie Börsenkurse;
- Diskussionsforen und schwarze Bretter, evtl. mit Archiv und Volltextsuche über alle Foren und Bretter. Dabei gibt es sowohl eigene, anbieterspezifische Foren als auch die Integration der bestehenden Internet-*Newsgroups*;
- Herunterladen von Programmen (z.B. von *Shareware* und *Freeware*);
- simultane Nachrichten und Konferenzen (*Chat* und *Instant Messaging*);
- eigene, kostenlose *Homepages*.

Weitergehende, speziellere Funktionalität ist seltener in den großen horizontalen Portalen anzutreffen; dies ist eher die Domäne von vertikalen Portalen. Bei diesen läßt sich die Ausrichtung anhand der Zielgruppen unterscheiden, die sich durch den regionalen oder demographischen Fokus, Themen, Inhalte und Produkte und die anbietende Organisation ausdrücken. Die Funktionalität von zielgruppen- bzw. themenspezifischen Portalen ist meistens mit den horizontalen vergleichbar. Angebote zu fast allen denkbaren Themen existieren, diese reichen von speziellen Finanzdiensten über Informationen zu allen Lebenslagen bis hin zu esoterischen Themen. Auch die Universitäts-Bibliotheken erkennen den Nutzen von Portalen, um personalisierte Informationsangebote für ihre Nutzer zu schaffen; eine interessante neue Entwicklung in diesem Bereich sind z.B. die sogenannten Dissertationsportale, die Zugriff auf alle archivierten Dissertationen liefern sollen [Dob00]. Gerade im Bibliotheksbereich erleichtert die schon gute Erfassung von Metadaten die Strukturierung und Kategorisierung der Informationen.

In [Gol99] werden sechs verschiedene Arten von Kategorien von Funktionalität genannt, die von horizontalen Portalen abgedeckt werden, und die selektiv von vertikalen Portalen verwendet werden:

Context: Gemeint sind Suche, Navigation und Verzeichnisse für weiterführende Information. Der Kontext wird, nachdem zunächst das Motto „*Content is King*“ galt, inzwischen als noch wichtiger als die Inhalte betrachtet.

Content: Inhalte wie Artikel, Nachrichten etc., die das eigenständige Informationsangebot eines Portals ausmachen und nicht nur Verweise auf die Informationsangebote anderer Anbieter sind.

Commerce: Die Möglichkeit zum Online-Einkauf (*e-Commerce*) verschiedenster Dienstleistungen und Güter. Ein Beispiel für ein vertikales Portal, das sich auf diesen Aspekt konzentriert, ist AMAZON. Auch das *Online-Banking* und die Portfolioverwaltung für Börsenaktivitäten fallen darunter.

Communication: Das Angebot zur Kommunikation über e-Mail, *Chat* etc.

Connectivity: Das Angebot von eigenen Internet-Zugängen über verschiedene Techniken wie analoges Telefon, ISDN, DSL oder ADSL. Hier streben viele Portale bzw. Zugangsanbieter eine Kopplung zum Zwecke der besseren Kundenbindung an. Besonders ausgeprägt ist diese Erscheinung im Bereich der mobilen Kommunikation (Mobiltelefone, WAP etc.).

Communities: Unterstützung virtueller Gemeinschaften durch Diskussionsforen, *Chat*, *Homepages* etc.

Spezialisierte (vertikale) Portale bieten neuartige Dienstleistungen an, die häufig mit den Funktionen horizontaler Portale angereichert sind, um den Benutzern höheren Komfort zu verschaffen und die Kundenbindung zu verstärken. Beispiele sind Diskussionsforen und schwarze Bretter bei Online-Buchhändlern. Grundsätzlich gilt, daß vertikale Portale viele der Funktionen horizontaler Portale besitzen, und diese auf ihr spezielles Thema anwenden. Abhängig von der Spezialisierung besitzen sie weitere Funktionen, die horizontale Portale nicht besitzen. Dies sind beispielsweise folgende Funktionen:

- Einkauf (*Online-Shopping*);
- Analyse von Börsenkursen, Portfolioverwaltung für Aktien;
- Online-Auktionen;
- Ablagemöglichkeit für eigene Dateien und Bilder;
- Landkarten, Reise- und Routenplanung, Fahrpläne;
- Auskunftsdienste, z.B. Postleitzahlen, e-Mail-Adressen, Telefontarife.

Als weitere Art des vertikalen Portals tritt das Unternehmensportal auf, das als Thema das eigene Unternehmen und seine Produkte hat, und als Adressat sowohl die Öffentlichkeit (Medien, Investoren) als auch Kunden, Partner und die eigenen Mitarbeiter. Das Unternehmensportal ist als Synonym zum *Enterprise Portal* und *Corporate Portal* zu sehen. Die eingangs erwähnten *Consumer-Portale* stellen als Gegensatz dazu horizontale sowie themen- und zielgruppenspezifische vertikale Portale dar. Insgesamt ergibt sich damit die in Abbildung 3.4 dargestellte Taxonomie von Portalen.

Eine in dieses Schema passende Definition von Unternehmensportal ist die folgende, wieder [KLT00] entnommene:

> „Ein Unternehmensportal ist eine Web- oder Intranet- oder Extranet-*Site*, die unternehmensrelevante Informationen aus verschiedenen, ausgewählten Internet-Quellen, Datenbanken und anderen unternehmensspezifischen digitalen Quellen zusammenfasst und ihren Nutzern – einer eingeschränkten, mit dem Unternehmen verbundenen Zielgruppe – über einen Standard-Web-Browser oder spezielle Software einen personalisierten Zugang zu diesen Informationen mittels Suche und/oder Navigation von Verzeichnisstrukturen bietet, ergänzt um redaktionellen Inhalt, Funktionalität zur Kommunikation und Informationsverarbeitung.“

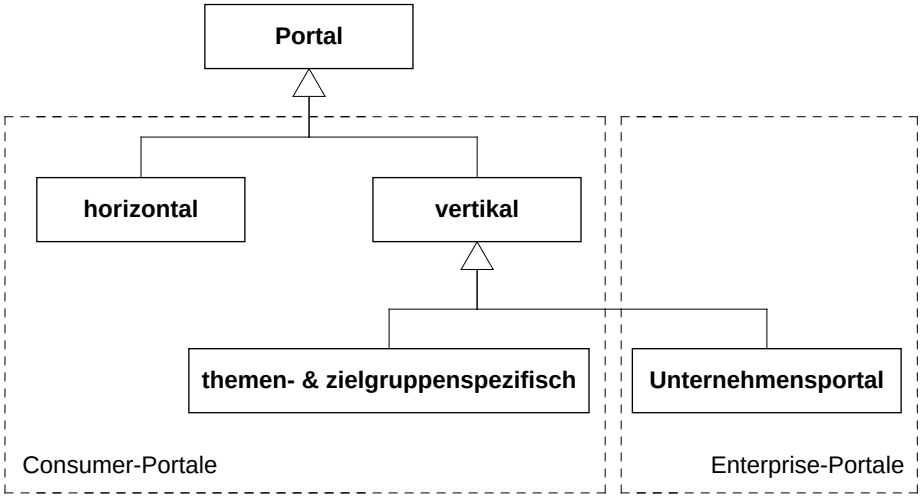

Abbildung 3.4: Taxonomie von Portalen

Eine weitere Differenzierung der Funktionalität von Unternehmensportalen stellt eine schwierige Aufgabe dar. Am sinnvollsten erscheint uns die Untergliederung anhand der Zielgruppe bzw. der beteiligten Parteien. Es gibt andere Ansätze, etwa den von [Ovu00] (gut dargestellt in [Win01]), der unter Portal öffentliche Web-Portale (horizontale und vertikale), Unternehmensportale, *Workspace*-Portale sowie spezialisierte Portale (etwa für ASP) und Marktplätze (B2B, B2C) subsumiert, und das Unternehmensportal nochmals in Partner-Portale, Intranet-Portale und *Web-Sites* differenziert, sowie das Wissensportal als Unterform des *Workspace*-Portals mit enger Verbindung zum Intranet-Portal sieht. Unsere Taxonomie fällt teilweise anders aus: Es lassen sich als Vereinigung und Integration der Ansätze von [KLT00, Mos00, Fau00, SS99, Bau01] folgende Bereiche erkennen:

1. Allgemeine Unternehmens-*Web-Sites*. Diese dienen der bloßen Information der Öffentlichkeit, z.B. von Medien, Investoren und Interessenten über das Unternehmen und seine Leistungen oder Produkte, etwa zum Zwecke des Marketings und der Öffentlichkeitsarbeit. Hier sind normalerweise keine personalisierten Informationen vorgesehen und der Zugang erfolgt immer über das Internet. Diese Art der Internetpräsenz bieten mittlerweile die meisten Unternehmen. Als Kürzel dafür schlägt [Bau01] B2P (*Business to Public*) vor.

2. Kundenportale. Die Zielgruppe sind Kunden – meistens Firmenkunden – des Unternehmens. Zweck des Portals ist die Unterstützung und Verbesserung der Verkaufsprozesse, gekoppelt mit einer engeren Kundenbindung durch die personalisierte Versorgung mit Informationen. Das Schlagwort dafür ist Kundenbeziehungsmanagement (*Customer Relationship Management*, CRM). Häufig wird zusätzlich dazu das elektronische Bestellwesen und die Warenwirtschaft angebunden. Die Orientierung der Zielgruppe kann auch auf Einzelkunden liegen, für die elektronische Bestellungen etc. möglich sind. Hier wird dann *e-Commerce* betrieben. Das gängige Kürzel für die Paradigmen in Kundenportalen lautet B2C (*Business to Consumer*). Bei Firmenkunden fällt allerdings die Abgrenzung zum B2B-Paradigma nicht leicht und ist oft willkürlich. Technisch gesehen sind Kundenportale meistens durch das Internet zu erreichen, in selteneren Fällen durch Extranets.

3. Partnerportale gehen in die Richtung von elektronisch unterstütztem Lieferketten-Management (*Supply Chain Management*, SCM), elektronischen Marktplätzen und Anwendungsvermietung (*Application Service Providing*, ASP). Die Zielgruppen sind die eigenen Mit-

arbeiter des Einkaufs oder Verkaufs, etablierte Geschäftspartner und Zulieferer. Gerade der elektronische Einkauf (*e-Procurement*) verspricht noch hohe Rationalisierungspotentiale und Einsparmöglichkeiten durch den automatisierten Austausch von Daten und Belegen. Das Paradigma dafür lautete B2B (*Business to Business*). Der Zugang erfolgt meistens durch Extranets.

4. Wissensportale dienen zum Wissensmanagement und sind daher für diese Arbeit von besonderem Interesse. Der Begriff *Enterprise Knowledge Portal* oder *Enterprise Expert Portal* findet sich in diesem Kontext häufiger [Fir00, SS99, MAB+99]. Letzteres dient zur Identifizierung von Experten (vgl. Abschnitt 2.4.2). Die Zielgruppe sind die eigenen Mitarbeiter, deshalb lautet das Paradigma B2E (*Bussiness to Employee*) bzw. E2E (*Employee to Employee*). Traditionell wichtige Nutzer von Wissensportalen sind Unternehmensberatungen, die in besonderem Maße auf den internen Wissensaustausch ihrer Berater angewiesen sind. Wissensportale werden entweder gänzlich im Intranet eines Unternehmens betrieben oder stellen vollkommen öffentliche Dienste dar und sind dann im Sinne der obigen Definitionen themenspezifische vertikale (*Consumer*-)Portale. In beiden Fällen ähneln sie von der Funktionalität her (Personalisierung, kollaborative Dienste wie Diskussionsforen) stark den *Consumer*-Portalen.

Die wesentlichen Merkmale der verschiedenen Typen von Unternehmensportal sind noch einmal in Tabelle 3.4 einander gegenübergestellt.

	WebSite	**Kundenportal**	**Partnerportal**	**Wissensportal**
Paradigma	B2P	B2C	B2B	B2E, E2E
Medium	Internet	Internet, (Extranet)	Extranet, (Internet)	Intranet, (Extranet)
Zweck	Marketing, allg. Information	CRM, e-Commerce	SCM, ASP, e-Procurement	Wissens-management
Zielgruppe	Öffentlichkeit	Firmenkunden, Einzelkunden	Einkauf/Verkauf, Geschäftspartner, Zulieferer	Mitarbeiter
Orientierung	extern			intern

Tabelle 3.4: Merkmale von Unternehmensportalen

Eine interessante Entwicklung, die den Anforderungen zumindest der Kunden-, Partner- und Wissensportale gerecht wird, ist die 1999 vorgestellte Lösung MYSAP.COM von der SAP AG [SAP01]. Sie umfaßt Komponenten für einen B2B-Marktplatz, einen elektronischen Arbeitsplatz für Mitarbeiter mit rollenbasiertem Zugang zu Anwendungen (R/3 und anderen), die Unterstützung für elektronischen Handel mit Transaktionen für Beschaffung und Verkauf, Kundenbeziehungsmanagement und Zulieferkettenmanagement, und ferner Möglichkeiten zum *Application Hosting* [Fau00]. Eine auf MYSAP.COM zugeschnittene Diskussion der für die Benutzer sichtbaren Funktionalität in [Wal01a] führt Nachrichten, Status, Organisation, Aktionen, Information, Dienste, Kommunikation und Gemeinschaften sowie Kollaboration als die wichtigsten enthaltenen Komponenten auf.

Eine noch weitergehende Differenzierung von Unternehmensportalen ist für den Rahmen dieser Arbeit nicht mehr gewinnbringend, außer im Bereich der Wissensportale. Die Funktionalität von

Wissensportalen läßt sich nach [MAB+99] in drei weitere Unteraspekte aufgliedern. Die dortige Unterscheidung von von *Consumer Portals* und *Enterprise Portals* weicht von der unsrigen (vgl. Abbildung 3.4) ab; das Unternehmensportal entspricht dort der unsrigen Definition von Wissensportal; ansonsten stimmen die Zielgruppen (Mitarbeiter) und Zwecke überein. Dies zeigt einmal mehr die disparate Begriffsbildung. Eine ähnliche Sichtweise wird in [IW01b, Tka99b] vertreten. Die drei Arten von Wissensportalen, die wir hier unterscheiden wollen, sind nun:

Enterprise Application Portals: Diese Anwendungsportale sind für den Einsatz im Umfeld von klassischen unternehmensweiten betrieblichen Informationssystemen (ERP-Systeme) konzipiert und haben das Ziel, den Mitarbeitern (Wissensarbeitern) personalisierten, rollenbasierten, uniformen und zentralen Zugriff auf alle vorhandenen Systeme und die darin vorhandenen Informationen und Funktionen zu geben. Der Leitgedanke ist, die bisher unverbunden vorhandenen Systeme unter einer einheitlichen, web-basierten Benutzungsoberfläche zu integrieren. Die Anmeldung soll nur einmal erfolgen und für alle Systeme gelten (*single sign on*). Besonders wichtig ist die Integration interner organisatorischer Dienste, die bisher ausschließlich von anderen Abteilungen geleistet wurden (z.B. Erfassung von Statusänderungen wie Umzug, Heirat etc. von der Personalabteilung, Reisekostenabrechnungen, Stundenerfassung etc.), die jetzt über das Portal von den Mitarbeitern selbst vorgenommen werden können sollen (*self service*).

Die Abgrenzung der Anwendungsportale (*Enterprise Application Portals*) von den Unternehmensportaltypen Partnerportal und Kundenportal ist in der Zielgruppe zu sehen: Hier handelt es sich lediglich um die Mitarbeiter, während dort sowohl (interne) Mitarbeiter als auch (externe) Kunden und Partner bzw. nur diese die Benutzer sind.

Der für das Wissensmanagement interessante Gedanke ist die Zusammenstellung aller von Wissensarbeitern benötigten Anwendungen und Funktionen unter einem Dach. Durch Verknüpfungsmöglichkeiten mit den Anteilen der Informationsportale und Expertenportale (s.u.) ergibt sich ein beträchtlicher zusätzlicher Mehrwert.

Enterprise Information Portals: Diese Art von Portal dient in erster Linie dazu, explizites Wissen in Form von Informationen und Inhalten aus verschiedenen Quellen wie dem WWW oder unternehmensinternen Datei-Servern, Nachrichten (*news*), Artikel, Berichte, interne Dokumente, Faxe, Dateien (ggf. aus Dokumentenmanagement-Systemen) zu erschließen und zu verwalten. Die nahtlose Integration und Präsentation sowohl von strukturierten Inhalten als auch unstrukturierten (oder semistrukturierten) Inhalten ist dazu nötig. Viele der dazu gebrauchten Funktionen sind in ähnlicher Form auch in den öffentlichen horizontalen Portalen zu finden. Wichtige Funktionen, die meistens darüber hinausgehen, sind die Kategorisierung und möglicherweise automatische Klassifikation der Informationen. Auch hier kommt ein personalisierter, rollenbasierter und uniformer web-basierter Zugang zum Einsatz.

Enterprise Expert Portals: Um das intellektuelle Kapital einer Organisation besser zu nutzen, muß neben dem durch die oben beschriebenen *Enterprise Information Portals* gut erschlossenen expliziten Wissen auch das implizite Wissen identifiziert werden. Das Expertenportal hat daher zum Ziel, Qualifikationen und Kenntnisse der Mitarbeiter zu erkennen und diese Informationen beispielsweise in Wissenslandkarten oder Verzeichnissen wie „gelben Seiten“ zur Verfügung zu stellen. Auch eine systematische Verwaltung der Fertigkeiten der Personen (*skill management*) kann hier stattfinden. Die Zugangstechnik ist dieselbe wie oben.

Ein Bereich, der hierzu gut passt und der in jüngster Zeit viel Aufmerksamkeit erhält, ist der des elektronisch unterstützten Lernens (*e-Learning*). Es existieren dafür bereits die ersten Portal-basierten Lösungen, die etwa virtuelle Schulungsräume mit Diskussionsforen und

Betreuung durch Tutoren etc., Lehrmaterialien und automatische Tests umfassen [Hyp01b]. Siehe dazu Abschnitt 2.4.3.

Für eine effektive Unterstützung von Wissensmanagementprozessen, wie sie in Kapitel 2 beschrieben ist, ist die Integration der Funktionen aller drei Arten in ein Portal nötig. Daher ist anzunehmen, daß eine Konvergenz dieser drei bislang häufig isoliert entwickelten Typen von Portalen stattfinden wird [MAB+99]. Dies darf sich nicht nur auf die Oberfläche, also die Präsentation und das Erscheinungsbild, und eine gemeinsame Administration der Benutzer und Rechte beschränken, sondern sollte dann eine vollständige Integration aller Funktionen und Informationen umfassen. Die Integration der Expertenfunktionen und der Funktionalität des Informationsportals stellt einen naheliegenden ersten Schritt dar; die Einbindung der Anwendungen den nächsten. Mögliche Integrationsstrategien sind in [MAB+99] detailliert beschrieben. Ansätze dafür, wie Anwendungs- und Informationsportale integriert werden können, gibt es zum Beispiel in Form des MYSAP.COM-Portals von SAP [SAP01, Fau00, SAP00], oder der FACTORY3-Portal-Suite von APPSOLUT, die eine entsprechende Integration verspricht [App00].

Die bis jetzt diskutierte Taxonomie von Unternehmensportalen und Wissensportalen ist noch einmal in Abbildung 3.5 wiedergegeben.

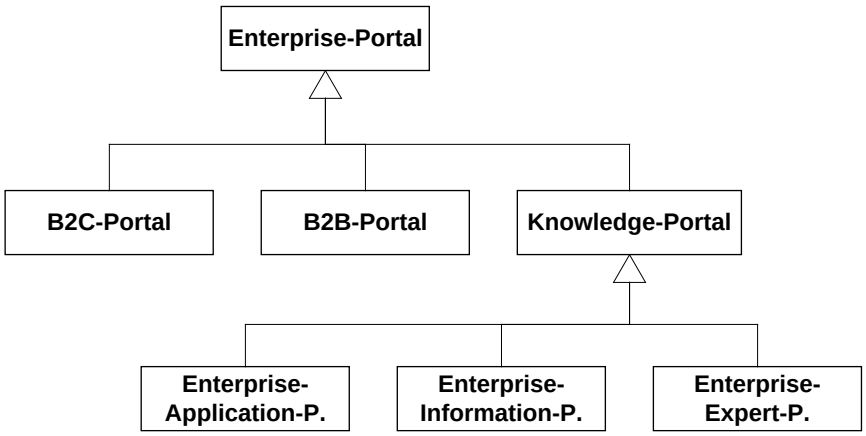

Abbildung 3.5: Taxonomie von Unternehmensportalen

Abschließend soll die wichtigste Funktionalität für Unternehmensportale und also auch Wissensportale kurz zusammengefasst werden, die [Del00b] sowie zum Teil [Del00a, Win01, App00] entnommen ist:

- Integration von bestehenden betrieblichen Informationssystemen, von Kommunikationssystemen (e-Mail etc.), von *Data Warehouses*, Dokumentenmanagement-Systemen, Dateisystemen sowie Workflow- und Groupware-Systemen. Zudem sollte die gewohnte Arbeitsweise mit Standard-Software-Produkten wie Textverarbeitungen weiterhin ermöglicht sowie hinreichend gut integriert werden, damit die Akzeptanz des Portals verbessert wird.
- Strukturierung und Klassifikation aller angebotenen Informationen zur Erleichterung des Zurechtfindens durch Bilden eines Kontextes. Dabei können Begriffstaxonomien und Wissenslandkarten [Tka99a] helfen. Eine mittlerweile immer wichtiger werdende Entwicklung ist die der automatischen Klassifikation, da der Umfang der Informationen ständig größer wird. Es sollten dafür sowohl manuelle als auch automatische Mechanismen vorhanden sein.

- Suchfunktionen sowohl über die Metadaten als auch die Volltexte aller vorhandenen Informationsobjekte. Eine Verknüpfung mit den Strukturierungsinformationen ist ebenfalls hilfreich. Die neusten Ansätze schließen Techniken des fallbasierten Schließens (*case based reasoning*, CBR), von natürlichsprachlichen Anfragesprachen oder der unscharfer Suche nach [Zad65] mit ein (*fuzzy search*).

- Publikation und Verteilung von neuen Informationen und Dokumenten. Dies fällt bereits stark in das Gebiet der Content-Management-Systeme, ist aber zur Wissensverteilung äußerst wichtig. Die in Abschnitt 3.3.2 aufgeführte Funktionalität muß dabei entsprechend vorhanden sein.

- Prozessunterstützung zur Abwicklung von Geschäftsprozessen, die durch das Portal ermöglicht werden, z.B. von *e-Commerce*, Kundenportal- oder Partnerportal-Funktionen. Eine durchgängige Unterstützung aller Schritte mit allen Informationen aus den verschiedenen Bereichen des Portals ist dazu notwendig. Die Publikations- und Freigabeprozesse im Bereich des Publizierens (vgl. oben) müssen ebenfalls durch dieselben Verfahren abgedeckt werden. Die Definition und Ausführung von ad-hoc-Prozessen sollte ebenfalls unterstützt werden.

- Kollaboration zwischen den beteiligten Personengruppen (Mitarbeitern, Kunden, Partnern etc.) über die Unterstützung bzw. Einbindung von Kommunikationsmitteln wie e-Mail, schwarzen Brettern, (moderierten) Diskussionsforen und Online-Konferenzen (via *Chat* oder *Instant Messages*) sowie Gruppen mit virtuellen Treffpunkten zum Dokumentenaustausch und Erstellen von Arbeitsplänen. Für verschiedene Bereiche (extern, unternehmensweit, projektgruppenbezogen, Interessengruppen) müssen diese Funktionen getrennt einsetzbar sein und die zugelassenen Personenkreise fein granular einstellbar sein. Ein durchgängiges Rechtekonzept ist wiederum Voraussetzung dafür.

- Personalisierung aller Inhalte und Einstellungen für jeden Benutzer ist eine wichtige Voraussetzung für erfolgreiches Arbeiten mit dem Portal. Die individuelle Zusammenstellung der wichtigsten gebrauchten Funktionen und Informationen sowie die Konfiguration des Erscheinungsbildes fällt darunter. Häufig wird die Metapher von *Informationskanälen* (*channels*) gebraucht, um die verschiedenen den persönlichen Interessen gemäßen Informationsangebote zu beschreiben, die gezielt abonniert werden können. Zusätzlich sollten persönliche Anmerkungen, Bewertungen und frei gestaltbare Verzeichnisse mit interessanten oder wichtigen Informationen das Angebot abrunden. Hinzu können Mechanismen zur Notifikation (via e-Mail, Listen) über neue oder geänderte relevante Inhalte kommen [Thu00].

- Präsentation der Inhalte in benutzer-, medien- und endgerätegerechter Art und Weise, d.h. das Angebot entsprechender Funktionalität in ausreichender Benutzbarkeit [Wal01c]. Farben und Layout spielen dabei eine wichtige Rolle, auch für das standardisierte Erscheinungsbild des Unternehmens. Bestimmte Funktionalitäten treten stets auf und damit bestimmte Arten von Seiten, die entsprechend wiedererkennbar gestaltet werden sollten [BLW01].

Orthogonal zu diesen acht Aspekten ist es nach [Del00b] wichtig, daß das Portalsystem ein lernendes System ist (dort als *Learning Loop* bezeichnet), das das Benutzerverhalten ständig analysiert und sich an die Bedürfnisse jedes Benutzers automatisch anpasst. Dazu ist die Messung des Verhaltens nötig (benutzte Bereiche, Suchanfragen, Klassifikation der Inhalte, Relevanz der Inhalte für verschiedene Benutzer etc.). Agententechnologie und neuronale Netze bieten beispielsweise Ansätze und Methoden dafür an.

Neben diesen funktionalen Aspekten kommen weitere, die Medien betreffende, hinzu. So sollen moderne Portalsysteme die Inhalte nicht nur in Form von HTML für Web-Browser zu Verfügung stellen, sondern genau so, wie es schon für Content-Management-Systeme gilt, eine Vielzahl von Endgeräten bedienen können, so etwa WAP-fähige Mobiltelefone und persönliche digitale Assistenten (PDA). Einige Portalserver ermöglichen dies bereits [IW01a]. Ein wichtiger Faktor mit großen technischen Auswirkungen ist ferner die Sicherheit: Sichere, abhörgeschützte und nicht manipulierbare Verbindungen sowie das Signieren von Dokumenten zum Beweisen der Authentizität sind die wichtigsten Anforderungen dabei.

Es gibt mittlerweile eine größere Anzahl von Produkten verschiedener Hersteller zum Betreiben von Portalen. Ein eindeutiger Marktführer ist noch nicht auszumachen [IW01b]. Die Anbieter kommen aus unterschiedlichen Bereichen, z.B. Datenbanken, Wissensmanagement-Systemen, *Business-Intelligence*, betrieblichen Informationssystemen, oder sie positionieren sich als dedizierte Hersteller von Portalsoftware. Bekannte Firmen in diesen Bereichen sind IBM bzw. LOTUS, VERITY, AUTONOMY, HYPERWAVE, OPENTEXT, FILENET, TIBCO, PLUMTREE, VIADOR, HUMINGBIRD und mittlerweile auch SAP.

3.3.4 Bewertung: DMS, CMS und Portale als Wissensmanagement-Systeme

Im Gegensatz zu den Systemklassen der Dokumentenmanagement- und Content-Management-Systeme, die zwar ein breit definiertes Spektrum haben, aber für die sich doch gute Abgrenzungen finden lassen, ist der Begriff des Wissensmanagement-Systems (WMS) bzw. des OMIS nicht ohne weiteres einheitlich faßbar. Wie bereits in Abschnitt 3.2 dargelegt, finden zahlreiche Systemklassen Verwendung in Wissensmanagement-Systemen. In der Literatur finden sich daher viele Sammlungen von Fallbeispielen, die jeweils an den speziellen Bedürfnissen der beteiligten Organisationen orientierte Lösungen implementiert haben, und die sich demgemäß stark unterscheiden. Beispiele finden sich etwa in [BÖV00, Seg00, Gen99, BVÖ99, Gro01, Gro00a, AS99, BP98a, KO96, Sch00b, BMS01, Wag99].

Als schon längere Zeit eigenständiger Begriff, der allerdings eine Abgrenzung zu Wissensmanagement-Systemen oft vermissen läßt, ist der des *Organizational Memory Information Systems* (OMIS) in Gebrauch. In [SAD99] wird es folgendermaßen definiert:

> „Ein OMIS entsteht durch die Integration von Basistechniken zu einem Computersystem, das in der Organisation Wissen und Informationen fortlaufend sammelt, aktualisiert, strukturiert und für verschiedene Aufgaben möglichst kontextabhängig, gezielt und aktiv zur Verbesserung des kooperativen Arbeitens zur Verfügung stellt."

Ein fachübergreifendes Konzept dafür hat sich bislang nicht durchgesetzt [Hab99]; allen Konzepten und Definitionen gemeinsam bleiben allerdings die Metaphern des Gedächtnisses und des Lernens. Ebenso heterogen wie bei Lösungen, die explizit für das Wissensmanagement gedacht sind, aber nicht als OMIS verstanden werden, präsentiert sich die Landschaft der für OMIS verwendeten Systeme. Sie reicht von Erfahrungsdatenbanken, *Data-Warehouse*-Systemen, Modelldatenbanken, Hypertext-Lösungen und Intranets, Wissenslandkarten bis hin zu Systemen der künstlichen Intelligenz [Hab99, Gaß99, GFS98, KA98].

Da sich, wenn man der gängigen Praxis und dem Sprachgebrauch folgt, keine weiteren und guten Kriterien zur Unterscheidung von Wissensmanagement-System und OMIS finden lassen, wollen wir ein OMIS als die zentrale Komponente der *technischen Infrastruktur* für ein Wissensmanagement-System sehen. Gängigen Definitionen von kooperativen Informationssystemen folgend [DDJ+97] umfaßt ein kooperatives Informationssystem (also auch jedes Wissensmanage-

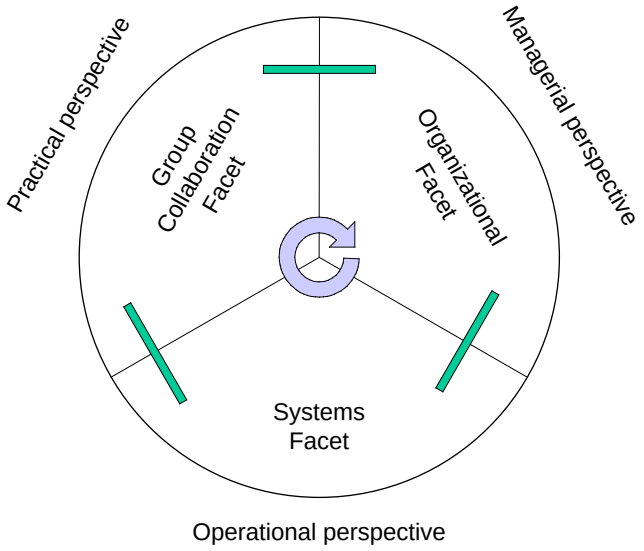

Abbildung 3.6: Drei Facetten kooperativer Informationssysteme

ment-System, denn diese Art von Informationssystem ist inhärent kooperativ) nicht nur die systemtechnischen Aspekte, sondern genauso die organisatorischen und praktischen, kollaborativen Aspekte wie Workflow für Geschäftsprozesse. Dies wird in [DDJ+98] durch die drei Facetten *Systems*, *Group Collaboration* und *Organization*, die die operationale, die praktische und die Manager-Perspektiven vertreten, zum Ausdruck gebracht (siehe Abbildung 3.6). Dieser Sichtweise folgend, kann man OMIS als die System-Facette von Wissensmanagement-Systemen begreifen. Erst zusammen mit der ganzheitlichen Betrachtung der anderen Facetten kann ein eigentliches, wirksames Wissensmanagement-System etabliert werden. Diese Einordnung paßt gut zur Sichtweise von OMIS als zentralem Konzept der informationstechnischen Unterstützung des Wissensmanagements [GFS98]. In diesem Sinne sind Dokumentenmanagement-Systeme, Content-Management-Systeme und viele Arten von Portalen, insbesondere manche der Unternehmensportale (z.B. Wissensportale) auch OMIS oder zumindest Bausteine davon.

Abschließend soll ein integrativer Ansatz zum Verständnis der Zusammenhänge und zur Nutzung der Synergien der in den vorangehenden Abschnitten besprochenen Dokumentenmanagement-Systemen, Content-Management-Systemen und Wissensmanagementsystemen bzw. OMIS vorgestellt werden. Der Ansatz ist in Abbildung 3.7 illustriert. Er basiert auf der zentralen Beobachtung aus Abschnitt 3.3.3, daß Portale und insbesondere die hier im Mittelpunkt stehenden Unternehmensportale uniformen, personalisierten Zugang über Internet-Technologien zu Informationen für verschiedene Benutzerkreise anbieten. Benutzer sind hier sowohl die außerhalb des Unternehmens stehenden Konsumenten, die Medien, Kunden und Geschäftspartner sowie Investoren, als auch die innerhalb der Organisation befindlichen Mitarbeiter. Die externen Personen greifen über Intra- oder Extranet auf für sie vorgehaltene allgemeine oder personalisierte Informationen und Funktionalitäten wie beispielsweise betriebliche Informationen wie Liefer- und Bestellwesen (ERP, SCM, CRM oder *Shops*) zu und können die vom Portal bereitgestellten Kommunikations- und Kollaborationsmöglichkeiten (e-Mail, Diskussionen, Benachrichtigungen etc.) nutzen. Auf der anderen Seite stellt das Portal die Benutzungsoberfläche des OMIS für die internen Nutzer dar. Gemäß der oben getroffenen Definition von OMIS als der systemtechnischen Facette des Gesamt-Wissensmanagement-Systems umfaßt das letztere die Personen, ihre organisatorische Einbettung in Aufbau- und Ablaufstruktur der Organisation sowie die system-

technische Unterstützung durch das OMIS über das Intranet-Portal, das Zugang zu allen unternehmensrelevanten Informationen, Funktionen und Vorgängen bietet. Dazu gehört einerseits die Unterstützung der Wahrnehmung der jeweiligen internen Aufgaben bezüglich der extern angebotenen Funktionalität (Geschäftsprozesse, Kommunikation etc.), und andererseits die Nutzung des OMIS als unternehmensweites Gedächtnis, also als Wissens-Werkzeug gemäß der in Abschnitt 2.4 beschriebenen Teilaufgaben. Eine essentielle Anforderung besteht dabei darin, beide Aspekte nahtlos zu integrieren um so den wesentlichen Mehrwert des Portals (Uniformität, Integration) ausnutzen zu können. Wesentliche Unterstützung erfährt der interne Nutzer hierbei durch die Nutzung von Content-Management-Funktionalität (Redaktion, Lebenszyklus des Inhalts, Publikation), um qualitativ hochwertige Informationsartefakte sowohl für die interne Nutzung als auch die externe Präsentation zu erstellen. Weiterhin basiert diese Unterstützung auf Dokumenten-Management-Funktionalität, die auch ohne Nutzung der Content-Management-Funktionen zum Archivieren, Indizieren, Suchen, Verteilen und kooperativen Nutzen von Informationen gebraucht wird und so orthogonal zum Verwalten aller Informationsartefakte der Organisation (also des OM) Verwendung findet. Um performante und effiziente Systeme skalierbar betreiben zu können, greifen alle Komponenten auf zumeist relationale oder objektorientierte Datenbanken als Informationsspeicher zurück, was als Kern in der Abbildung 3.7 dargestellt ist.

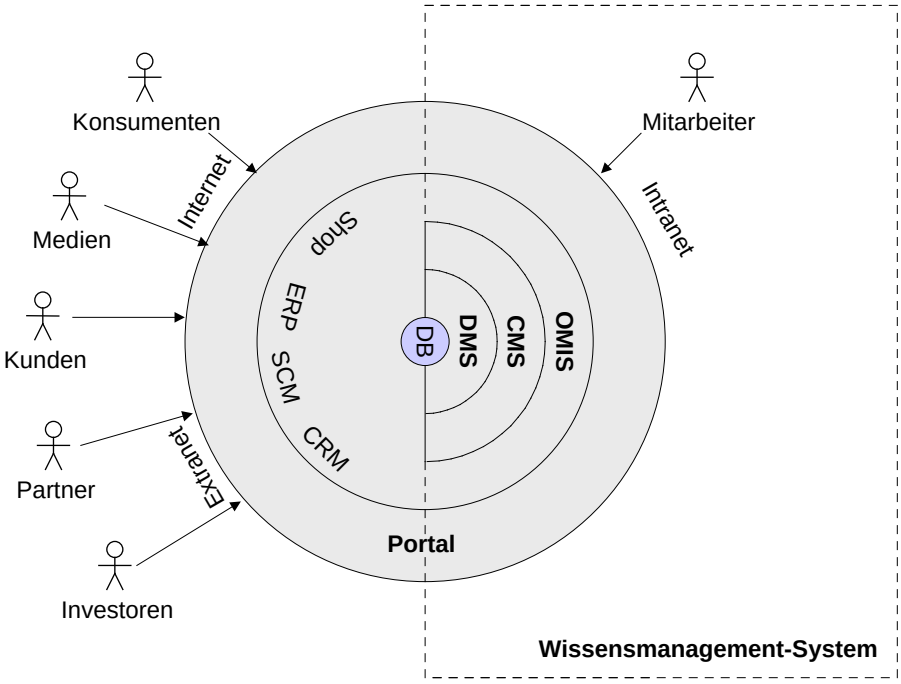

Abbildung 3.7: Portale als Wissensmanagement-Systeme

Um eine bruchlose Verwendung aller Systemkomponenten zu gewährleisten, muß eine umfassende Integration der Funktionalität (Benutzerverwaltung und Berechtigungskonzepten, einheitlicher Präsentation, orthogonaler Nutzung von Informationsartefakten) stattfinden [MAB+99].

Je nach konkretem Einsatzzweck und Unternehmensziel sind unterschiedliche Schwerpunkte zu setzen; so mag in einigen Fällen die externen Benutzern angebotene Funktionalität geringer ausfallen oder ganz entfallen, ohne das Wesen des Wissensmanagement-Systems nachhaltig zu beeinträchtigen. Wenn allerdings etwa elektronischer Handel oder das Angebot von Dienstleistungen über das Internet eine Rolle spielen, so bringt die Synergie von OMIS und den benötigten internen Funktionen im Vergleich zu isolierten Lösungen erhebliche Produktivitätsgewinne.

3.4 Ausgewählte Produkte

Nachdem eben die relevanten Systemklassen analysiert, klassifiziert und ein Integrationsmodell für Datenbanken, DMS, CMS, OMIS und Portale im Kontext des Wissensmanagements entwickelt wurde, sollen nun, bevor im folgenden Kapitel 4 systematisch ein Anforderungskatalog an Portale für das Wissensmanagement entwickelt wird, einige wichtige oder besonders interessante kommerzielle Systeme, die sich in den Bereich der Portale und des Wissensmanagements einordnen lassen, vorgestellt und näher untersucht werden. Das grundsätzliche Ziel ist dabei, zu analysieren, welche Modelle von den bislang erfolgreichsten und umfassendsten Produkten eingesetzt werden, nachdem der vorangegangene Abschnitt schwerpunktmäßig die generelle Funktionalität von Portalen, CMS und DMS untersucht hat.

Die Fragestellungen dieser Untersuchungen sind dabei unter anderem die folgenden:

- Die wichtigsten Konzepte (beispielsweise das Modell der behandelten Informationsartefakte bzw. der Dokumente) sollen analysiert und in Beziehung zu den oben bereits definierten gesetzt werden.
- Die für das Wissensmanagement wichtigen Funktionen sollen identifiziert und dargestellt werden.
- Die vorgesehenen Einsatzszenarien und Anwendungsfälle im Kontext von unternehmensweitem Wissensmanagement und dem Betrieb von Unternehmensportalen sollen vorgestellt werden.
- Architektur, Komponenten und Konfiguration der Systeme sollen dargestellt, untersucht und verglichen werden.

Weiterhin dient die Betrachtung aus bestimmten Gründen besonders interessanter Systeme einerseits dazu, die in den vorhergehenden Abschnitten erarbeiteten Definitionen und beschreibungen der Funktionalität für Portalsoftware im Wissensmanagement zu erhärten und zugleich Anregungen für den Anforderungskatalog zu erhalten, und andererseits dazu, die Konzepte und Architekturen zu analysieren und zu bewerten. Diese Analyse soll aufzeigen, inwieweit es es bereits abstrakte Modelle und softwaretechnische Architekturen gibt, die die eine bruchlose Unterstützung der Aufgaben und Prozesse des Wissensmanagements gestatten.

Betrachtet werden in den folgenden Abschnitten der MICROSOFT SHAREPOINT SERVER (Abschnitt 3.4.1), der HYPERWAVE INFORMATION SERVER und das HYPERWAVE INFORMATION PORTAL (Abschnitt 3.4.2), sowie der LOTUS DISCOVERY SERVER und die LOTUS K-STATION (Abschnitt 3.4.3). Die drei ausgewählten Systeme sind aus den folgenden Gründen besonders interessant:

1. Der MICROSOFT SHAREPOINT PORTAL SERVER, weil diese Technologie vom unbestrittenen Marktführer für Betriebssysteme im unteren Segment (der *Personal Computer*) kommt, und da die derzeitigen Bestrebungen von MICROSOFT stark in die Richtung der höheren Segmente (Server-Bereich) gehen. Bislang haben sich die Entwicklungen von MICROSOFT meist als sehr erfolgreich erwiesen, man betrachte etwa den Erfolg des INTERNET EXPLORER und der damit verbundenen und in letzter Zeit erheblich verstärkten Strategie der Nutzbarmachung von Internet-Diensten für eigene, neue Geschäftsmodelle. Die zur Zeit heftig diskutierte neue Version WINDOWS XP mit den dort vorgesehenen Möglichkeiten zur Lizensierung und integrierten Nutzung von im Internet durch MICROSOFT bereitgestellte Dienste (z.B. *Passport* zur Autorisierung und zur Verwaltung von persönlichen Pro-

filen) erhärten diese Argumente. Eine nicht weniger interessante Entwicklung stellt das Angebot eines eigenen Content-Management-Servers im Server-Bereich dar. Diese Entwicklungen werden vermutlich noch erheblichen Konkurrenzdruck für die etablierten Anbieter erzeugen. Weitere Anzeichen dafür sind in der gerade beginnenden DOT.NET-Initiative zu sehen. Inhaltlich interessant ist der SHAREPOINT-SERVER, weil es sich um eine Portallösung handelt, die im Bereich Unternehmensportale, Content-Management, Dokumentenmanagement und Wissensmanagement anzusiedeln ist, und so ein breites Spektrum an Funktionen abdeckt, das vergleichbar mit dem Integrationsmodell aus Abschnitt 3.3.4 ist.

2. Der HYPERVAWE INFORMATION SERVER und das zugehörige, damit realisierte HYPERWAVE INFORMATION PORTAL stellte beim Erscheinen eine stark beachtete innovative Lösung dar, die in kurzer Zeit große Erfolge vorzuweisen hatte. HYPERWAVE gehört heutzutage zu den wichtigsten Anbietern im Portalmarkt und hat eine deutliche Ausrichtung auf Unternehmensportale und Wissensmanagement. Was das System sehr interessant in diesem Kontext macht, ist die Tatsache, daß die konzeptuellen Wurzeln aus dem akademischen Umfeld der Hypermedia-Systeme stammen. Hier ist also mit interessanten Konzepten und Modellen zu rechnen.

3. Die LOTUS K-STATION und der sie unterstützende LOTUS DISCOVERY SERVER stellen eine lange Zeit mit größter Spannung erwartete integrierte Portallösung für das Wissensmanagement dar, deren Erscheinungsdatum häufig verschoben wurde. Da sich LOTUS zuletzt stark als Anbieter von Wissensmanagement-Lösungen positioniert hat, mit LOTUS NOTES und den darauf basierenden Lösungen bereits große Erfahrungen und viele Kunden in diesem Gebiet hat und zahlreiche innovative Ansätze (z.B. zur automatischen Klassifikation) angekündigt wurden, ist das früher unter dem Codenamen *Raven* firmierende Projekt außerordentlich interessant.

Es gibt zahlreiche weitere Systeme, die mehr oder weniger gut in das Thema der Portale und des Wissensmanagements passen. Da der Fokus dieser Arbeit aber nicht auf einem systematischen Vergleich aller oder möglichst vieler Systeme liegt, sei auf entsprechende andere Publikationen verwiesen [BBM01, Ver00b, Gol99, SW00b, BWP97, Ovu00, MAB+99, Kes00, HV98, GFS98, WS98, ADI+00, Met00].

3.4.1 Microsoft Sharepoint Portal Server

Der MICROSOFT SHAREPOINT PORTAL SERVER verspricht Wissensarbeitern, die bisher mit der Standard-Bürosoftware von MICROSOFT arbeiten, zusätzliche, eng integrierte neue Arten zum Organisieren, Nutzen und Verteilen von Informationen. Es lassen sich damit Unternehmensportale, Dokumentenmanagement-Lösungen, sogenanntes *enterprise content indexing* sowie Unterstützung für kollaborative Gruppenarbeit verwirklichen [Sha01].

Wie der Name bereits andeutet, handelt es sich um ein Client/Server-System, wobei der Server als HTTP-Server (Web-Server im Intranet) fungiert und eine oder mehrere *Web-Sites*, die sogenannten *Digital Dashboards* für HTTP-Clients (Web-Browser) anbietet, die auch als Portale betrachtet werden können. Dabei wird ein allgemeines, voreingestelltes und vorkonfiguriertes *Dashboard* mit ausgeliefert, das unternehmensweit benutzbar sein soll; aber auch die Erstellung und der gleichzeitige Betrieb mehrerer weiterer, individueller *Dashboards* ist möglich, etwa für spezielle Belange von Projektteams. Diese Portale sind dann in weiten Grenzen personalisierbar und anpaßbar. Es lassen sich verschiedene wiederverwendbare, konfigurierbare Komponenten, die sogenannten *Web Parts* installieren, die Inhalte aus verschiedenen Quellen (Dokumente, *Web-Sites*, weitere Dienste etc.) anzeigen und bearbeiten können.

Folgende Funktionen bietet das Portal an [Höh01, Sha01, Mic01, Met01b]:

1. Wichtige Dokumentenmanagement-Funktionen, die über die von anderen MICROSOFT-Produkten wie MS OFFICE und MS WINDOWS angebotenen Möglichkeiten (Netzwerk-Dateisysteme, Freigaben, Dokumenteigenschaften) weit hinaus gehen:

   - Ablage von unterschiedlichen Versionen von Dokumenten und Verfolgung der Historie inklusive Sperrmechanismen mit Ein- und Auschecken der Dokumente.
   - Ablage von weiteren Informationen wie internen und externen Web-Seiten und Verknüpfungen (Links) darauf, Einbindung des Inhalts von Dateisystemen, Web-Servern, Datenbanken und Kommunikationswerkzeugen (*content sources*).
   - Verwaltung von aussagekräftigen, zusätzlichen und frei definierbaren Metadaten für die gespeicherten und eingebundenen Inhalte.
   - Klassifikation von Inhalten über frei erstellbare, hierarchische Kategorienschemata; die automatische Kategorisierung von Dokumenten wird dabei ebenfalls unterstützt.
   - Suche über den Volltext aller Inhalte sowie über Metadaten und Kategorien; dabei wird die Relevanz der Treffer benutzerspezifisch errechnet. Die Suche erstreckt sich dabei über alle eingebundenen Quellen.
   - Subskriptionen und automatische Benachrichtigungen für und über neue und geänderte Inhalte.
   - Rollenbasiertes Berechtigungskonzept zur Steuerung der Zugriffe der Benutzer und Benutzergruppen, das auf den WINDOWS NT-Autorisations- und Authentifikationsfunktionen aufbaut.

2. Zusätzlich zu den als Dokumentenmanagement-Funktionen zu bezeichnenden kommen zahlreiche weitere Funktionen hinzu, die gemäß der Diskussion in Abschnitt 3.3.2 eher den Content-Management-Systemen zuzurechnen sind. Die wichtigsten sind:

   - Unterstützung der Unterscheidung zwischen privatem und öffentlichem Status eines jeden Dokuments; dabei sind nur die als öffentlich markierten Dokumente für andere Benutzer sichtbar. Der Wechsel des Status von privat nach öffentlich ist als Freigabe zu verstehen.
   - Freigabemechanismen für zu publizierende Dokumente, die sicherstellen, daß vor der Freigabe eines Dokuments dieses von weiteren Personen begutachtet und genehmigt wird. Hier sind verschiedene – fest vorgegebene – Strategien möglich: Einerseits die Freigabe durch mehrere Personen nacheinander (*serial approval*), andererseits durch mehrere parallel (*parallel approval*), wobei dann noch die Entscheidung zwischen Freigabe durch alle (*all vote*) oder durch mindestens eine Person (*one vote*) möglich ist. Für diese Prozesse hilfreich ist die Einführung von Rollen mit rollenspezifischen Rechten für Autoren, Leser und Koordinatoren.
   - Integrierte Diskussionsforen, die sich auf jeweils ein Dokument beziehen. Dies erleichtert das Führen von Diskussionen in der Gruppe insbesondere über den Publikationsprozess und vermeidet den dafür eher umständlichen Austausch über e-Mail. Kommentare und Antworten darauf sind so jederzeit an genau einer Stelle für alle Personen verfügbar.

3. Die Arbeit mit dem Portal wird zudem dadurch erleichtert, daß Erweiterungen für die OFFICE XP-Produkte und den WINDOWS-EXPLORER existieren, die das direkte Einbringen von Inhalten und Änderungen in den Server ermöglichen. So lassen sich beispielsweise alle Dokumente und die gesamte Hierarchie der Kategorien im WINDOWS-EXPLORER als Netzlaufwerk-Dateisystem darstellen [Höh01]).

Die Gestaltung der HTML-Oberfläche des Portals folgt dem inzwischen gängigen Trend zur Aufteilung in mehrere, unabhängig voneinander zu positionierende und durch den Benutzer anpaßbare Unterbereiche, die jeweils eine bestimmte Informationsquelle oder Funktion bereitstellen. Obwohl dieses Konzept mittlerweile von vielen Portalen verwendet wird, hat sich für diese Bereiche noch kein weithin akzeptierter Name durchgesetzt; Bezeichnungen wie *Track*, *Area*, Fenster, Informationsblöcke, *Portlet* etc. werden verwendet. Häufig werden diese Bereiche graphisch so ähnlich wie die Fenster der graphischen Benutzungsoberfläche des Betriebssystems oder des Web-Browsers gestaltet. In [Bau01] finden sich einige Hinweis zur Gestaltung dieser Elemente.

Die Architektur des MICROSOFT SHAREPOINT PORTAL SERVER ist die eines klassischen dreischichtigen Client/Server-Systems [MW00]; die Benutzer greifen mit Web-Browsern, direkt über OFFICE-Produkte oder durch Sonderfunktionen des WINDOWS-EXPLORER über standardisierte Internet-Protokolle (HTTP etc.) auf den Portal-Server zu. Dieser besteht aus mehreren Komponenten und Schichten und ist in Abbildung 3.8 illustriert, die aus [Sha01] leicht verändert übernommen wurde.

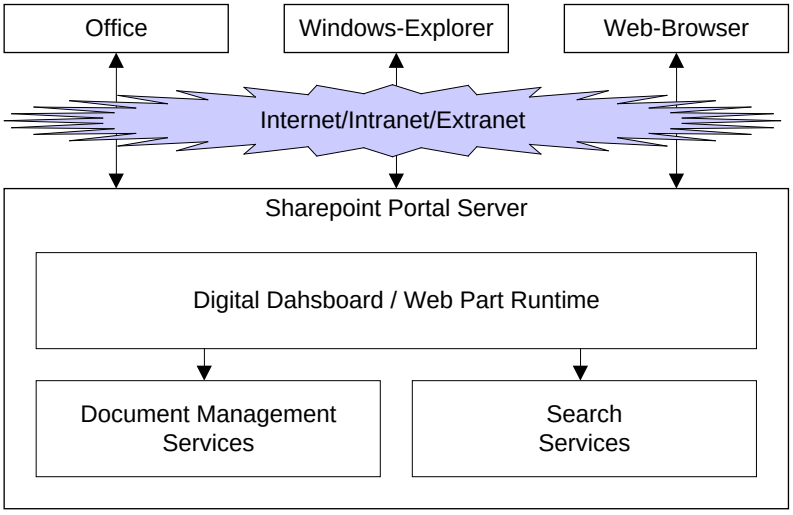

Abbildung 3.8: Architektur des MICROSOFT SHAREPOINT PORTAL SERVER

Das DIGITAL DASHBOARD und die *Web Part Runtime* bildet dabei die Anwendungsschicht des Portal-Servers, die dafür verantwortlich ist, einerseits Anfragen der o.g. Erweiterungen abzuwickeln und andererseits die HTML-Oberfläche der DASHBOARDS dynamisch zu generieren und zu personalisieren. Die Anwendungsschicht nutzt dabei den MICROSOFT INTERNET INFORMATION SERVER (IIS). Zur Bewältigung seiner Aufgaben greift die Anwendungsschicht auf weitere Dienste innerhalb des Servers zurück: Dienste zum Dokumentenmanagement und Dienste zum Indexieren und Suchen. Der *Document Management Service* basiert auf dem *Microsoft Web Storage System* und stellt Dokumentenmanagement-Funktionen wie Versionierung, Zugriffskontrolle etc. bereit.

Der *Search Service* erfüllt umfangreiche Aufgaben zur Indexierung von Inhalten und der Unterstützung der Suche und besteht aus den folgenden Unterkomponenten:

- Einer Komponente zum Zusammentragen aller Inhalte aus verschiedenen Quellen zum Zwecke der Indexierung (*Crawler*): In Frage kommen die Inhalte der verschiedenen vom Server verwalteten *Dashboard-Workspaces*, der Inhalte von Web-Servern, Datei-Servern, Kommunikationsservern (z.B. MICROSOFT EXCHANGE) und LOTUS NOTES-Datenbanken.

Mithilfe eines separat erhältlichen Entwicklungswerkzeugkastens (SDK) lassen sich Anbindungen für eigene, neue Quellen erstellen.

- Einer Komponente zum Erschließen der von der vorigen Komponente gelieferten Inhalte; dabei werden Metadaten und textuelle Inhalte aus den Quellen herausgefiltert. Es existieren bereits Filter für die gängigen Dokumentarten wie HTML, Textdateien und Bürodokumente. Auch hier ist es Programmierern möglich, neue Filter für weitere Dokumenten-Formate zu erstellen.

- Einer Komponente zum Einfügen der zuvor erschlossenen Informationen in den Index des Dienstes. Dazu werden sprachspezifische Module verwendet, die irrelevante Füllwörter erkennen und entfernen (*Stopwords*) und Normalformen von flektierten Formen bilden können (*Stemmer*).

- Einer Komponente zum Durchführen der eigentlichen Suchanfragen. Diese unterstützt zusätzlich WEBDAV-Anfragen.

Aufgrund des relativ breiten Anwendungsspektrums durch die Kombination von Dokumentenmanagement- und Content-Management-Funktionalität in Verbindung mit einem personalisierbaren Portal und der Integration vorhandener Server-Dienste sind sehr viele Einsatzszenarien und Schwerpunkte der Nutzung des Portals möglich. Der Portal-Server gestattet daher die Einstellung verschiedenster Konfigurationen insbesondere in Abhängigkeit vom Umfang der zu speichernden und zu indizierenden Informationsmengen durch einen hohen Grad an Skalierbarkeit:

- Es lassen sich je nach Anforderungsprofil (Last durch die Benutzer, Arten der Nutzung etc.) mehrere Portal-Server betreiben, die dann für unterschiedliche Aufgaben zur Verfügung stehen; beispielsweise zum Betreiben der *Dashboards* oder zum Durchführen der Suchanfragen. Werden diese Server auf verschiedenen Rechnern betrieben, entlastet dies den Server insgesamt und erlaubt höhere Durchsätze.

- Zusätzlich dazu lassen sich weiterhin die Komponenten, die sich in Abbildung 3.8 innerhalb des Portal-Servers befinden, auf verschiedene Rechner verteilen, um ebenfalls die Last besser aufzuteilen. Dies gilt auch für die Teilkomponenten des *Search Service*, so kann beispielsweise die Indizierung auf anderen Rechnern stattfinden als die Auswertung der Suche.

- Schließlich können die vom Portal-Server verwendeten Dateisysteme und sekundäre Web-Server auf weiteren Rechnern betrieben werden, um den Portalserver von diesen zusätzlichen Aufgaben zu entlasten.

Konzeptionell schwach ausgeprägt ist der Begriff des Dokuments als zentrales Informationsartefakt des Portals. Die traditionelle Sichtweise der Datei, wie sie aus dem Arbeiten mit der Oberfläche des Betriebssystems per WINDOWS-EXPLORER und den Büroanwendungen gewohnt ist, wird in das Portal getragen, wobei die Integration weiterer Metadaten geschieht. Eine weitere Differenzierung des Dokumentenbegriffs, wie sie bei modernen Content-Management-Systemen durch die Einführung von teilweise frei definierbaren Dokumententypen vorgenommen wird (vgl. Abschnitt 3.3.2), findet nicht statt. Lediglich die Definition und Nutzung von zusätzlichen Metainformationen gestattet eine etwas weitergehende Modellierung. Die im Portal vorhandenen Artefakte (Kategorien, *Quicklinks*, Nachrichten, Ankündigungen, Abonnements) und ihre Struktur sind fest vorgegeben.

Insgesamt gesehen ist also mit dem SHAREPOINT PORTAL SERVER eine interessante Synthese einer Dokumentenmanagement-Lösung mit einer Content-Management-Lösung unter dem Dach eines Intranet-Portals entstanden. Die Integration insbesondere der von MICROSOFT selbst angebotenen sogenannten *Back-Office*-Funktionalität wie e-Mail- und Kommunikationsserver (EXCHANGE SERVER, LOTUS NOTES), Web-Server und weiterer Dienste verspricht in Verbindung mit der nahtlosen Einbindung der *Front-Office*-Anwendungen (Textverarbeitung, Tabellenkalkulation etc.), der Web-Browser und des WINDOW-EXPLORERS eine durchgängige und damit gut benutzbare Lösung zu werden. Problematisch kann das Gesamtszenario dadurch werden, daß gerade auf der Serverseite die Anzahl, die Komplexität und die untereinander vorhandenen Abhängigkeiten der zu konfigurierenden Dienste und Komponenten sehr groß sind und nur noch schwer beherrschbar sind.

3.4.2 Hyperwave Information Server und -Portal

Die HYPERWAVE AG [Hyp01a] ist einer der führenden Anbieter im Bereich von Knowledge-Management-Lösungen und Wissensportalen (EIP) am internationalen Markt [Met01a] und bietet im wesentlichen drei Produkte an: Den HYPERWAVE INFORMATION SERVER (HIS), das als Anwendung darauf basierende HYPERWAVE INFORMATION PORTAL (HIP) sowie ferner die HYPERWAVE ELEARNING SUITE, die ebenfalls auf dem HIS basiert und als virtuelles Schulungszentrum konzipiert ist [Hyp01b], aber hier wegen ihrer speziellen Zielsetzung nicht weiter betrachtet wird.

Obwohl das Unternehmen HYPERWAVE erst um 1997 gegründet wurde, sind doch die Wurzeln der HYPERWAVE-Systeme älter. Sie stammen von dem etwa seit 1990 in Graz entwickelten HYPER-G-System[7], einem Hypermedia-System, über das es zahlreiche Veröffentlichungen sowohl der beteiligten Entwickler selbst (etwa [Mau96, AKM95, Kap95, KAM95, Kap91]) als auch von dritter Seite gibt [DH96, DH95]. In [Sch97a] findet sich eine konzentrierte Zusammenfassung der Eigenschaften sowie Vergleiche mit anderen verteilten Hypertextsystemen.

Im folgenden werden zunächst das HYPER-G-System etwas genauer vorgestellt und dessen Konzepte analysiert. Dies ist sinnvoll, da die HYPERWAVE-Systeme immer noch weitgehend darauf basieren. Zudem sind die Entwicklungsgeschichte und einige Eigenschaften des neuen Systems so besser verständlich. Anschließend werden Modelle und Konzepte der neueren HYPERWAVE-Systeme dargestellt und untersucht, um daraufhin die Funktionalität des Portals zu beschreiben. Diese beiden Abschnitte bilden den Kern der Untersuchung von Modell und Funktionalität. Anschließend wird die Systemarchitektur unter besonderer Berücksichtigung der Skalierbarkeit und der Erweiterbarkeit der Dokumentenklassen diskutiert. Eine Bewertung der Eigenschaften des Systems beendet den Abschnitt.

3.4.2.1 Hyper-G

Die ursprüngliche Intention des HYPER-G-Systems war die eines Hypermedia-Systems, das Zugriff auf alle mögliche Arten von Informationen bieten sollte [Kap91], also Texte, Bilder, Ton, Filme, Animationen, Karten etc. Die wesentlichen Anforderungen an das System waren daneben die Verknüpfung der Informationen durch *Links*, die Möglichkeit zur Verwaltung von sehr großen Datenbeständen, einfache Erweiterbarkeit (Benutzerschnittstelle, Dokumentenarten), die Benutzbarkeit sowohl durch unerfahrene Personen als auch durch Experten, das Angebot verschiedener Sprachen, von hochqualitativen Navigationswerkzeugen, personalisierbaren Einstellungen für jeden Benutzer mit dem gleichzeitigem Angebot des anonymen Zugangs, sowie die

[7] Das „G“ steht dabei für Graz.

Integration von Bezahlungsmöglichkeiten. Zusätzlich sollten Autoren bei der Erstellung durch Werkzeuge unterstützt werden, verschiedene Arten von *Terminals*, d.h. Endgeräten, verwendet werden können, in Abhängigkeit davon Nutzern unterschiedliche Rechte zugewiesen werden können und die Möglichkeit zum Entfernen veralteter und unwichtiger Information geschaffen werden. Schließlich sollte der Zugriff auf alle Informationen hinreichend schnell erfolgen können. Erstaunlich an diesem in noch größerer Tiefe in [Kap91] vorgestellten Anforderungsprofil ist zum einen, daß diese Anforderungen zusammengenommen erst jetzt in einigen kommerziellen Systemen erfüllt werden, und zum anderen, wie wenig die damals schon als wissenschaftlicher Stand der Kunst geltenden Einsichten die Entwicklung des als Hypertext-Systems zu betrachtenden WWW beeinflußt haben. Gerade letzteres wird im folgenden bei der genaueren Betrachtung der Konzepte deutlich.

HYPER-G ist als ein verteiltes Mehrbenutzer-Informationssystem entworfen worden, das mehrere Protokolle unterstützt, strukturierte Hypermedia-Informationen verwaltet und über das Internet als Client/Server-System betrieben wird [KAM95]. Es wurde objektorientiert entworfen und in C++ implementiert [Kap91]. Die wichtigsten Informationsartefakte im System sind Dokumente (oder Objekte), die unstrukturiert sind und zusätzlich eine Reihe von Metadaten-Attributen besitzen, Verzeichnisse (*Collections*), die weitere Verzeichnisse und Dokumente aggregieren, *Cluster*, die zusammengehörige Dokumente bündeln und bidirektionale Verknüpfungen (*Links*) zwischen Dokumenten. Zusätzlich existiert das Konzept der Tour, die Dokumente in einer von den Verzeichnissen unabhängigen Weise zu einer geführten Reihenfolge, die nicht notwendigerweise linear sein muß, zusammenfaßt. Alle Möglichkeiten eines Hypertext-Systems stehen dazu zur Verfügung; genutzt werden kann dies beispielsweise für Lektionen eines rechnergestützten Kurses oder zum Anlegen assoziativer Spuren (*associative trails*) [Bus45], die die persönliche Sicht und spezielle Interessen einzelner Personen zum Ausdruck bringen.

Die Architektur ist verteilt entworfen, so daß alle Objekte transparent und orthogonal über eine Vielzahl von Servern verteilt werden können. So kann eine Kollektion Dokumente von verschiedenen Servern umfassen und die Verknüpfungen zwischen Dokumenten können über beliebige Servergrenzen hinweg reichen. Auch Suchanfragen können über Servergrenzen hinweg gestellt werden. Konzeptuell enthält ein HYPER-G-Server drei Datenbanken: eine für die Metadaten der Dokumente, eine für die Verknüpfungen und eine für die Volltexte (Inhalte) der Dokumente. Ein skalierbares Server/Server-Protokoll (der *p-flood*-Algorithmus) stellt dabei sicher, daß die Aktualität und Integrität der Verknüpfungen, die über einen Server hinausgehen, innerhalb des Serververbundes gewahrt bleibt [Kap95]. Gemäß der Client/Server-Architektur bedient ein HYPER-G-Server verschiedene Arten von Clients: Unterstützt werden ein nativer HYPER-G-Client (HARMONY genannt), der über ein proprietäres Protokoll (HP-CSP) kommuniziert (das im Gegensatz zu HTTP verbindungsorientiert ist und so viele Probleme vermeidet) [KP94], HTTP-Clients, für die die Inhalte des Servers transparent in HTML umgewandelt werden, sowie GOPHER und WAIS. Anders als im WWW per HTTP kommuniziert ein Client immer nur mit genau einem Server, der im Falle von Anfragen an Dokumenten, die er nicht selbst besitzt, diese dann transparent vom entsprechenden Server anfordert und an den Client ausliefert (*Proxy*). Dies gestattet einfachere Clients, besseres Cache-Verhalten und einfachere Benutzerverwaltung, sowohl für die Authentifizierung, die nur einmal vorgenommen werden muß, als auch für die Sitzungsverwaltung. Für die Dokumente wurde eine eigener SGML-Dialekt geschaffen, HTF genannt, der viele Ähnlichkeiten mit HTML hat [Kap94]. Ferner werden diverse Bildformate und Multimediatypen (MPEG, Animationen etc.) unterstützt.

Interessant ist, wie stark sich im Laufe der Entwicklung das HYPER-G-System an die gleichzeitig stattfindende Entwicklung des WWW angepaßt hat (vgl. dazu [AKM95]): Anfänglich mit eigenen Protokollen (z.B. HG-CSP), Formaten (HTF) und Clients ausgestattet, wurden schrittweise die Standards des WWW adaptiert; zunächst durch zusätzliche Unterstützung, und mittlerwei-

le z.B. durch die Ersetzung von HTF durch HTML. Die Entwicklung mündete konsequenterweise in der Kommerzialisierung des Systems und der Positionierung als Portal und Wissensmanagement-System. Der Gedanke von verteilten, automatisch die Konsistenz bewahrenden Hypermedia-Servern ist dahingegen in den Hintergrund getreten; die lokale Integrität allerdings wird durch die entwickelten Mechanismen weiterhin erhalten.

3.4.2.2 Konzepte und Modelle

Nach diesem historischem Exkurs werden nun das HYPERWAVE-System in seiner heutigen Erscheinungsform als HYPERWAVE INFORMATION SERVER [Kap99a] und als HYPERWAVE INFORMATION PORTAL [HIP00] sowie seine Architektur [AHI00] und die Einsatzmöglichkeiten für das Wissensmanagement [Kap99b] näher untersucht. Die angesprochenen Grundkonzepte finden sich natürlich auch in dem neuen System immer noch teilweise wieder; jedoch hat sich die Terminologie leicht verändert.

Zunächst soll das oben bereits angerissene Modell von Dokumenten etc. genauer analysiert werden. Eine wichtige Beobachtung bei dem HTML-basierten Hypertext-System des WWW ist, daß dort nicht zwischen strukturellen und referentiellen Verknüpfungen (*Links*) unterschieden wird. Strukturelle Links stellen etwa Gliederung, Reihenfolge und Sammlungen von Dokumenten und ihrer Teile dar und bilden im wesentlichen wichtige Navigationsstrukturen ab. Referentielle Links hingegen nehmen die Aufgaben von klassischen Literaturverweisen, Fußnoten, Hinweisen auf externe Quellen etc. wahr. Beide werden in HTML technisch gesehen auf die gleiche Art behandelt; es gibt keine unterschiedlichen Modellierungskonstrukte dafür. Zudem sind die in HTML vorhandenen Links lediglich unidirektional, was eine Fülle von Problemen (referentielle Integrität, tote Links, Waisen) verursacht. Diese Probleme sind in HYPERWAVE anders gelöst; hier existieren sowohl explizite referentielle, bidirektionale Links, als auch strukturelle Elemente in Form der sogenannten *Container*.

Strukturelle Verknüpfungen: Ein Container ist eine Sammlung von weiteren Containern und Dokumenten. Ihre Aufgabe ist es, die Informationsressourcen zu strukturieren; dazu können sie beliebig geschachtelt werden. Jedes Dokument und jeder Container kann in mehreren Containern enthalten sein. Dadurch ergibt sich ein azyklischer, gerichteter Graph (DAG), dessen Wurzel die sogenannte *Root-Collection* ist. Es gibt mehrere konkrete Unterklassen der Container, z.B. *Collection* (die wichtigste), *Sequence* (bietet eine automatische Ordnung auf ihren Elementen) und mehrere Arten von *Clustern*. In den neuen Versionen von HYPERWAVE ist nur noch von Kollektionen die Rede. Das System gewährleistet die automatische Integrität der Kollektionen und baut aus diesen Informationen beim Zugriff entsprechende Navigationsstrukturen dynamisch in HTML auf.

Referentielle Verknüpfungen: Im WWW und der dort für die Inhalte verwendeten Auszeichnungssprache HTML sind alle Links (sowohl strukturelle als auch referentielle) unmittelbar in den Dokumenten enthalten, von denen sie ausgehen, und sind somit unidirektional. Dies erschwert die Erhaltung der Konsistenz nachhaltig. Im HYPERWAVE-System wurde ein grundsätzlich anderer Ansatz gewählt: Die referentiellen Verknüpfungen sind nicht direkt in den Dokumenten gespeichert, sondern werden separat in einem speziellen Repositorium (einer Datenbank) abgelegt. Dabei werden sowohl Quelle als auch Ziel des Links vermerkt, so daß er bidirektional traversierbar ist; wobei dennoch eine explizit definierte Richtung existiert. Die Bidirektionalität und die separate Speicherung bieten folgende Vorteile:

- Es lassen sich leicht alle Dokumente finden, die auf ein fragliches Objekt verweisen.

Dadurch lassen sich z.B. leicht Listen oder graphische Übersichten (Wissenslandkarten) erzeugen, die angeben, wie die Beziehungen zwischen den Dokumenten beschaffen sind.

- Die Erhaltung der Integrität vereinfacht sich drastisch; die Anpassung der Links beim Löschen oder Verschieben eines Dokuments ist sehr einfach.
- Die Tatsache, daß Links durch eigene Objekte modelliert werden, gestattet die Definition typischer Metadaten auch für die Link-Objekte. So kann jeder Link Autoren, Erstellungs- und Änderungsdatum sowie Rechte besitzen, die die Erstellung, Pflege und Darstellung personalisierter Kollektionen und Sichten ermöglicht, die unter Umständen für andere Benutzer nicht sichtbar sind.
- Ferner können Links auch für Dokumente definiert werden, für die die erzeugende Person keine Schreibrechte hat, oder die per se nicht beschreibbar sind (CD-ROMs). Dies wird zum Zwecke der Annotation ausgenutzt.

Weiterhin kann ein Link verschiedene Arten von Zielen haben: Ein Dokument als ganzes, einen bestimmten Teil eines Dokumentes oder eine Stelle darin, oder eine Kollektion und damit eine Sammlung von Dokumenten etc. Die ersten beiden Möglichkeiten entsprechen grob den auch in HTML möglichen Arten der Verknüpfung. In dem ursprünglichen HYPER-G-System [Kap91] bestand ein Link konzeptuell aus dem Link selber, der sich allerdings nicht direkt auf ein Dokument bezieht, sondern erst durch einen sogenannten Anker an ihm befestigt wird. Dabei hat ein Link genau zwei Anker, einen Quell- und einen Zielanker. Erst der Zielanker enthält die genaue Beschreibung des referenzierten Bereichs, z.B. einen Absatz des Dokuments. Anker können an allen Arten von Dokumenten befestigt sein, so auch an Bildern, Ton- und Filmdateien und entsprechend Ausschnitte daraus referenzieren. Jedes Dokument hat zudem einen *default*-Anker, der das Dokument als Ganzes referenziert. Eine Sonderform ist der aktive Anker, der erst bei Auswahl gemäß einer an bestimmter Stelle hinterlegten Programmlogik dynamisch Links generiert.

In den neuen Versionen von HYPERWAVE ist diese Komplexität weitgehend versteckt. Im Falle von HTML-Dokumenten, der neuen Lingua franca des Systems, müssten die Anker sogar entgegen den alten Prinzipien im Dokument selber erscheinen, was aber durch die Extraktion und das Einfügen der Link- und Anker-Informationen beim Ausgeben und Speichern der Seiten umgangen wird. Die Möglichkeit, daß einem Quellanker mehrere ausgehende Links zugeordnet sind, scheint nicht mehr benutzt zu werden, ebenso die aktiven Anker. Auch das Konzept der Tour findet sich nicht mehr explizit wieder; es läßt sich aber durch Kollektionen und Links gut nachbilden. Diese Konzepte werden hier so ausführlich geschildert, weil sie eine wichtige Grundlage für die zentrale Anforderung nach Verknüpfungskonsistenz bilden; eine Anforderung, die von dem in Kapitel 6 entworfenen Portal erfüllt werden muß.

Dokumente: Dokumente im HYPERWAVE-System sind nicht nur flache, unstrukturierte Dateien, sondern haben neben diesem Anteil noch Attribute und Metadaten, die in einer objektorientierten Datenbank gehalten werden, in der auch die strukturellen und referentiellen Informationen stehen. Die Begriffe Attribute und Metadaten bedeuten in diesem Kontext dasselbe. Es wird dabei zwischen systemdefinierten und benutzerdefinierten Attributen unterschieden. Die systemdefinierten Attribute umfassen Angaben über Autor bzw. Eigentümer des Dokuments, Erstellungsdatum, Änderungsdatum, Rechte des Eigentümers und anderer Benutzer (Lesen und Schreiben), Verfallsdatum, einen Namen, die verwendeten Sprachen, einen Dokumenttyp und eine Identifikationsnummer (ID). Anwendungsentwickler können eigene Dokumenttypen erstellen und bestehende Dokumenttypen um benutzerdefinierte Attribute erweitern. Es besteht die Möglichkeit, Attribute zum schnelleren Zugriff zu indizieren. Ebenso läßt sich spezifizieren, ob es für ein Attribut die Pflicht

zur Erfassung gibt (*mandatory attributes*). Es wird bereits eine Reihe unterschiedlicher Dokumentarten angeboten und unterstützt.

Jedes Dokument muß in zumindest einer Kollektion enthalten sein, damit es überhaupt erreichbar ist. Dieses Konsistenzkriterium wird vom System automatisch erzwungen.

Konzeptuell elegant gelöst ist, daß sowohl Links als auch Dokumente sowie Kollektionen in dem zugrundeliegenden objektorientierten Modell Subklassen derselben abstrakten Klasse sind und daher gemeinsam ererbte (System-)Attribute haben. Dies gestattet die orthogonale Suche auch über Links, da auch sie u.a. Eigentümer, Rechte und Datumsfelder besitzen. Die bisher beschriebenen Zusammenhänge sind in Abbildung 3.9 in Form eines konzeptuellen UML-Klassendiagrammes zusammengefaßt (rekonstruiert nach den Angaben aus [Kap91, Kap99a, DH95]). Die Referenzierung von Kollektionen durch Links wird dadurch ermöglicht, daß die Kollektion eine Subklasse des Dokuments ist. Ferner ererben Kollektionen wichtige Attribute der Dokumente wie Schlagwörter etc. Durch die Subklassenbildung ist der uniforme Zugriff auf Dokumente und Kollektionen möglich. Diese Art der Modellierung findet sich in ähnlicher Form im Entwurfsmuster *Composite* nach [GHJV95] wieder.

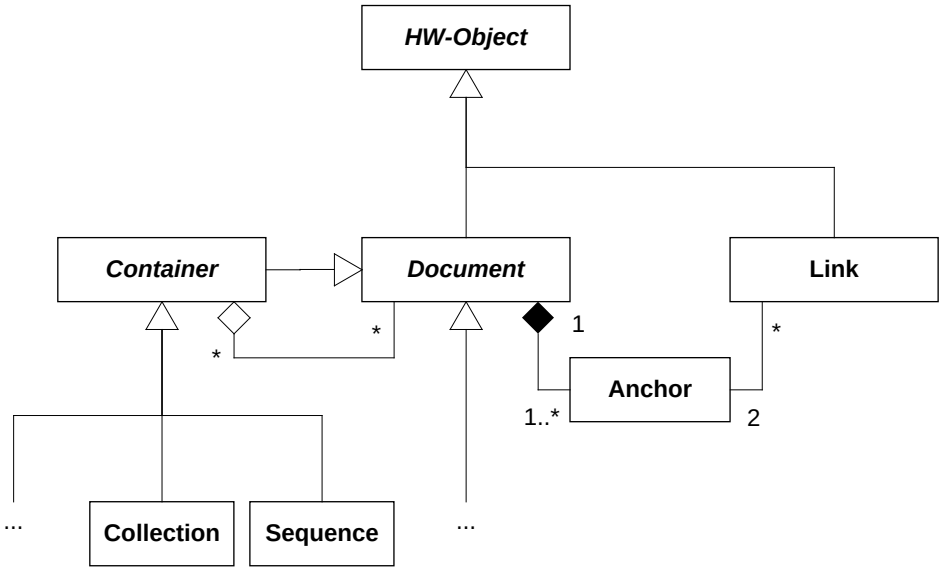

Abbildung 3.9: Konzepte des HYPERWAVE-Systems

Das System bietet komfortable Anfragefunktionen über Dokumente, Kollektionen und Links an, die die Attribute und den Volltext (der Dokumente) einbeziehen können. Die Volltextsuchfunktionen sind sehr mächtig und berücksichtigen Wortstämme (*stemming*), phonetische Ähnlichkeit und Nachbarschaft (Abstand der Wörter im Text). Ermöglicht wird dies durch die Nutzung der Volltext-Indexierungsmaschine VERITY SEARCH, die in der Lage ist, über 200 gängige Dokumentenformate zu indexieren [Ver01, Ver00a, Kap99a]. Dabei läßt sich gezielt in einzelnen Kollektionen suchen. Die Treffer werden sortiert nach Relevanz angezeigt.

Eine hilfreiche Eigenschaft ist die Möglichkeit, die Anfragen benutzerspezifisch abspeichern und so bei Bedarf wiederverwenden zu können. Sie erhalten dazu einen Namen und werden in eine Kollektion eingefügt. Suchen erscheinen dann wie eine weitere Kollektion, die die dynamisch berechneten Ergebnisse des Suchlaufes enthalten. Besonders nützlich ist die zusätzliche Möglichkeit, die gespeicherten Suchen gemäß eines Terminplanes regelmäßig im Hintergrund (auch wenn der Benutzer nicht mit dem System arbeitet) ablaufen zu lassen und die Ergebnisse per

e-Mail an den jeweiligen Benutzer zu verschicken. Diese Funktionalität bildet die Grundlage für Techniken, die neuerdings als Kanäle (*channels*), *push technology* oder Subskription bezeichnet werden und das Paradigma des Benutzers in einer aktiven Rolle gegenüber passivem Inhalt und Systemen (*pull*) weiterentwickeln zu einem System, das aktiv auf die Informationsbedürfnisse der Benutzer eingeht und passende Informationen von sich aus anbietet (*push*).

3.4.2.3 Funktionalität

Auf der Basis der bis hierhin beschriebenen Konzepte baut der HYPERWAVE INFORMATION SERVER (HIS) auf [HIP00]. Er ist damit eine im Prinzip generische Plattform zur Realisierung beliebiger Web-Anwendungen im Kontext des Dokumenten-, Content- und Wissensmanagements und bietet den uniformen, zentralen Zugang zu allen Informationen des Systems (*single point of exchange*). Der Server wird mit einer generischen Standardanwendung, dem HYPERWAVE INFORMATION PORTAL, ausgeliefert. Es umfaßt weitreichende Funktionen zur kollaborativen Erstellung und Publikation von Informationen. Für konkrete Projekte wird diese Standardanwendung jedoch meist den vorhandenen Bedürfnissen angepaßt, beispielsweise bezüglich Benutzungsoberfläche, organisationsweitem Erscheinungsbild und speziellen Anforderungen. Im folgenden werden nun die Funktionen des Portals im einzelnen diskutiert, um die Abdeckung der für Wissensmanagement-Portale nötigen Funktionalität zu untersuchen. Daran schließt sich im nächsten Abschnitt eine Analyse der Software-Architektur an.

Das generische HYPERWAVE INFORMATION PORTAL bietet bereits in der Standardversion eine Fülle von Funktionen an:

- Definition von Benutzern und Gruppen: Dazu existiert ein System, das die Vergabe von fein abgestuften Rechten pro Benutzer und Gruppe erlaubt. Benutzergruppen können zudem hierarchisch geschachtelt werden, und Benutzer können Mitglied in beliebigen Gruppen sein. Allen Objekten (Dokumenten, Links, Kollektionen etc.) werden Rechte wie Lesen, Schreiben und Löschen zugewiesen, die für Eigentümer, Gruppen und Personen getrennt einstellbar sind. Das Sicherheitskonzept von UNIX-Dateisystemen hat dabei offenbar Pate gestanden.

  Um die Integration in bereits vorhandene Infrastruktur und Organisationsstrukturen zu erleichtern, kann die Benutzerverwaltung durch Kopplung an bestehende LDAP-Server, WINDOWS NT-Benutzerverzeichnisse oder SUNS *Network Information Services* (NIS) betrieben werden.

- Strukturierten Zugang zu den Informationen des Servers: Jeder Benutzer besitzt (optional) eine ausgezeichnete, eigene Kollektion, die sogenannte *Home collection*, die die persönliche Wurzel des in Kollektionen organisierten Informationsangebots bildet. Hier werden auch abgespeicherte Suchanfragen abgelegt. Für alle Kollektionen lassen sich angepaßte Sichten in Form von Visualisierungsvorlagen schaffen, die in Abhängigkeit vom Benutzer, seiner Gruppe oder seiner Rolle gewählt werden können.

- Grundlegende Funktionen für das Dokumentenmanagement: Neben der Sperrung eines Dokuments zur Bearbeitung können Dokumente versioniert abgelegt werden. Kollaborative Arbeit wird dadurch unterstützt, daß während des Editierungsvorgangs eines gesperrten Dokuments andere Benutzer bis zur Freigabe die alte Version sehen. Alte Versionen können eingesehen, wiederhergestellt oder gelöscht werden.

  Über die Benutzungsoberfläche des Web-Browsers können Dokumente erzeugt (hochgeladen) werden, deren Attribute eingesehen und verändert werden sowie in den Kollektionen angeordnet werden. Neben dieser Art des Publizierens von Dokumenten kann dies auch

direkt aus Büroanwendungen geschehen, die die Schnittstelle ODMA (*Open Document Management API*) oder MICROSOFTS WEBPOST-API unterstützen. Eine weitere Möglichkeit der Ansicht und Nutzung besteht über die sogenannten *Virtual Folders*, eine Sicht der Kollektionen und Dokumente als Netzlaufwerk im WINDOWS-EXPLORER.

- Funktionen zur kollaborativen Arbeit (*Groupware*-Funktionen): Alle Dokumente und Kollektionen können von den Benutzern mit Annotationen versehen werden. Eine Annotation ist konzeptuell wieder ein Dokument und kann so wiederum annotiert werden. Da die Annotation über eine Verknüpfung mit dem annotierten Dokument verbunden ist, läßt sich gemäß der Anker-Link-Semantik nicht nur das Dokument als ganzes, sondern auch jeder beliebige Teil davon oder jede Stelle darin gezielt annotieren. Wie alle Dokumente haben auch Annotationen Zugriffsrechte, können also praktisch als privat oder öffentlich gekennzeichnet werden. Über die Nutzung der oben beschriebenen subskribierbaren Suchfunktionen und die Möglichkeit, Annotationen zu annotieren, lassen sich Diskussionen in der Gruppe effizient verwirklichen. Auf diesen Eigenschaften baut ein integriertes Diskussionsforum auf. Es gibt verschiedene Typen von Beiträgen (Fragen, Antworten, Bemerkungen, Hinweise), die wiederum mit Anlagen versehen werden können, was intern durch Verknüpfungen zu den angefügten Dokumenten realisiert wird.

- Neben dem Publizieren von Dokumenten wird ein Freigabe-Prozess zur Sicherung der Qualität der Dokumente angeboten, in den eine Reihe von Benutzern in verschiedenen Rollen (Autor, Auditor, Leser) eingebunden werden können. Darüberhinaus können durch Integration spezieller Komponenten aufwendigere, teilweise automatisierte Transformationsprozesse von Dokumenten gestaltet werden.

Die Benutzungsoberfläche des Portals wurde vollständig mithilfe von HTML und JAVASCRIPT realisiert, wobei in Abhängigkeit vom benutzten Web-Browser auch dynamisches HTML zum Einsatz kommt (DHTML). Die einzelnen Funktionsbereiche (etwa Kollektionen, Dokumente, Suchergebnisse, Informationen aus externen Quellen) werden in sogenannten *Information Tracks* angezeigt, kleinen, in HTML gestalteten schematischen Unterfenstern des Browsers. Sie basieren (serverseitig) auf den objektorientierten Dokumentklassen, und es können von Entwicklern neue erstellt werden. Eine Reihe von vordefinierten *Tracks* sind im Portal bereits enthalten; diese ermöglichen Funktionen wie Verwaltung von Lesezeichen, Anzeige von Dokumenten (HTML, Bilder etc.), Nachrichten (*Newsticker*), die Integration von Informationen aus LOTUS NOTES und MICROSOFT EXCHANGE sowie die Anzeige von gespeicherten Suchen. Serverseitig werden die Seiten aus HTML-Bausteinen (Vorlagen) und einer darin eingebetteten Makrosprache namens PLACE zusammengesetzt. Für weitergehende Aufgaben kommt serverseitiges JAVASCRIPT zum Einsatz, das über zahlreiche Schnittstellen Zugriff auf den Server und alle Objekte und Dokumente hat.

3.4.2.4 Architektur

Die Architektur des HYPERWAVE INFORMATION SERVERS[8] ist prinzipiell dieselbe wie die des HYPER-G-Servers geblieben; sie hat aber einige Weiterentwicklungen erfahren, um die neu hinzugekommenen Möglichkeiten realisieren zu können. Grundsätzlich besteht ein Server aus einer Reihe von Komponenten, die falls nötig auf unterschiedliche Rechner verteilt werden können. Ein Überblick über die Architektur ist in Abbildung 3.10 gegeben.

Auf der obersten Strukturierungsebene sind vier Schichten zu erkennen: oben die der Clients (Web-Browser sowie die erwähnten *Virtual Folders*), darunter die Serverschichten der Anwen-

[8] im folgenden kurz Server genannt

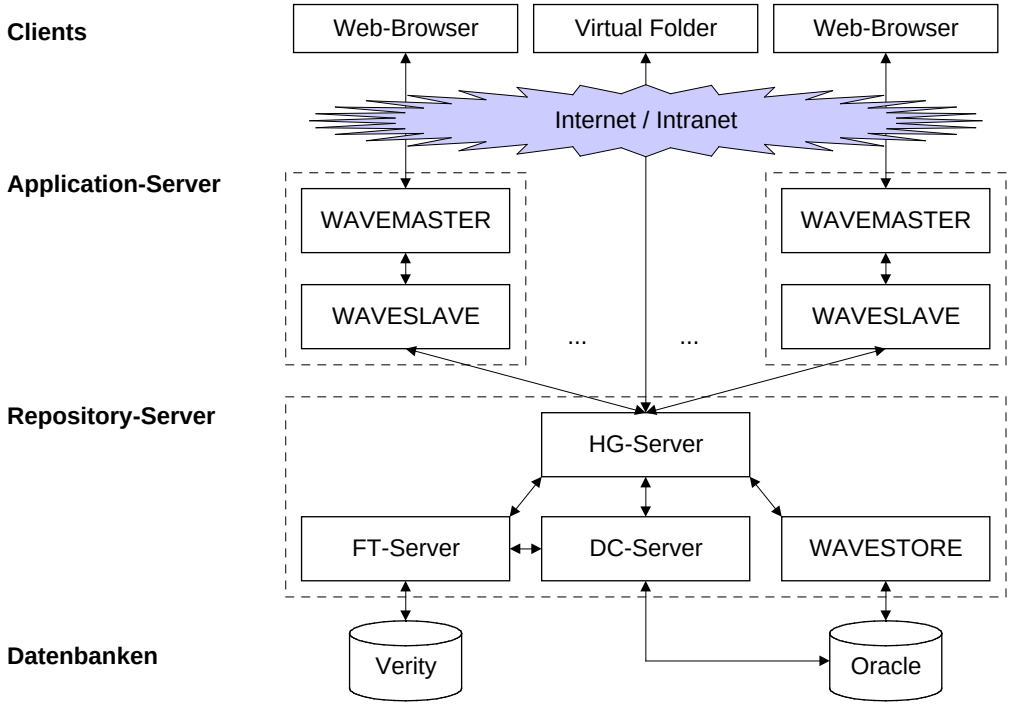

Abbildung 3.10: Architektur des HYPERWAVE INFORMATION SERVERS

dungsserver, der Speicherungsschicht sowie der Datenbanken. Es kann pro Server einen oder mehrere Anwendungsserver geben (in der Abbildung sind zwei dargestellt), um eine bessere Lastverteilung zu erreichen. Der Einsatz der ORACLE-Datenbank ist optional. Die beiden Serverschichten gliedern sich folgendermaßen:

1. Ein Anwendungsserver (*Application Server*) besteht aus zwei weiteren Unterkomponenten: dem WAVEMASTER und dem WAVESLAVE. Insgesamt gesehen bildet jeder Anwendungsserver einen Client des Speicherungsservers, und kommuniziert mit diesem über das bereits erwähnte Protokoll HG-CSP [Kap94], das früher schon von den nativen Clients des HYPER-G-Systems verwendet wurde. So ist klar, wieso ohne Probleme beliebig viele Anwendungsserver aufgesetzt werden können.

   Ein Anwendungsserver ist dafür zuständig, HTTP-Anfragen aus dem Internet oder Intranet zu bearbeiten und die Antworten zu generieren. Er enthält die gesamte Anwendungslogik des Portals. Dazu teilen sich die Komponenten WAVEMASTER und WAVESLAVE die Arbeit folgendermaßen:

   - Die WAVEMASTER-Komponenten nimmt Verbindungen aus dem Internet oder Intranet (über HTTP und das proprietäre HWTP-Protokoll) an und erzeugt und verwaltet die daraus resultierenden gleichzeitigen Sitzungen der verschiedenen Benutzer. Für jede Sitzung eines identifizierten Benutzers wendet sich die Komponente an den WAVESLAVE, der daraufhin einen neuen Prozess zur Abwicklung der Kommunikation startet. Anonyme Benutzer werden aus einem Pool von Prozessen des WAVESLAVE bedient.
   - Die WAVESLAVE-Komponente erzeugt auf Anfrage des WAVEMASTERS für jede Sitzung einen neuen Prozess (bzw. *thread* auf den Plattformen, die dies unterstützen),

der dann für die Ausführung der Anwendungslogik für diesen Benutzer zuständig ist. Jeder Prozess verfügt über eine eigene HG-CSP-Verbindung zum Speicherungsserver, um die erforderlichen Informationsartefakte (Dokumente, Kollektionen) zu erhalten; diese werden dann dynamisch in die Vorlagen eingebettet. Dazu werden von einer sogenannten *Template engine* die in HTML vorliegenden Visualisierungsvorlagen unter Nutzung der Makrosprache PLACE und des serverseitigen JAVASCRIPTS interpretiert. Zur Steigerung der Effizienz werden Vorlagen vorkompiliert und in schnellen Zugriffsspeichern (*Caches*) abgelegt. Zusätzlich werden vom WAVESLAVE die objektorientierten Dokumentenklassen verwaltet und ausgeführt, die unten noch genauer besprochen werden.

2. Der Speicherungsserver (*Repository Server*) besteht aus weiteren vier Komponenten: Einer zentralen Koordinierungsinstanz und drei Datenbankservern, die die verschiedenen Arten von Informationen verwalten. Alle vier werden als separate Prozesse betrieben.

   - Der HG-Server (der Name stammt wohl noch vom alten HYPER-G-Server) ist die zentrale Instanz des gesamten Speicherungsservers. Er nimmt die Anfragen aller Clients über das Protokoll HG-CSP entgegen, leitet sie an die weiteren Komponenten weiter und übermittelt die Ergebnisse zurück. Ferner obliegt dem HG-Server die Benutzerverwaltung, entweder über eine integrierte Verwaltung oder durch Ansprache von externen Diensten wie LDAP, NIS etc. Zusätzlich dazu speichert der HG-Server bestimmte Objekte (Anker etc.) zwischen, um den Systemdurchsatz zu verbessern.

     Erst im Falle der Nutzung von mehreren verteilten HYPERWAVE INFORMATION SERVERN (also vollständiger Systeme wie in Abbildung 3.10 dargestellt), die einen Verbund bilden (einen sogenannten *Server Pool*), kommt dem HG-Server eine wichtigere Rolle zu, nämlich die Koordinierung der Weiterleitung von Suchanfragen an entfernte Server, die Übermittlung von entfernten Objekten (Kollektionen, Dokumenten) und insbesondere die Erhaltung der referentiellen Integrität der serverübergreifenden Verknüpfungen. Dies entspricht genau den Aufgaben im ursprünglichen, verteilten Hypermedia-Modell des HYPER-G-Systems. Interessant ist, daß diese konzeptuell weitreichende Idee des alten Systems sich in der eher technisch begrenzten Form des *Server Pools* wiederfindet.

   - Die Datenbank-Komponente WAVESTORE hat die Aufgabe, alle Arten von Attributen und Metadaten aller HYPERWAVE-Objekte, also von Kollektionen, Dokumenten, Verknüpfungen etc. zu verwalten, mit Ausnahme der eigentlichen Massen-Dokumenteninhalten. Diese werden vom DC-Server verwaltet; die entsprechenden Zugriffspfade eines Dokuments darauf werden aber vom WAVESTORE genauso wie die übrigen Attribute gespeichert. Außerdem führt der WAVESTORE Suchanfragen nach den Teilen der Schlüsselworte durch, die als Metadaten in Dokumentattributen abgelegt sind. Der WAVESTORE kommuniziert mit den höheren Schichten lediglich via HG-Server über ein spezielles TCP/IP-basiertes Protokoll namens HG-WS. Bereits diese Komponente prüft beim Zugriff auf Objekte die Berechtigungen der Benutzer.

     Zur Persistentmachung der vom WAVESTORE administrierten Informationen können zwei unterschiedliche Datenbanken verwendet werden: Entweder eine native, von HYPERWAVE selbst entwickelte sogenannte objektorientierte Datenbank[9] (*object soup* genannt), die alle Attribute aller Objekte eines Typs „flach“ in Textdateien als Zeichenketten ablegt und zum Zwecke des schnellen Zugriffs auf indizierte Attribute B-Bäume in separaten Dateien erstellt und pflegt; oder es kann alternativ dazu die

[9] Der Terminus „objektorientiert“ ist in diesem Kontext verglichen mit den eigentlich an objektorientierte Datenbanken gestellten Anforderungen [Cat00] natürlich falsch, da lediglich Daten, aber keine Funktionen abgelegt werden und so auch keine Mechanismen wie Vererbung möglich sind.

relationale Datenbank ORACLE 8 benutzt werden, wobei dann pro Objekttyp eine Tabelle angelegt wird, die alle Attribute aller entsprechenden Objekte aufnimmt.

- Die neben den Attributen der Dokumente vorhandenen eigentlichen Inhalte der Dokumente (z.B. Dateien von Büroanwendungen) werden von dem DC-Server (*Document Content Server*) verwaltet. Sie werden entweder als Dateien im Dateisystem abgelegt, wobei kleine Dateien zu größeren zusammengefaßt werden und größere in eigenen Dateien gespeichert werden, oder unter Nutzung der *Binary Large OBjects* (BLOBs) in einer ORACLE 8-Datenbank. Hier kann entweder dieselbe Datenbank wie für den WAVESTORE verwendet werden, wie in Abbildung 3.10 angedeutet ist, oder aber eine eigene, um die Leistung des Gesamtsystem weiter zu erhöhen. Der DC-Server kommuniziert über eine ganze Reihe von speziellen Protokollen mit dem HG-Server, dem FT-Server zum Zwecke der Volltext-Indizierung sowie direkt mit den jeweiligen WAVEMASTER/WAVESLAVE-Komponenten, um die Inhalte ohne zusätzliche Umwege über den HG-Server schneller ausliefern zu können (in der Abbildung nicht dargestellt). Ferner werden Dokumente von entfernten Servern hier lokal zwischengespeichert.
- Der FT-Server (*FullText Server*) ist für die Erstellung und Wartung der Volltext-Indizes sowie für die Durchführung der Volltext-Suchanfragen verantwortlich. Dazu kommuniziert er über proprietäre Protokolle einerseits mit dem HG-Server als Koordinierungsinstanz, die die Anfragen absendet und den FT-Server über geänderte, neu hinzugekommene oder gelöschte Dokumente informiert, und andererseits direkt mit dem DC-Server, um ohne zusätzliche Verzögerungen an die zu indizierenden Dokumente zu gelangen.

  Die eigentliche Volltextindizierung wird entweder durch eine eigene, native und eher einfache Lösung, die einen umgekehrten Wortindex erstellt und lediglich reine Text- und HTML-Dateien zu indizieren vermag, oder durch das vom Drittanbieter VERITY [Ver01] stammende Produkt VERITY SEARCH [Ver00a] geleistet. Die eigene Lösung kann durch das Erstellen von Filtern erweitert werden, VERITY hingegen ist bereits in der Lage, über 200 Dokumentenformate zu erkennen und den Text daraus zur Indexierung zu extrahieren.

Das System ist stark skalierbar; je nach vorhandenem Lastprofil, das von der Art der Benutzung des Systems abhängt, können alle beschriebenen Komponenten (bis auf jeweils zusammengehörige WAVEMASTER/WAVESLAVE-Paare) beliebig auf verschiedene Rechner verteilt werden.

3.4.2.5 Dokumentenklassen

Abschließend wird noch ein bemerkenswerter Bestandteil des HYPERWAVE-Server-Systems vorgestellt, nämlich die Dokumentenklassen. Wie bereits bei der Schilderung der Aufgaben des Anwendungsservers angedeutet, entsteht die Funktionalität des Portals nicht nur durch die Interpretation der PLACE-Makrosprache und des serverseitigen JAVASCRIPTS in den Visualisierungsvorlagen (*templates*), sondern auch durch die sogenannten Dokumentenklassen [AHI00, Kap99a]. Gemäß des objektorientierten Konzepts der relevanten Informationsartefakte sind alle Objekte, also Dokumente, Kollektionen etc., durch Klassen im System repräsentiert. Insbesondere die verschiedenen Dokumenten-Arten werden durch Subklassen der abstrakten Klasse Dokument gebildet. Eine Reihe von vordefinierten Klassen, die die Basis-Funktionalität bereitstellen, sind bereits im System fest eingebaut. In den Visualisierungsvorlagen besteht über eine umfangreiche Schnittstelle Zugriff auf alle benötigten Objekte, Methoden und Attribute. Wenn die generische Funktionalität, die das HYPERWAVE INFORMATION PORTAL implementiert, nicht

ausreicht, so können neben der Anpassung der vorhandenen Vorlagen zur Änderung des Erscheinungsbildes eigene, neue Dokumentenklassen erstellt und verwendet werden.

Die Definition neuer Klassen folgt ganz dem objektorientierten Paradigma: Klassen werden von bestehenden (evtl. abstrakten) Basisklassen abgeleitet und ererben alle Methoden und Attribute von ihnen. Die Exemplare dieser Klassen sind als Dokumente automatisch persistent und werden im Speicherungsserver abgelegt. Die Besonderheit des Systems besteht darin, daß die Klassendefinition für neu zu erzeugende Klassen ebenfalls als Dokument, genauer gesagt als Kollektion, persistent in dem INFORMATION SERVER abgelegt wird, also innerhalb des Dokumentenspeichers selbst. Dazu muß innerhalb einer durch einen speziellen Namen ausgezeichneten Kollektion einfach eine weitere Kollektion mit einer Reihe von zusätzlichen Attributen, die u.a. den Namen, die Superklasse und die Typen der Attribute angeben, angelegt werden. Die Methoden werden durch Dokumente einer speziellen Dokumentenklasse innerhalb dieser Kollektion definiert, die als zusätzliche Dokumenten-Attribute die Signatur der Methode (Parameter, Rückgabewert etc.) erhalten und als Dokumenteninhalt den JAVASCRIPT-Code der Methode. Es gibt beispielsweise überschreibbare Methoden zur Visualisierung eines Exemplars der Klasse, die zu bestimmten Zeitpunkten aufgerufen werden, um dynamisch HTML oder clientseitiges JAVASCRIPT zu erstellen. Weitergehende Funktionalität ist so einfach realisierbar. Die JAVASCRIPT-Schnittstellen des Systems und die dahinterliegende Ausführungseinheit für die Anwendungslogik der WAVESLAVE-Komponente verarbeiten dann die so definierten Klassen beim Zugriff auf ihre Exemplare (Dokumente). In dem gesamten JAVASCRIPT-System werden mit großem Aufwand die klassischen objektorientierten Eigenschaften wie dynamisches Binden, Klassen- und Instanzmethoden, Konstruktoren, Destruktoren, implizite `this`-Parameter und vieles mehr durch spezielle Namenskonventionen und explizite Metadatenobjekte simuliert. Die Exemplare der neu definierten Klassen können wie alle Dokumente an beliebigen Orten in den Kollektionshierarchien abgelegt werden. Zur Definition und Pflege der neuerstellten Klassen kann das Portal selbst benutzt werden; eine optionale Erweiterung versieht es mit den notwendigen Bestandteilen der dafür erweiterten Oberfläche. Die Definition von Klassen ist so während der Laufzeit des Systems möglich, was das interaktive Erstellen und Testen der Software erheblich erleichtert.

3.4.2.6 Bewertung

Das Konzept, sowohl die Exemplare von Klassen als auch die Klassendefinitionen selbst uniform mit den Mitteln des zugrundeliegenden Speichersystems abzulegen, erinnert stark an die Konzepte persistenter Objektsysteme, in denen wie zum Beispiel im TYCOON-System die ablaufgerechte Repräsentation übersetzter Funktionen orthogonal neben den Daten im persistenten Objektspeicher gehalten wird [Mat93]. Die konsequente Erweiterung der Persistenz von Daten und Code auf (aktive) Prozesse [Mat96] wurde allerdings im HYPERWAVE-System nicht nachvollzogen; diese Aufgabe ist aber wegen der eingesetzen Sprache (JAVASCRIPT) vermutlich nicht mit vertretbarem Aufwand lösbar. Erstaunlich ist, daß bei der Realisierung dieses anspruchsvollen Konzepts als Implementierungssprache JAVASCRIPT gewählt wurde: Objektorientierte Sprachkonzepte, die hier, verglichen mit anderen objektorientierten Sprachen wie C++, JAVA und TYCOON-2 [GW98, Wah98], fast vollständig vorhanden sind, in einer grundsätzlich nicht wirklich objektorientierten Sprache zu realisieren, ist zumindest sehr aufwendig und führt durch die zahlreichen benötigten Hilfskonstruktionen zu schlecht lesbarem, aufwendig wartbarem und fehleranfälligem Code. Die Wahl einer objektorientierten Sprache wie JAVA hätte – zumindest sprachlich gesehen – sicher eine wesentlich elegantere Lösung hervorgebracht. Diese Unsicherheiten werden durch die Wahl einer schwach und dynamisch typisierten Skriptsprache noch erhöht. Inwieweit die Vorkompilierung des Codes diese Schwächen aufhebt, ist nicht bekannt.

Eine abschließende Bewertung des HYPERWAVE INFORMATION SERVERS und des Portals ergibt folgende Beobachtungen: Konzeptionell ist das System aufgrund seiner langen, akademisch geprägten Historie wesentlich ausgereifter als andere Systeme wie beispielsweise der SHAREPOINT SERVER. Das Dokumentenmodell strukturierter Dokumente unter Beachtung der Trennung von Struktur, Inhalt und Layout entspricht dem in Abschnitt 3.3.2 besprochenen Modell für moderne Content-Management-Systeme weitestgehend. Die uniformen Strukturierungsmöglichkeiten durch Kollektionen und die bidirektionalen Verknüpfungen sind herausragende Eigenschaften, die die meisten anderen Systeme in der hier vorgefunden konzeptionellen Stärke und Durchgängigkeit nicht bieten. Lediglich die aus diesem Rahmen fallende Modellierung der Benutzer und Personen verringert die Durchgängigkeit des Modells etwas. Die Idee der orthogonalen Persistenz der Dokumentenklassen ist bemerkenswert fortschrittlich. Die Wahl der Implementierungssprache JAVASCRIPT und Mischung mit der Makrosprache PLACE in den Vorlagen stellt einen Bruch in der guten Tradition der Konzepte dar, der durch den wirtschaftlichen Druck der Kommerzialisierung des Produkts zu erklären ist. Die Funktionalität des Portals deckt gemessen an den Anforderungen aus Abschnitt 3.3.1 weite Bereiche des Dokumentenmanagements sowie elementare Content-Management-Funktionen ab. Der Schwerpunkt ist aber eher auf Intranet-basierten Informations- und Wissens-Portallösungen zu sehen, wofür das System normalerweise noch für den konkreten Einsatzzweck umfangreiche Anpassungen und Vorbereitungen erfahren dürfte. Insgesamt ist das HYPERWAVE-System als eines der interessantesten und leistungsfähigsten Systeme im Bereich der Portalsoftware zu betrachten.

3.4.3 Lotus Discovery Server und K-station

Als drittes und letztes kommerzielles System wird das LOTUS KNOWLEDGE DISCOVERY SYSTEM von LOTUS betrachtet, das (bisher) im wesentlichen aus zwei Produkten besteht: Der LOTUS K-STATION, einem Intranet-Wissensportal [Lot01b, Lot01c], und dem optional zusätzlich dazu einsetzbaren LOTUS DISCOVERY SERVER [Lot01d]. Beide Produkte sind eng sowohl untereinander als auch mit den bisher entwickelten Produkten von LOTUS verzahnt; so ist zum Betrieb der K-STATION ein LOTUS DOMINO SERVER erforderlich. Nachdem sich die Entwicklung des in der Fachwelt mit Spannung erwarteten Portals, das währenddessen den Codenamen *Raven* trug, unerwartet lange hingezogen hat [IW00g], scheint sich kurz nach der Verfügbarkeit bereits wieder eine Neuausrichtung der Produktpalette abzuzeichnen: Die Integration des Portals in die Anwendungsserver-Plattform WEBSPHERE der Konzernmutter IBM und die (optionale) Nutzung des DISCOVERY SERVERS ohne das Portal.

Das allen beiden Produkten zugrundeliegende Konzept ist eine Sicht der Informationsartefakte, die diese in drei grundsätzliche Kategorien aufteilt: *People*, *Places* und *Things*, also Personen, Orte und Dinge [Web01b].

People: Mit Personen sind sowohl die zur Benutzung der Systeme berechtigten internen Mitarbeiter gemeint als auch externe Personen wie beispielsweise Kunden, Partner und weitere Experten. Für jede Person wird ein umfangreiches Profil erstellt, das Aussagen über Tätigkeiten, Kenntnisse und vieles mehr enthält.

Places: Unter Orten werden reale oder virtuelle, oft strukturierte Plätze und Ablagemöglichkeiten verstanden, an denen Gruppen von zusammenarbeitenden Personen sich virtuell „treffen“, Informationen austauschen und miteinander interagieren können. Diese beispielsweise von der K-STATION in Form von Gemeinschaftsplätzen angebotenen Orte können vom DISCOVERY SERVER klassifiziert werden, um anderen Personen die Nutzung der dort vorhandenen Informationen zu ermöglichen.

Things: Als Dinge werden alle reinen Informationsartefakte wie Dokumente, Datenbankinhalte und Web-Seiten, aber auch Prozesse gesehen, die innerhalb der Organisation erzeugt, verteilt und genutzt werden. Dinge werden in der Regel an Orten aufbewahrt und gehören Personen.

Das Portal der K-STATION bietet den uniformen Zugriff (*single point of access*) auf alle unternehmensrelevanten Informationen an. Die besondere Stärke liegt dabei in der inhärenten Verwaltung der Kompetenzen aller Personen und der Beziehungen zwischen den Informationsobjekten. Die wichtigsten Funktionen, die das Portal anbietet, sind:

- Zwei Arten von Orten (*Places*) werden angeboten: Einerseits persönliche (*Personal places*), in denen der Benutzer seinen individuellen Gewohnheiten und Bedürfnissen folgend Informationsquellen wie Anwendungen, Web-Seiten, Diskussionsforen, e-Mail-Werkzeugen etc. anordnen und konfigurieren kann, andererseits Gemeinschaftsplätze (*Community places*), die zum Austausch von Informationen in einer Gruppe dienen. Hier können ebenso Diskussionsforen, Referenzen zu Personen, Inhalten und Prozessen sowie Unterstützung für ad hoc-Prozesse benutzt werden.
- Die web-basierte Oberfläche wird durch sogenannte *Portlets* sowie *Pages* aufgebaut. Erstere entsprechen vom Erscheinungsbild und der Funktionalität her Fenstern und können individuell in Abhängigkeit von den wirksamen Rechten des Benutzers konfiguriert werden. Eine große Anzahl von vordefinierten *Portlets* steht im System bereit; diese reichen von Werkzeugen zur Anzeige von (externen) HTML-Seiten verbunden mit der Möglichkeit, darüber Anwendungen einzubinden, über Zugriff auf e-Mail-Systeme, LOTUS NOTES-Datenbanken, Diskussionsforen, Lesezeichenverzeichnisse bis hin zu Werkzeugen wie Kalendern und Projektplanern. Die *Pages* genannten Konstrukte ermöglichen die Kategorisierung von Informationen anhand von Themen oder Funktionen unter Nutzung der üblichen Verzeichnis-Metaphern.
- Über ein *People awareness* genanntes Werkzeug ist ersichtlich, welche Benutzer existieren sind und welche davon gerade aktiv mit dem System arbeiten. Diese können durch eingebaute Kollaborationswerkzeuge beispielsweise sofort zu Online-Diskussionen eingeladen werden (*Instant collaboration*).
- Ein rollenbasiertes Rechtekonzept teilt die Benutzer in verschiedene Gruppen mit abgestuften Rechten ein, die zwischen Administrator, Manager, Designer und Teilnehmer rangieren und die Personen unterschiedlich stark an der Gestaltung der Oberfläche (Plätze etc.) teilhaben lassen.

Die Intention und die Entwurfsziele der K-STATION sind, ein einfach zu benutzendes, in weiten Grenzen konfigurierbares und vor allen Dingen leicht erweiterbares Portal zu schaffen. Der Schwerpunkt ist die Zusammenarbeit von Individuen und Gruppen. Dazu besitzt das Portal eine erweiterbare Architektur, die die Integration bestehender Anwendungen sehr einfach macht, indem Standards wie dynamisches HTML (DHTML), XML, JAVASCRIPT und JAVA SERVLETS im Server verwendet werden. Insgesamt bildet das Portal nur einen Rahmen, in den weitere Funktionalität eingebettet werden muß, wie dies beispielsweise durch den DISCOVERY SERVER geschieht. Der Rest des Abschnitts behandelt deswegen ausschließlich diesen, da er zahlreiche insbesondere für das Wissensmanagement interessante Eigenschaften aufweist.

Der DISCOVERY SERVER ist gedacht als die zentrale Komponente in einer Wissensmanagementlösung, die über die K-STATION benutzt wird. Erst dieser Server fügt wesentliche Funktionalität hinzu, und darin ist auch der Grund für die Spannung zu sehen, mit der das *Raven-*

System erwartet wurde. Der Server stellt insbesondere vielversprechende Funktionalität im Bereich der automatischen Klassifikation bereit: Er erkennt automatisch Beziehungen zwischen Dokumenten, Personen und Orten und ist in der Lage, Taxonomien automatisch aufzubauen [Lot01f, Lot01e, Lot01g]. Im einzelnen weist der Server folgende wichtige Funktionen auf:

- Die wichtigste Subkomponente des DISCOVERY SERVERS, der sogenannte *K-map Builder*, dient zur automatischen Kategorisierung bzw. Klassifizierung von Dokumenten, Personen und Orten. Dazu wird eine Wissenskarte (*K-map* genannt) aufgebaut, die die Zusammenhänge zwischen Dokumenten (*Things*), Personen (*People*) und von der K-STATION verwalteten Orten (*Places*) aufdeckt. Aus dieser Karte werden alle weiteren Informationen abgeleitet.

- Der *K-map Builder* verwendet zur Klassifikation Vektoranalysen (unter Nutzung von z.B. *Support Vector Machines*, siehe dazu [Büc01, Die01]) und baut in einer Trainingsphase initiale *Cluster* (also Gruppen) von zusammengehörigen Dokumenten auf. Dies geschieht in zwei Schritten:

  1. Im ersten Schritt werden die eigentlichen Cluster identifiziert und die zum Training verwendeten Dokumente den initialen Clustern zugewiesen.
  2. Im zweiten Schritt werden wiederum automatisch Bezeichnungen für die entstandenen Cluster gesucht und zugewiesen. Dies geschieht durch Auswahl bestimmter, in dem Cluster besonders häufig vorkommender Bezeichnungen, wobei allerdings häufig eine manuelle Nachbearbeitung nötig ist. Einige praktische Erfahrungen sind in [Web01b] beschrieben.

  Diese Technik ermöglicht es, automatisch Taxonomien zu erstellen und erspart damit das in der Praxis gerade bei Neueinführung von Systemen zeitaufwendige manuelle Nacherfassen von Kategorien [Lot01e]. Neue Dokumente werden nach der Trainingsphase durch ähnliche Algorithmen den Themen (Begriffen) der bestehenden Taxonomie zugeordnet. Die Taxonomie läßt sich dann im weiteren Verlauf von berechtigten Personen manuell weiter (um)strukturieren. Diese Änderungen werden vom System beobachtet und sollen die Klassifikation weiter trainieren.

  Die entstandene Taxonomie läßt sich interaktiv (etwa durch das Portal der K-STATION) inspizieren. Durch den intuitiven Zugang und das Stöbern in den Informationen sollen sich neue, unerwartete, aber hilfreiche Zusammenhänge und Informationen entdecken lassen.

- Zusätzlich zu der Taxonomie werden für die Personen, die dem System bekannt sind, Affinitäten zu allen Themen aus der Taxonomie berechnet. Dazu werden mit Hilfe einer komplexen, konfigurierbaren Metrik unter anderem die Inhalte der von den jeweiligen Personen publizierten Dokumente und die Nutzungshäufigkeit (Publikation, Konsumption, Suche) bestimmter Informationen herangezogen. Es ergibt sich so die Möglichkeit, zu einem Thema Experten ausfindig machen zu können, die sogenannte Expertensuche. Die Liste der Experten kann dabei nach der Stärke der jeweiligen Affinität geordnet sein. Die Analyse der Affinitäten ergibt einen Teil des persönlichen Profils jeder Person, der aber auch von ihr selbst modifiziert werden kann, so daß jeder Benutzer die volle Kontrolle über die Ergebnisse hat und so bestimmen kann, was andere davon sehen.

- Aus Affinitäten, Zugriffshäufigkeiten, Alter, Autorenschaft etc. wird für jedes Dokument der Wert (*Document Value*) für jeden Benutzer berechnet, also eine Maßzahl dafür, inwieweit das Dokument für die Person interessant sein könnte. Dies wird für die Durchführung von Suchanfragen und zur Sortierung der Ergebnisse nach Relevanz ausgenutzt.

- Andersherum als bei der Klassifikation werden beim Anzeigen eines Dokuments eine Zusammenfassung sowie die am besten dazu passenden Begriffe der Taxonomie angezeigt, die Aufschluss über weiterführende Informationen und Experten geben können.

Die Funktionalität der Wissenskarte und aller daraus abgeleiteter Informationen ist dabei nicht von dem Portal abhängig; es können auch andere Quellen wie Datenbanken, Web-Seiten, LDAP-Server, LOTUS NOTES-Datenbanken etc. integriert werden. Dafür existiert eine umfangreiche Sammlung von Filtern, den sogenannten *Spidern*, die es ermöglichen, verschiedene Quellen zu erschließen und zu analysieren. Dazu gehören Internet- und Intranet-*Web-Sites*, Diskussionsforen im WWW, Dateisysteme und die Dateien darin (wofür wiederum umfangreiche Filter für diverse gebräuchliche Dokumentenformate existieren), LOTUS DOMINO und QUICKPLACE-Anwendungen und natürlich die Inhalte der K-STATION wie öffentliche Orte, Personen und Dinge. Die zu klassifizierenden Personen können auch aus LOTUS DOMINO-Verzeichnissen oder LDAP-Servern stammen.

Im Gegensatz zu den explizit vorhandenen Modellen für die Objekte wie Dokumente und Kollektionen im HYPERWAVE-System steht das hier beschriebene System diesbezüglich schwach fundiert dar. Ein konzeptuelles Modell der wesentlichen Informationsartefakte (Personen, Orte, Dinge) wird in den einschlägigen Publikationen [Lot01b, Lot01c, Lot01d, Lot01e, Lot01f, Lot01g] nicht präsentiert, die Sicht bleibt oberflächlich und nimmt kaum Bezug auf grundlegende Abstraktionen und Architekturen. Auch die Systemarchitektur wird leider nicht sehr weitgehend erläutert, immerhin scheint im Rahmen der vorhandenen Verteilungsmöglichkeiten der LOTUS NOTES/DOMINO-Server die Verteilung verschiedener Komponenten zur Lastbalancierung möglich zu sein [Lot01a]. Offenbar wird das Datenbanksystem DB2 von IBM sowie der LOTUS DOMINO SERVER in jeweils alleine nicht lauffähigen Basisversionen zur Sicherung der Persistenz der Wissenskarte und auch der Inhalte und Einstellungen der K-STATION eingesetzt [Web01b]. Im Falle der Verteilung werden die Replikationsmechanismen von LOTUS NOTES intensiv verwendet.

Aufgrund des geringen Alters des Systems sind bislang noch wenige Erfahrungsberichte darüber vorhanden, wie sich die Klassifikation in der Praxis bewährt; einen ersten kurzen Eindruck liefert [Web01b]. Die lückenlose automatische Verknüpfung aller Inhalte stellt aber ein vielversprechendes Konzept dar.

3.5 Zusammenfassung

In diesem Kapitel wurde zu Beginn zunächst die historische Entwicklung von Informationssystemen in den letzten 30 Jahren aufgezeigt. Es zeigte sich, daß der gegenwärtig stattfindende Übergang vom Client/Server-Paradigma zum *Network-Computing*-Paradigma auch Systeme für das Wissensmanagement nicht unberührt läßt. Es schlossen sich Untersuchungen genereller Art über im Wissensmanagement verwendete Systeme an, eine Analyse der Abdeckung der Anforderungen an die Funktionalität von Wissensmanagement-Systemen durch verschiedene Klassen von Systemen und eine Untersuchung von drei verschiedenen konkreten Produkten im Bereich der Portale für das Wissensmanagement. Die wesentlichen Erkenntnisse sind:

1. Die Transition zum Paradigma des *Network-Computing* wird durch unternehmensübergreifende Lösungen vorangetrieben, die sich der Techniken, Konzepte und Modelle des Internets bedienen. Gleichzeitig müssen aber immer noch innerhalb von Organisationen Medienbrüche und Systemgrenzen überwunden werden. Moderne Systeme für das Wissensmanagement müssen also einen hoch integrativen Charakter besitzen und sich dazu

der Techniken und Protokolle des Internet bedienen.

2. Die bereits länger bestehenden Systeme wie Datenbanken lassen sich ebenso wie neuartige Entwicklungen (etwa CMS) immer noch gut in ein Klassifikationsschema einordnen, das zwischen prozessorientierten und produktorientierten Systemen unterscheidet. Wichtiger noch als diese Unterscheidung ist die Frage nach der Art der Strukturierung der benutzten Informationsartefakte. Hier scheint die Differenzierung in strukturierte, semistrukturierte und unstrukturierte Informationen der prädestinierte Ansatz zu sein. In der historischen Entwicklung von Informationssystemen erscheinen semistrukturierte Informationen als jüngster Sproß der konzeptuellen Sicht auf die Objekte. Der Umgang mit (stark und homogen) strukturierten Objekten sowie gänzlich unstrukturierten hat bereits eine verhältnismäßig lange Tradition und es gibt deswegen gut ausgereifte Modelle und Systeme dafür.

3. Die Frage nach geeigneten Metamodellen für die beobachteten Informationsartefakte (auch Dokumentenmodell genannt) kristallisiert sich als entscheidendes Kriterium für das Verständnis und die Beschreibung von Systemen heraus, die explizit die Verwaltung von Wissen und die Unterstützung von Wissenstransformationsprozessen sowie Publikationsprozessen zur Aufgabe haben. Elaborierte neue Modelle sind ganz eindeutig tradierten Sichtweisen überlegen (z.B. durch Möglichkeiten zur automatischen Klassifizierung).

4. Die wichtigen Systemklassen der Dokumenten-Management-Systeme und der Content-Management-Systeme spiegeln in der zeitlichen Reihenfolge ihrer Entstehung die Entwicklung des Verständnisses von unstrukturierten und stark strukturierten Informationen hin zu semistrukturierten wider. Die Modelle der jüngeren Systeme weisen eine wesentlich ausgeprägtere Differenzierung auf, die stark von den Ergebnissen der Forschung über Hypermedia-Systeme geprägt ist.

5. Die nahezu einstimmig von allen Beobachtern geäußerte Erwartung ist, daß die Märkte und damit auch die Systeme für Dokumenten-Management und Content-Management konvergieren. Orthogonal zu diesen Veränderungen setzt gemäß des emergierenden Paradigmenwechsels eine starke Konvergenz von öffentlichen Internet-Diensten und organisationsinternen, restriktiv genutzten Intranet-Systemen zu beidseitig und organisationsübergreifend von allen Akteuren genutzten Lösungen ein. Die Schlüsseltechnologie ist dabei in Portalen zu sehen.

6. Die begriffliche Vielfalt im Bereich der Portale ist auf noch große Unsicherheit aufgrund der erst ganz am Anfang stehenden rasanten Entwicklung zurückzuführen. Dennoch zeichnen sich erste Strukturen und Klassen von Portalsystemen ab. Die Differenzierung nach Zielgruppen und Aufgaben leistet dabei gute Dienste.

7. Ein eindeutig als Wissensmanagement-System benennbares Konzept oder System existiert wegen der viel zu stark divergierenden Anforderungen nicht. Eine Vielzahl von verschiedenen – teilweise relativ alten Ansätzen – prägt das Gesamtbild der unter diesem Etikett firmierenden Lösungen. Gemeinsam mit den systemtechnischen Aspekten stehen zur Verwirklichung von Wissensmanagement-Systemen zwei weitere Aspekte gleichberechtigt nebeneinander: die der organisationalen und der kollaborativen Fragen. Eine gute Abdeckung der Systemfacette versprechen die *Organizational Memory Information Systems*. Eine Integration der propagierten Konvergenz von externen und internen sowie Dokumentenmanagement- und Content-Management-Systemen unter der umfassenden Klammer des OMIS in Gestalt eines Unternehmens- oder Wissensportals stellt unter Einbeziehung des restlichen organisatorischen Umfeldes die Vision eines Wissensmanagement-Systems dar (vgl. Abbildung 3.7).

8. Die untersuchten Systeme, die Portale im Kontext des Dokumenten-, Content- und Wissensmanagements anbieten, weisen oberflächlich betrachtet relativ einheitliche Merkmale und Funktionalitäten auf. Die zugrundeliegenden Modelle, Konzepte und Systemarchitekturen variieren allerdings beträchtlich und reichen von elaborierten, reflektiven Objektmodellen und sorgsam entworfenen und untersuchten Architekturen und Protokollen (im Falle von HYPERWAVE) bis zu der pragmatisch orientierten Fortschreibung der tradierten Dateimetaphern ohne erkennbares Dokumentenmodell und tendenziell monolithischen, ad hoc entworfenen Systemarchitekturen (etwa im Falle von MICROSOFT).

9. Eine erfolgreiche Modellierung von Informationsartefakten muß zwei essentielle Einsichten berücksichtigen: Die konsequente Gleichbehandlung der auf einem ausgereiften Dokumentenmodell beruhenden verschiedenen konzeptuellen Entitäten wie Personen, Dokumenten und Verzeichnisse gestattet einerseits den uniformen Zugriff und produktive, effiziente und generische Lösungen. Die ebenfalls durch ein weitreichendes konzeptuelles Modell ermöglichte differenzierte Behandlung der Artefakte und explizite Ausnutzung der verschiedenartigen (semantischen) Beziehungen der Objekte zueinander gestattet andererseits erst den Aufbau nutzbringender Eigenschaften wie automatischer Klassifikation oder semantischer Netze mit einem hohen Mehrwert für die Benutzer.

Aufbauend auf der obigen Zusammenfassung der Erkenntnisse läßt sich die Kernthese dieser Arbeit formulieren: Erstens werden von den bisher existierenden Systemen nur ansatzweise problemadäquate Modelle zur durchgängigen Beschreibung der Informationsartefakte verwendet, die es gestatten, alle benötigten Arten von Informationen aus allen Arten von Systemen bruchlos zu integrieren. Zweitens ist die Abdeckung der für das Wissensmanagement benötigten fachlichen Funktionalität im Bereich der Wissensverteilung und Wissensnutzung durch Internet-basierte Systeme noch relativ gering und könnte mit einem integrierten Ansatz zur Beschreibung der fachlichen Anforderungen und der Nutzung des eben genannten integrierten Modells deutlich umfassender gestaltet werden. Diese These wird folgendermaßen belegt:

- Die produktorientierte Sichtweise (vgl. Abschnitt 3.2.1) bietet jeweils nur Systeme an, die auf genau eine Art von Information (strukturiert, semistrukturiert, unstrukturiert) spezialisiert sind. Die in Abbildung 3.2 aufgeführten prozessorientierten Werkzeuge und Systeme decken nur jeweils Teilbereiche des möglichen Umfangs ab.

- Internet-basierte Systeme für das Wissensmanagement sind selten gezielt für die Aufgaben des Wissensmanagements (Wissensverteilung und Wissensnutzung) konzipiert, sondern entstehen häufig wegen des Marktdrucks durch Hinzufügungen von Funktionalität zu bereits existierenden anderen Systemen, beispielsweise zu Dokumenten-Management-Systemen, Content-Management-Systemen oder sonstigen bestehenden Internet-Informationssystemen.

- Alle existierenden Internet-basierten Wissensmanagement-Systeme (also Wissensportale) haben ein Modell für die verwalteten Informationsartefakte, das entweder wenig problemadäquat ist oder aber nicht fähig zur Integration von heterogenen, bestehenden Informationsbeständen und -systemen ist.

Diese Lücken werden in den folgenden Kapiteln systematisch geschlossen. Die hier gewonnenen Erkenntnisse dienen als Grundlage für das in Kapitel 5 entwickelte abstrakte Modell eines Wissensportals, den in Kapitel 6 vorgenommenen Entwurf einer Portalsoftware und bestimmen den im nächsten Kapitel 4 aufgestellten Anforderungskatalog wesentlich mit.

Kapitel 4

Fachliche und technische Anforderungen an ein Wissensportal

Gib der Alltäglichkeit ihr Recht, und sie wird dir mit ihren Anforderungen nicht zur Last fallen.
– CLEMENS VON BRENTANO

Aufgabe dieses Kapitels ist es, aufbauend auf den in den vorangegangenen Kapiteln gewonnenen Erkenntnissen systematisch fachliche Anforderungen sowie wichtige Systemanforderungen an Portalsoftware für das Wissensmanagement zu sammeln, zu strukturieren und darzustellen. Das Ziel ist dabei, einen umfassenden Katalog aller sich aus den zentralen Wissensmanagementaufgaben der Verteilung und Nutzung ergebenden Anforderungen zu erstellen, der das Ziel der Einbeziehung von Wissen bzw. Informationen sowohl verschiedener Art (explizit vs. implizit) als auch auf den verschiedenen Strukturierungsebenen (produktorientiert vs. prozessorientiert, vgl. Abschnitt 3.2) berücksichtigt. Dies schließt den ersten Teil der in Abschnitt 3.5 identifizierten Lücke.

In diesem Kapitel wird dazu zunächst eine Zielbestimmung in Abschnitt 4.1 vorgenommen, um dann in Abschnitt 4.2 nach einer Diskussion der grundsätzlichen Methodik einige grundlegende konzeptuelle Anforderungen an das zugrundeliegende Modell zu formulieren. Auf dieser Basis wird schließlich unter Bezugnahme auf die zuvor präsentierten Konzepte in den Abschnitten 4.3, 4.4 und 4.5 systematisch ein feingranular strukturierter Katalog von Anforderungen an die Funktionalität (fachliche Anforderungen), die Benutzungsoberfläche und an das Systemmanagement entwickelt. Dieser Katalog reflektiert das gegenwärtige Verständnis derjenigen Anforderungen, die heute an Wissensportale gestellt werden können.

4.1 Zielbestimmung

Der detaillierten Aufstellung von Anforderungen an die Funktionalität vorausgehen muß eine Bestimmung der Art und des generellen Einsatzbereichs und Verwendungszwecks des zu definierenden Softwaresystems: Zu definieren sind die fachlichen Anforderungen und Systemanforderungen an ein Softwaresystem, das ein Internet-Portal zur Unterstützung des Wissensmanagements innerhalb von Organisationen betreibt. Das Portal soll im wesentlichen die beiden Teilprozesse des Wissensmanagements der *Wissensverteilung* und *Wissensnutzung* sowie – implizit – der Wissensbewahrung (vgl. Abschnitt 2.4) unterstützen. Der Schwerpunkt ist dabei auf der

Erfassung, Aufbereitung und Nutzung sowohl von explizitem als auch möglichst weitgehend von implizitem Wissen zu sehen. Gemäß der in Abschnitt 3.3.3 entwickelten Taxonomie ist das zu definierende Portal als Wissensportal (*Knowledge Portal*), also eine Unterart des Unternehmensportals, einzuordnen (vgl. Abbildung 3.5).

Die Zielsetzung umfaßt insbesondere die folgenden Aspekte:

- Abbildung und Unterstützung des Wissenszyklus von Erfassung, Aufbereitung und Nutzung expliziten und impliziten Wissens;
- Integration von sowohl strukturierten, semistrukturierten als auch unstrukturierten Informationen;
- Verwendung von verschiedenartigen, benutzergerechten Leitsystemen und Klassifikationssystemen zur Strukturierung der Informationen wie Wissenslandkarten, Taxonomien (Ontologien), semantische Netze und traditionelle Metaphern wie Datei-Verzeichnissen;
- Berücksichtigung grundlegender mehrdimensionaler Benutzeranforderungen wie der Unterstützung verschiedener Sprachen, verschiedener Medien (WWW, WAP etc.) und verschiedener gleichzeitig unabhängig voneinander benutzbarer Projekte;
- Erfüllung grundlegender für Informationssysteme relevanter Anforderungen wie Verfügbarkeit, Konsistenz, Aktualität, Relevanz, Bedienbarkeit, Wartbarkeit, Erweiterbarkeit und Skalierbarkeit;
- Integration in und von heterogenen, bestehenden Systemlandschaften.

Diese Ziele bedingen einige Basisannahmen, die im folgenden geschildert werden.

4.2 Methodik und Basisanforderungen

Aus den Zielsetzungen für ein Wissensportal unmittelbar Anforderungen abzuleiten, eröffnet einen äußerst breiten Gestaltungsspielraum. Um diesen auf praktikable Art und Weise einzuengen, sind grundsätzlich zwei Vorgehensweisen möglich:

1. Die Erstellung einer umfassenden, klassischen Anforderungsanalyse anhand eines gedachten oder realen Einsatzes im Rahmen eines Softwareentwicklungsprojektes. Dazu würden im Rahmen eines an den objektorientierten Softwareentwicklungsprozeß angelehnten Vorgehensmodells (etwa nach [JBR99, Kah98, Boo96, JCJÖ92]) zunächst im großem Umfang Anwendungsfälle und Aktoren identifiziert und beschrieben werden, um darauf aufbauend ein konzeptuelles Modell und Systeminteraktionen zu definieren. Beides wäre wegen der Komplexität der zu modellierenden Domäne eine äußerst umfangreiche Aufgabe.
2. Die Postulierung einiger Basiskonzepte und Basismodelle unabhängig von konkreten aus Projekten stammenden Anforderungen, die dann einen Rahmen für die weiteren Anforderungen vorgeben und sie einschränken. Diese Basisanforderungen wären aus den Einsichten der vorangegangenen Untersuchungen abzuleiten.

Aufgrund des in dieser Arbeit nicht zu bewältigenden Umfangs der detaillierten Präsentation von Anwendungsfällen und der Tatsache, daß eine solche Darstellung keine signifikanten neuen

Erkenntnisse bringt, wird der zweite Weg beschritten. Die Anwendungsfälle und Aktoren eingeschränkter und ausgewählter Bereiche der Funktionalität, deren Untersuchung und Umsetzung im Rahmen verschiedener Teilprojekte angegangen wurde, sind etwa in [Leh01, Die01, Büc01] ausführlich beschrieben.

Zudem sehen moderne Softwareentwicklungsprozesse eher iterative und inkrementelle Vorgehensweisen vor, die Zyklen enthalten und bei denen die Ergebnisse einer Phase auf die vorherige zurückwirken [Bal01]. Dies spiegelt sich insofern in den entstandenen Anforderungen wider, als daß diese bereits rückgekoppelt die Ergebnisse der Entwurfs- und Implementierungsphase prototypischer Portalsysteme enthalten. Überdies sind die Erkenntnisse erster im kommerziellen Umfeld stattfindender Projekte mit aus diesen Grundlagen entstandenen Systemen enthalten. Einen ersten Erfahrungsbericht liefert etwa [RMS+01]. Weitere Anwendungsbeispiele sind in Kapitel 7 beschrieben.

Gerade die Untersuchung verschiedener Systemklassen und die Analyse einiger konkreter Portalsysteme hat wertvolle Einsichten geliefert, die in Abschnitt 3.5 zusammengefaßt sind. Im einzelnen resultieren daraus folgende Basisforderungen:

- Die in Content-Management-Systemen übliche und bewährte Trennung von Struktur, Inhalt und Gestaltung muß auch hier vollzogen werden. Ebenso muß eine Abtrennung der Funktionalität stattfinden, die sonst oft mit den anderen drei Elementen verwoben ist.
- Die Metapher der drei Basiselemente Personen, Dinge und Orte (*People, Places, Things*) soll verwendet werden.
- Ein konzeptuelles Objektmodell, das die uniforme Behandlung aller Informationsartefakte (Personen, Dinge, Orte etc.) gestattet, soll die Grundlage der Modellierung bilden. Auch wenn dieser Aspekt für den Systementwurf wesentlich größere Bedeutung als für die benutzerorientierte und anforderungsbezogene Sicht der Anwender hat, so birgt er doch bereit gewisse Implikationen für die Anforderungen, die bei der Darstellung der Anforderungen herausgearbeitet werden.
- Die tiefe Erschließung, die Klassifikation und die semantische Vernetzung aller Informationsartefakte ist die Basis für alle wissensbezogenen Strukturierungsansätze. Die bidirektionale Verweisverwaltung (*Linkmanagement*) ist eine dafür notwendige Voraussetzung. Zu den drei Hauptmetaphern der Personen, Dinge und Orte tritt als vierte, gleichberechtigte Dimension der Begriff als konstituierendes Element einer Taxonomie.
- Die von den Benutzern gewohnte Arbeitsweise mit Büroanwendungen etc. muß weitgehend beibehalten und punktuell unterstützt werden, da sonst die Akzeptanz eines Portalsystems fraglich ist. Die komplette Ersetzung der gewohnten Werkzeuge durch im Portal integrierte Informationssysteme und Anwendungen scheint nicht realistisch zu sein, auch wenn das aus Sicht des Informationsmanagements oft wünschenswert erscheint.

Zahlreiche weitere Konzepte lassen sich aus den Ergebnissen aus Abschnitt 3.5 ableiten, die aber keine direkte Relevanz für die Anforderungen haben. Sie werden erneut in Kapitel 6 für den softwaretechnischen Entwurf des Portalsystems aufgegriffen.

Die konkreten Anforderungen werden im folgenden unterteilt in solche, die die Funktionalität aus Benutzersicht fachlich beschreiben, solche, die konkrete Gestaltungsmerkmale der Oberfläche betreffen sowie abschließend solche, die den Systemanforderungen zuzurechnen sind und das Systemmanagement, die Einführung und Anpassung in bestehende Infrastrukturen von Informationssystemen betreffen.

4.3 Fachliche Anforderungen an die Funktionalität

Die funktionalen Anforderungen aus Benutzersicht lassen sich in zahlreiche Unterkategorien aufgliedern. Diese Unterteilung trägt einerseits der übergeordneten Forderung nach der Umsetzung der Metapher von Personen, Orten, Dingen und Begriffen Rechnung und berücksichtigt andererseits die als wichtig angesehenen Eigenschaften moderner Werkzeuge für das Wissensmanagement in strukturierter Weise.

4.3.1 Dokumente

Der Bereich der Dokumente stellt den der „Dinge“ (*Things*) gemäß der o.g. Metapher dar. Der Terminus *Dokument* ist der hier dafür gewählte, da er der Wahrnehmung von (elektronischen) Informationsartefakten als Träger expliziten Wissens am besten entspricht. Als Dokumentenbegriff wird hier ein generalisierter Begriff eingeführt, der alle Arten von explizitem Wissen (Dinge, *Things*) umfaßt und dem strukturierten Begriff aus Abschnitt 3.3.2 sehr ähnlich ist. Das Meta-Modell wird in Kapitel 5 näher beschrieben. Dokumente sind hier also klassische Dateien (beispielsweise Dokumente aus Standard-Büroanwendungen wie Textverarbeitung, Tabellenkalkulation etc.), Notizen, Anmerkungen, Nachrichten (e-Mail, *News*), Diskussionsbeiträge, Meldungen von Nachrichtenagenturen, Verweise auf externe Quellen (Links) oder auch (evtl. wissenschaftliche) Literaturreferenzen auf Bücher und Artikel und haben zusätzliche, strukturierte Informationen. Die wesentlichen Anforderungen an die Dokumente sind somit:

D.1: **Generalisierter Dokumentenbegriff:** Die eben als Grundlage genannte Generalisierung des Dokumentenbegriffs. Alle Dokumente müssen auf die gleiche Art und Weise und ohne Unterschied erfaßt, abgelegt und recherchiert werden können. Erreicht wird damit eine einheitliche Sicht auf alle Dokumente einer Organisation, die die Grundlage für die folgenden Anforderungen ist.

D.2: **Strukturierte Ablage:** Alle Dokumente müssen als primäres Ordnungskriterium in einer Verzeichnisstruktur abgelegt werden (siehe dazu auch Abschnitt 4.3.3). Durch diese der gewohnten Arbeitsweise entsprechenden Sicht der Benutzer ist der Aufwand zur Umstellung auf die Arbeit mit dem Portal geringer. Durch die Nutzung des Verzeichnisses wird sichergestellt, daß jedes Dokument zumindest auf einem Weg erreichbar bleibt. Beim Einstellen von Dokumenten durch die Benutzer müssen die wichtigsten Metainformationen erfaßt werden (s.u.). Eine Unterstützung durch automatisch generierte Vorgaben, insbesondere für Stichworte und Bemerkungen, ist eine essentielle Funktionalität, die für eine gleichbleibend hohe Informationsqualität sorgt. Diese Aspekte werden ausführlicher in Abschnitt 4.3.7 behandelt.

D.3: **Systemdefinierte Metainformationen:** Jedes Dokument, gleich welcher Art, besitzt eine Reihe von Metaattributen, die es neben seinem Inhalt, der von der speziellen Art des Dokuments abhängt, näher beschreiben. Dazu zählen insbesondere:

- der oder die Autoren, die aus der Menge der vorhandenen Personen stammen müssen (vgl. Abschnitt 4.3.2);
- der Bearbeiter, der nicht identisch mit einem der Autoren sein muß. Dies ist diejenige Person, die ein Dokument eingepflegt hat – eine Information, die oft hilfreich ist. Ebenso kann die Person festgehalten werden, die die Metaattribute zuletzt verändert hat;

- Erstellungs- und Änderungsdatum des eigentlichen Dokuments sowie der Zeitpunkt des Einpflegens in den Bestand des Portalsystems. Optional könnte ein Gültigkeitszeitraum angegeben werden, nach dem das Dokument entweder automatisch entfernt wird oder zumindest als veraltet gekennzeichnet wird und ggf. an eine andere Stelle innerhalb der Verzeichnisse verschoben wird;
- Stichworte oder Schlagworte, die das Dokument näher beschreiben. Diese stammen ausnahmslos aus der Begriffstaxonomie (vgl. Abschnitt 4.3.4);
- eine Zusammenfassung oder Bemerkung über den Inhalt (ggf. ein *Abstract*), die näheren Aufschluß über den Inhalt gibt, ohne daß das Dokument selbst ganz gelesen werden muß. Die Verwendung lohnt sich natürlich nur bei umfangreichen Dokumenten und ist deshalb optional;
- dynamische Daten wie Anzahl der lesenden Zugriffe insgesamt. Diese statistischen Daten können dazu verwendet werden, besonders relevante Dokumente zu identifizieren und *Hot Spots* auf Wissenslandkarten zu entdecken (vgl. Anforderung E.4 und B.4). Eine Gesamtübersicht über die beliebtesten Dokumente läßt sich so ebenfalls erzeugen;
- optional weitere Informationen wie Größe und sonstige dokumentartspezifische Angaben.

Ein generischer Dokumententyp, die sogenannte Basisdokumentenklasse, die die genannten Metainformationen bereithält, muß bereits im System integriert sein.

D.4: Benutzerdefinierte Metainformationen: Neben den oben aufgeführten Metainformationen müssen weitere Meta-Attribute definiert werden können, die dann entweder für alle Dokumentenarten vorhanden sind oder nur für spezielle Unterklassen. Dies gestattet die Erweiterung für spezielle, zum Entwurfs- und Implementierungszeitpunkt noch unbekannte Anforderungen. Diese Attribute müssen z.B. bezüglich der Suchfunktionen genauso gehandhabt werden können wie die fest eingebauten. Für Metaattribute müssen zumindest eine Reihe von Basisdatentypen angeboten werden wie Ganzzahl, Text, Datum oder (typisierter) Verweis auf weitere Informationsartefakte des Portals. Zusätzliche Multimedia-Datentypen wie „Bild“ können je nach Anwendungsgebiet sinnvoll sein.

D.5: Definition von Dokumentenklassen: Die Definition von neuen, spezielleren Arten von Dokumenten (d.h. Dokumentenklassen) mit zusätzlichen Attributen (Metainformationen) muß möglich sein. Eine objektorientierte Modellierung, die dem Paradigma von Klassen und Vererbung gehorcht, kann dazu ausgenutzt werden, neue Dokumentenklassen von bestehenden ableiten zu können und so ihre Definitionen wiederverwenden und spezialisieren zu können. Die Exemplare (Dokumente) der neu definierten Dokumentenklassen müssen unterschiedlos zu den bestehenden behandelt werden können.

D.6: Zugriffsrechte: Für Dokumente muß festlegbar sein, welche Personen (Benutzer oder Benutzergruppen) welche Zugriffsrechte auf das Dokument besitzen. Zu unterscheiden ist zwischen dem Leserecht (ob das Dokument einsehbar ist, zumindest sichtbar oder womöglich die Existenz verborgen ist) und dem Schreibrecht, also dem Bearbeitungsrecht für das Dokument und seine Metainformationen. Als Einheit der Rechtevergabe werden die Verzeichnisse verwendet, die die Dokumente enthalten (vgl. Anforderung V.4).

D.7: Versionierung: Alle Dokumente müssen versionierbar sein, d.h. daß die Historie der Änderungen jederzeit eingesehen und nachvollzogen werden kann. Dabei muß der Zugriff auf jede alte Version mit allen alten Metadaten möglich sein. Die konsequente Durchsetzung dieses Prinzips kann dazu führen, daß sogar gelöschte Dokumente wiederherstellbar sind

(durch Einsicht der letzten Version); dies kann aber aus pragmatischen Gründen (Datenvolumen, Speicherplatz) nicht immer möglich sein. Die Umsetzung dieser Funktion kann also nicht zwingend gefordert werden.

D.8: **Status und Sperren:** Jedes Dokument muß in jeder Version einen Status aus einer frei definierbaren Menge von Zuständen besitzen, die Aufschluß über den Fortschritt seiner Bearbeitung geben. Damit verbunden muß die Möglichkeit sein, in bestimmten Zuständen das Dokument exklusiv durch einen Benutzer zu sperren, d.h. daß nur er alleine ungeachtet anderer Personen mit den gleichen Privilegien das Dokument bearbeiten darf. Der typische Lebenszyklus von Dokumenten (Bearbeitung in Vorversion, nicht freigegeben, zu revidieren, freigegeben etc.) muß unterstützt werden. Dazu können Funktionen aus dem Bereich des Workflows (vgl. Abschnitt 4.3.9) herangezogen werden.

D.9: **Traditionelle Arbeitsweise:** Die traditionelle, von den Benutzern gewohnte Arbeitsweise mit Büroanwendungen wie Textverarbeitung, Tabellenkalkulation etc. muß weiterhin möglich sein. Die erzeugten Dokumente müssen möglichst bruchlos und komfortabel in das Portalsystem importiert und exportiert werden können; beispielsweise durch direktes Publizieren oder Abspeichern in das Portal und direkte Bearbeitungsmöglichkeiten der im Portal aufgefundenen Dokumente ohne große Umwege (Herunterladen, Öffnen etc.).

Eine entscheidende Eigenschaft des Portalsystem ist in dem Aufbau des Dokumentenmodell zu sehen. Die Details (die genaue Ausprägung der Metadaten, die Funktionalität und der Mechanismus der Vererbung und der Definition von Metaattributen) werden erst im Entwurf des Systems genauer spezifiziert werden können; das abstrakte Dokumenten-Modell wird in Abschnitt 5.1 definiert.

4.3.2 Personen und Gruppen

Der Bereich der Personen deckt den zweiten Bereich der Metapher ab. Eng mit den Personen gekoppelt sind die Gruppen als Strukturierungsmittel und Einheit der Rechtevergabe. Durch Informationen über Personen, die Wissensträger sind, wird in dem Wissensportal ihr implizites Wissen beschrieben und somit (teilweise) explizit gemacht.

P.1: **Uniformes Personenmodell:** Alle Personen, die in verschiedensten Rollen innerhalb des Systems erscheinen, müssen durch das gleiche Personenmodell beschrieben werden. Dies umfasst gleichermaßen die registrierten internen und externen Benutzer des Systems, anonyme Gastbenutzer, externe Personen wie Kunden, Partner und Lieferanten, ehemalige Mitarbeiter ohne Zugang zum System, Bewerber, Autoren von Dokumenten externen Ursprungs und alle weiteren in der Praxis erforderlichen Rollen. Dies vermeidet die Doppelerfassung von Personen etwa sowohl als registrierter Benutzer als auch als Autor von Dokumenten. Die dahinter stehenden Überlegung ist, daß jede real existierende Person durch genau ein Informationsartefakt vom Typ Person im System repräsentiert wird und niemals durch mehrere. Die Differenzierung der selbstverständlich vorhandenen und zu berücksichtigenden Unterschiede geschieht durch Gruppen und Rollen.

P.2: **Gruppen und Rollen:** Durch Gruppen und Rollen wird sowohl ein rollenbasiertes Berechtigungskonzept als auch die nötige Differenzierung zwischen den einzelnen Arten von Personen ermöglicht. Jede Person muß Mitglied in beliebig vielen Gruppen sein können; auch Gruppen müssen Mitglieder in Gruppen sein können und so eine Hierarchie von Gruppen aufbauen können. Drei ausgezeichnete Gruppen muß es geben: Die Gruppe aller existierenden Personen, die der zur Benutzung des Systems überhaupt berechtigten Personen

sowie eine Gruppe von Administratoren, die automatisch umfassende Rechte für alle Operationen besitzen. Die Art der Mitgliedschaft einer Person in einer Gruppe wird durch seine Rolle beschrieben. Dazu muß eine frei definierbare Menge von abstrakten Rollenkonzepten existieren, die dann den konkreten Rollen zugewiesen wird. Denkbare Rollen sind etwa Leiter einer Gruppe, Administrator einer Gruppe und einfache Mitgliedschaften, eventuell gestaffelt nach intendierten Rechten innerhalb der Gruppe wie (passiven) Lesern und (aktiven) Schreibern. Die angesprochenen Rechte für Verzeichnisse müssen unter Bezugnahme auf diese Rollen und Gruppen definiert werden (vgl. Anforderung P.6 und V.4).

Eine Gruppe als solche stellt ein selbständiges Informationsartefakt wie auch eine Person oder ein Dokument dar und muß demzufolge von orthogonal über alle Objekte angebotenen Funktionen wie z.B. Suche und Klassifikation behandelt werden können.

Die einheitliche Gruppenverwaltung gestattet so die integrierte Beschreibung der Aufbauorganisation in Abteilungen und Arbeitsgruppen sowie die Unterstützung von Projektgruppen und informellen Interessengruppen (*Communities of Practice*). Jede Gruppe muß dazu als Gemeinschaft aufgefaßt werden können und so die kollaborative Gruppenarbeit unterstützen. Eine Gruppe muß dazu u.a. als Verteiler für Nachrichten an die Mitglieder dienen. Weitere Funktionen dieser Art fallen in den Bereich der Gemeinschaftsfunktionen (vgl. Abschnitt 4.3.8).

P.3: Personenprofil: Jede Person muß eine Reihe von aussagekräftigen Attributen besitzen, die insgesamt das Profil einer Person bilden. Die Sichtbarkeit und der Detaillierungsgrad der Personenprofile muß für unterschiedliche Gruppen getrennt einstellbar sein, um dem Schutz personenbezogener Informationen Rechnung zu tragen. Die informationelle Selbstbestimmung muß dabei der grundlegende Gedanke sein und jeder Person ermöglichen, alle über sie selbst vorhandenen Informationen einzusehen und nötigenfalls korrigieren zu können.

Wichtige Informationen des Profils umfassen etwa:

- persönliche Daten wie Name, Vorname, Geburtstag, Anschrift und private Kommunikationsverbindungen;
- dienstliche Anschriften und Kontaktinformationen (e-Mail, Telefonnummern, separate *Homepages* etc.) sowie Informationen über Aufgabe und Stellung in der Organisation;
- Verweise auf alle von diesem Benutzer generierten Dokumente und Mitgliedschaften in Gruppen;
- Beschreibungen von Fähigkeiten und Kenntnissen. Diese dem Kompetenzmanagement zuzurechnenden Funktionen werden beispielhaft in Abschnitt 7.3 besprochen und weiter unten unter dem Punkt Kompetenzmanagement vertieft.
- Informelle Informationen wie Selbstbeschreibungen und Fotos.

Die Profile aller Mitarbeiter einer Organisation erfüllen so die Funktion eines Mitarbeiterverzeichnisses, also den häufig für Wissensmanagement-Systeme geforderten *Yellow pages* zum Finden von Experten. Die Expertensuche (vgl. auch Anforderung S.2) selbst muß durch weitere Funktionen der Wissensentdeckung (vgl. Abschnitt 4.3.7) unterstützt werden.

P.4: Kompetenzmanagement: Diese als *Skill Management* bezeichnete Funktion muß zumindest eine Selbsteinschätzung der Mitarbeiter über Fähigkeiten und Kenntnisse auf vordefinierten Skalen gestatten. Die Publikation dieser Bewertungen fällt natürlich eindeutig in den Rahmen der informationellen Selbstbestimmung. Neben diesen expliziten Methoden

der Bewertung muß für ein gutes Kompetenzmanagement die automatische Klassifikation von Dokumenten und Begriffen herangezogen werden. Dies wird in Abschnitt 4.3.4 und 4.3.7 näher beschrieben.

Zusätzlich zu Selbstbewertungen und automatischen Analysen kann die gezielte Verwaltung von Wissensprofilen der Personen durch Erfassung von abgeleisteten Fortbildungsmaßnahmen, stattgefundenen Trainingseinheiten und angebotenen Kursen zur Strukturierung und Analyse des Wissens der Mitarbeiter genutzt werden [Vog00].

Darüber hinaus können im Sinne einer Bewerberdatenbank die in stattgefundenen Bewerbungsgesprächen oder Einstellungstests erkannten Fähigkeiten und Kenntnisse von Personen zur gezielten Akquisition von Wissensträgern dienen. Hier sind natürlich verschärft datenschutzrechtliche Bestimmungen zu beachten, die u.U. nur eine eingeschränkte Dauer der Speicherung zulassen. In Abschnitt 7.3 wird als Anwendungsbeispiel ein web-basiertes Online-Testsystem vorgestellt.

P.5: Integration bestehender Personalsysteme: Gerade im Umfeld bestehender betrieblicher Informationssysteme und Personalmanagementsysteme ist eine wichtige Anforderung, Informationen ggf. sogar automatisiert aus bestehenden Systemen zu übernehmen, um die fehleranfällige und aufwendige doppelte Erfassung zu vermeiden. Diese Problematik gehört eher in den Bereich des Systemmanagements; sie wird hier dennoch genannt, da eine bruchlose Nutzung der Personen wie oben beschrieben auch im Falle der (dann transparenten) Anbindung anderer Systeme zu fordern ist. Es darf also kein Unterschied sichtbar sein, wenn die Benutzerdaten z.B. automatisch aus einem LDAP-Server importiert und aktualisiert werden, oder sie direkt in dem Portalsystem verwaltet werden.

In vielen Systemen werden einerseits die Benutzer und die sonstigen Personen getrennt modelliert, und andererseits nicht als zu Dokumenten etc. gleichberechtigtes Konzept betrachtet.[1] Durch die uniforme Sicht auf Personen und Gruppen werden die nachfolgend beschriebenen Funktionen – gerade später im Entwurf – erheblich durchgängiger gestaltbar und vielfältiger anwendbar. Die Autorenschaft von Dokumenten ist ein erstes Beispiel, wo die uniforme Repräsentation von internen und externen Personen zu einfacheren Lösungen führt.

4.3.3 Verzeichnisse

Die Verzeichnisse stellen das primäre Leitsystem zur strukturierten Ablage und zum Zugriff auf die Dokumente dar. Sie erfüllen damit die Funktion der Orte (*Places*). Der Terminus *Verzeichnis* wurde gewählt, weil er der gewohnten Metapher des Verzeichnisses, das Dokumente enthält, sehr ähnlich ist.

V.1: Generalisierter Verzeichnisbegriff: Der Begriff des Verzeichnisses umfaßt alle möglichen von dem Portalsystem integrierten Informationsquellen und -senken, z.B. Ordner in einem Dateisystem oder Ordner bzw. Verzeichnisse, die in Dokumenten-Management-Systemen oder Content-Management-Systemen vorhanden sind. Durch einen entsprechenden Entwurf und eine geschickte Implementierung müssen alle Quellen in die integrierte Verzeichnissicht unterschiedlos eingebettet werden, wobei prinzipiell jedes Verzeichnis nebst Inhalten (Dokumenten) aus einer anderen Informationsquelle (z.B. Datenbanken, CMS oder DMS) stammen können muß. Dies gestattet die uniforme Sicht auf alle organisationsweit vorhandenen Informationsartefakte und ermöglicht orthogonale Dienste wie Indexierung, Suche und Klassifikation der enthaltenen Dokumente.

[1] Verglichen mit dem Entwurf von Programmiersprachen erinnert diese Modellierung an die Diskussion um Objekte erster Klasse. Siehe dazu etwa [Mat00].

V.2: Speicherung von Dokumenten: Neben der Integration von verschiedensten Informationsquellen in einer uniformen Verzeichnissicht muß ein Wissensportal eine selbständige Lösung zur Speicherung von Dokumenten besitzen.

V.3: Struktur: Die Verzeichnisstruktur muß – im Rahmen der Möglichkeiten der zugrundeliegenden Systeme – beliebig gestaltet werden können. Dazu zählen beliebige Namen und beliebige Schachtelungstiefen. Jedes Dokument muß zumindest an einer Stelle in den Verzeichnissen erscheinen. Ist dies sinnvoll und gewünscht, so darf ein und dasselbe Dokument in verschiedenen Verzeichnissen erscheinen. Zyklen innerhalb der Verzeichnisstruktur sollten der Übersichtlichkeit halber nicht gestattet sein.

V.4: Attribute von Verzeichnissen: Ebenso wie Dokumente müssen Verzeichnisse eine Reihe von Metainformationen oder Attributen besitzen, die sie neben dem Inhalt selbst näher charakterisieren. Diese Attribute umfassen:

- Name und textuelle Beschreibung des Verzeichnisses sowie Erstellungsdatum und das Datum der letzten Änderung;
- Besitzer und Rechte in Form von Personen und Gruppen. Für jedes Verzeichnis muß getrennt einstellbar sein, welche Personen, Gruppen oder Rollen zu folgenden Aktionen berechtigt sind: Einsehen des Verzeichnisses, d.h. Anzeigen der Inhalte und der Metainformationen, Ändern der Metainformationen, Anlegen von Dokumenten innerhalb des Verzeichnisses und Ändern der Dokumente des Verzeichnisses. Das Anlegen ist sinnvollerweise vom Ändern zu unterscheiden, damit auch Benutzer mit geringen Rechten Verzeichnisse praktisch als Briefkasten verwenden können, und das Ändern der Verzeichnisse und ihrer Hierarchie vom Ändern der Dokumente, damit die Strukturierung ausgewählten Personengruppen vorbehalten bleibt.
- Über die Hierarchie kumulierte Angaben über die Anzahl und Größe der enthaltenen Dokumente. Dies gestattet den schnellen Überblick, an welchen Stellen viele Informationen vorhanden sind. Ein an sich unverständliches Manko der heute üblichen Dateisysteme ist, daß sie in der Regel keine Statistik über die kumulierte rekursiv ermittelte Anzahl und Größe der Inhalte führen. Dies müssen die Verzeichnisse eines Portalsystems auf jeden Fall leisten.

V.5: Operationen auf Verzeichnissen: Auf Verzeichnissen müssen eine Reihe von Operationen möglich sein:

- Ändern der Metainformationen wie Name, Beschreibung etc.;
- Anlegen eines neuen Unterverzeichnisses;
- Verschieben eines Verzeichnisses in ein anderes Verzeichnis;
- Löschen des Verzeichnisses inklusive aller enthaltener Dokumente und aller rekursiv enthaltener Unterverzeichnisse.
- Einstellen von neuen Dokumenten, Lesen von vorhandenen Dokumenten und Überschreiben von vorhandenen Dokumenten mit neuen Versionen.

In Abhängigkeit von den den einzelnen Verzeichnissen zugrundeliegenden Informationsquellen oder -senken können einzelne Operationen nicht möglich sein; so böte die Integration von schreibgeschützten Datenträgern wie CD-ROMs natürlich keine Änderungsmöglichkeiten an.

V.6: Definition von Metainformationen: Genau wie für Dokumente müssen auch für Verzeichnisse neue, zusätzliche Metainformationen (Attribute) definierbar, suchbar und wartbar sein (vgl. Anforderung D.4). Die Definition neuer, abgeleiteter Verzeichnisklassen ist hingegen nicht sinnvoll, um eine einheitliche Verzeichnisstruktur zu erhalten.

V.7: Import: Gerade in der Anfangsphase der Nutzung eines Wissensportals muß der Massenimport von Dokumenten in die Verzeichnisse unterstützt werden. Eine typische Situation bei einer erfolgreichen Einführung ist, daß entweder bereits existierende Dateiserver-Inhalte großen Umfangs komplett unter Kontrolle des Portals gestellt werden sollen, oder daß die einzelnen Benutzer beginnen, ihre bislang auf dem persönlichen Rechner abgelegten Dokumente massenweise in die Verzeichnisse zu verlagern. Für beide Szenarien muß entsprechende Unterstützung vorhanden sein: Für den ersten Fall ist sie eher im Bereich des Systemmanagements zu suchen, während die individuelle Unterstützung beispielsweise das Hochladen ganzer Verzeichnisse oder von in Archiven gepackten Verzeichnissen ermöglichen könnte.

V.8: Persönliche Verzeichnisse: Jeder Benutzer muß ein persönliches Verzeichnis besitzen, das entsprechende Rechte besitzt und ihm dazu dient, nicht für die Öffentlichkeit bestimmte Dokumente abzulegen (vgl. dazu auch Abschnitt 4.3.5).

Das System der Verzeichnisse stellt ein mächtiges Strukturierungswerkzeug für Inhalte aller Art aus diversen Informationsquellen- und senken dar. Zusammen mit generischen Dokumentenmodellen ergibt sich so eine hohe Flexibilität im Umgang mit Informationsartefakten.

4.3.4 Begriffe

So fängt denn alle menschliche Erkenntnis mit Anschauungen an,
geht von da zu Begriffen und endigt mit Ideen.
– IMMANUEL KANT

Das Leitsystem der Verzeichnisse leistet nur eingeschränkte Dienste zur Wissensverteilung und -nutzung. Das wesentliche Konzept, um Dokumente, Personen und Verzeichnisse zu klassifizieren und inhaltlich zu erschließen, ist eine in Form eines Begriffsnetzes aufgebaute Taxonomie. Neben den drei essentiellen Artefakten der Personen, Orte und Dinge wurde der Begriff als Element der Taxonomie als viertes eingeführt. Durch aus Begriffen aufgebaute Taxonomien läßt sich die Terminologie, die in einer Organisation gebraucht wird, strukturiert abbilden. Die Taxonomie erfüllt zugleich die Funktion eines unternehmensweiten Glossars und einer Wissenslandkarte. Nicht zu verwechseln ist diese Taxonomie mit dem semantischen Netz, das sich aus den zahlreichen Verknüpfungen aller Informationsartefakte untereinander durch die zahlreichen speziellen Beziehungen ergibt. Folgende Anforderungen an die Begriffe bestehen:

B.1: Taxonomie: Die Begriffe müssen in erster Linie eine Taxonomie bilden, die Oberbegriffe und Unterbegriffe kennt. Jeder Begriff muß eine beliebige Zahl von Unterbegriffen besitzen können, und darf höchstens einen ihm übergeordneten Begriff besitzen. Das Begriffsnetz bildet in dieser Beziehung einen gerichteten, azyklische Graphen. Dabei darf es mehrere Wurzeln geben; das wären alle Begriffe ohne Oberbegriff.

Neben dieser strikten Baumstruktur zum Ausdruck einer Taxonomie ist es oft wünschenswert, vielfältigere Beziehungen wie etwa überlappende Hierarchien in demselben Netz zum Ausdruck bringen zu können. Rein assoziative oder Teil-von-Beziehungen lassen sich nicht durch Vererbungsbeziehungen ausdrücken. Zu fordern sind also weitere Arten von Beziehungen zwischen den Begriffen.

Neben der Unter/Oberbegriffsbeziehung muß eine zusätzliche Spezialisierungs- bzw. Generalisierungsbeziehung existieren, die es gestattet, einen Begriff beliebig viele zusätzliche Begriffe spezialisieren zu lassen und seinerseits beliebig viele Spezialisierungen durch weitere Begriffe erfahren zu lassen. Hier sind auch Zyklen erlaubt. Diese Anforderung trägt

der Tatsache Rechnung, daß es Mehrfachvererbung geben kann, daß also ein Begriff unter mehrere Oberbegriffe fallen kann. Dennoch bleibt ein ausgezeichneter Oberbegriff bestehen.

Zusätzlich zu diesen beiden Beziehungen muß eine reine Assoziations- oder Verweisbeziehung zwischen beliebigen Begriffen etabliert werden können. Hier muß jeder Begriff beliebig viele andere Begriffe referenzieren können. Die Verweise sind dabei als bidirektionale Referenzen ohne bevorzugte Richtung anzusehen.

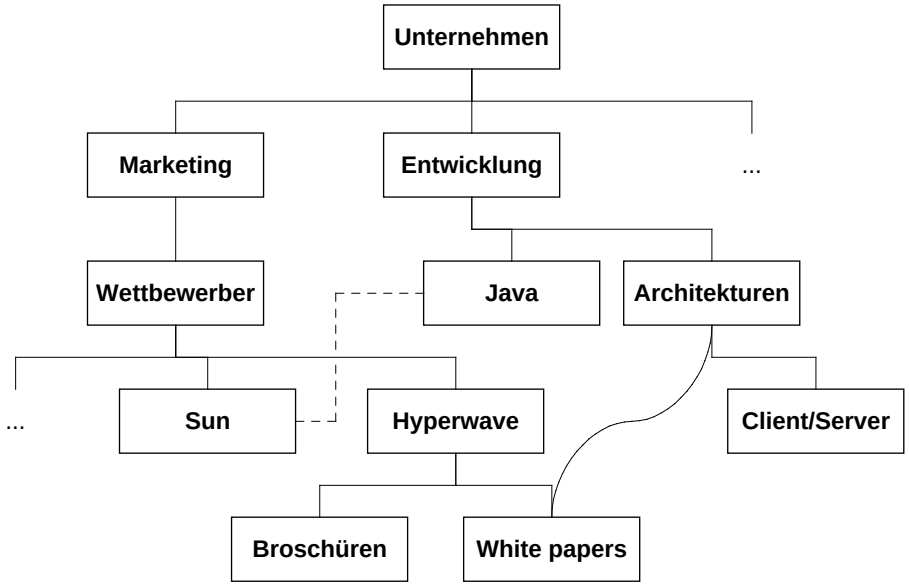

Abbildung 4.1: Beispiel einer generalisierten Taxonomie

Durch diese drei Beziehungen lassen sich aussagekräftige Taxonomien formulieren. Um die Nützlichkeit der unterschiedlichen Arten von Beziehungen zu illustrieren, ist in Abbildung 4.1 ein Beispiel wiedergegeben. Die Ober/Unterbegriffsbeziehung ist hier durch gewinkelte Verbindungen dargestellt, die zusätzlichen Spezialisierung durch gekrümmte und die Assoziation durch gestrichelte Verbindungen. Das Beispiel macht deutlich, daß ein Begriff (*White papers*) primär unter den Oberbegriff des entsprechenden Wettbewerber eingeordnet werden muß, aber zusätzlich auch unter den Begriff Architektur fallen kann. Zusätzlich besteht eine – naheliegende – Assoziation zwischen dem (hier als Wettbewerber klassifizierten) Unternehmen SUN und der unter Entwicklung angesiedelten Sprache JAVA.

B.2: Begriffe: Ein Begriff muß neben den Beziehungen zu anderen Begriffen folgende Merkmale und Attribute aufweisen:

- Jeder Begriff muß einen ihn gut beschreibenden, kanonischen Namen haben und über eine Liste von Synonymen verfügen, die die Identifikation des Begriffes etwa bei Suchanfragen erleichtern.
- Falls bei der automatische Klassifikation (vgl. Abschnitt 4.3.7) Dokumente in verschiedenen Sprachen anfallen, so müssen die Namen und Synonyme multilingual erfaßt sein, d.h. in jeder benötigten Sprache angegeben sein. Nicht nur in international tätigen Organisationen, sondern bereits in kleinen und mittleren Unternehmen ist es mittlerweile eher der Normalfall, daß Dokumente in verschiedenen Sprache nebeneinander existieren, insbesondere in deutsch und englisch. Alleine die Dominanz ameri-

kanischer Softwarehersteller führt dazu, daß Dokumentation oft nur (oder zuerst) auf Englisch erhältlich ist.

Ein Problem dabei bleibt allerdings, daß viele Begriffe nicht direkt übersetzbar sind und daß aufgrund kultureller Unterschiede sogar Taxonomien in anderen Sprachen anders aufgebaut sein können.

- Eine ausführliche, textuelle Beschreibung des Begriffs in zumindest einer Sprache. Eventuell sollte eine weitergehende Erläuterung (etwa eine exakte Begriffsdefinition im Stile von Nachschlagewerken) angeboten werden, die zur Illustration mit Bildern versehen werden kann. Letztere wären wiederum als spezielle Dokumentenklasse zu modellieren.
- Für jeden Begriff müssen Metainformationen wie Autor und Erstellungs- bzw. Änderungsdatum festgehalten werden.

B.3: Klassifikation: Die Taxonomie muß dazu genutzt werden können, alle Arten von anderen Informationsartefakten wie Dokumente, Personen und Verzeichnisse zu kategorisieren. Dadurch lassen sich alle Informationen zu einem Thema, also die Träger expliziten und impliziten Wissens (Dokumente und Personen) leicht lokalisieren.

Dazu müssen bidirektionale Verknüpfungen zwischen einem Begriff und einer beliebigen Anzahl von im Wissensportal verfügbaren Artefakten möglich sein, die zur Anzeige der assoziierten Begriffe bzw. Objekte auf beiden Seiten genutzt werden. Die bereits beschriebene Verschlagwortung von Dokumenten (vgl. Anforderung D.3) muß genau über diesen Mechanismus geschehen.

B.4: Wissenslandkarte: Die thematische Orientierung gerade in großen Taxonomien ist durch rein textuelle Präsentation meistens nicht hinreichend gewährleistet. Die graphische Präsentation in Form von Wissenslandkarten leistet einen intuitiveren Zugang und ermöglicht zugleich die Darstellung wesentlich größerer Ausschnitte aus dem Netz, was zur besseren Orientierung führt. Ein graphischer Begriffsnavigator muß also als zusätzliche Komponente vorhanden sein.

Mit den Möglichkeiten der graphischen Darstellung der Netze lassen sich leicht weitere Informationen einbinden: Durch entsprechende Farbgebung lassen sich beispielsweise Begriffe nach Anzahl der vorhandenen Beziehungen zu Dokumenten oder Personen auszeichnen, die Häufigkeit des Besuchs durch andere Benutzer oder Veränderungen (neue Begriffe, Änderungen an bestehenden) innerhalb bestimmter zeitlicher Intervalle. So kann leicht ein guter und schneller Überblick über Aktivitäten und Brennpunkte des Geschehens vermittelt werden [CMS99].

B.5: Rechte und Pflege: Ebenso wie für Dokumente und Verzeichnisse ist eine Rechtevergabe für Erstellung und Pflege der Begriffsnetze erforderlich. Es muß zwischen folgenden Privilegien unterschieden werden:

- Einsicht in alle Begriffe ohne die Möglichkeit der Veränderung. Dieses Privileg sollten alle Benutzer automatisch haben.
- Erstellung und Pflege der Begriffe und ihrer Struktur. Dazu gehören alle Veränderungen der Struktur (Ober- und Unterbegriffe, Spezialisierungen und Generalisierungen sowie Assoziationen) und der Attribute eines Begriffes.
- Definition von Beziehungen zwischen Begriffen und anderen Informationsartefakten. Dieses Recht ist unabhängig von dem Recht zur Pflege der Begriffsnetze selbst, da diese dadurch nicht modifiziert werden.

Gerade die Erstellung und Pflege sollte nur einem begrenzten Personenkreis ermöglicht werden, um Wildwuchs und unkoordinierte Veränderungen der Taxonomie verhindern zu können.

Ein sich direkt aus der Pflege ergebendes Problem ist das der Versionierung: Da zumindest Dokumente versionierbar sein müssen (vgl. Anforderung D.7), muß strenggenommen auch die Begriffstaxonomie versioniert werden, um den ursprünglichen Zustand rekonstruieren zu können. Dieses Problem zeigt sich hier besonders deutlich, ist aber prinzipiell bei allen Beziehungen zu Dokumenten vorhanden. Da dieser Themenkomplex außerordentlich umfangreich ist und die – ggf. rechtssichere – Versionierung und Archivierung nicht zu den zentralen Aufgaben eines Wissensportals zählt, wird er für diese Arbeit ausgeklammert bzw. als optionale Eigenschaft der vom Portal zu integrierenden Systeme betrachtet.

B.6: **Interoperabilität:** Da es zahlreiche Ansätze zur Strukturierung und zum Austausch von Wissen (eigentlich von Informationen) gibt, müssen die Begriffsnetze auf einfache Art und Weise über standardisierte oder gebräuchliche Austauschformate transportierbar sein.

Ein für die diese Forderung besonders interessanter Standard, der sich beginnt, im Umfeld der Wissensrepräsentation, von semantischen Netzen, Metadaten und unter dem Schlagwort „*Semantic Web*“ [BLHL01] zu etablieren, ist der der *Topic Maps* [LD01, RP99, Ksi99, Pep99, Rat99a, Rat99b], für die bereits eine XML-Variante (XTM) existiert [Top01]. *Topic Maps* gestatten die Repräsentation und den Austausch von semantischen Netzen, eine also insbesondere für die Begriffstaxonomie und die zahlreichen Beziehungen der Klassifikation interessante Möglichkeit der Interoperabilität.

Die Benutzung komfortabler externer Werkzeuge zur Erstellung und Pflege semantischer Netze wie z.B. von THE BRAIN [The01] muß unterstützt werden, indem deren Austauschformate importierbar und exportierbar sind.

Die orthogonale Verschlagwortung und damit die Erschließung aller Informationsartefakte des gesamten Portals stellt eine wesentliche Voraussetzung für Wissensverteilung und Wissensnutzung dar. Die Begriffe sind somit ein Schlüsselmechanismus des Portals. Ebenso wie für das Dokumentenmodell gilt, daß der genaue Aufbau der Begriffe und die Funktionsweise der Verknüpfungen eine wesentlich den Entwurf beeinflussende Aufgabe darstellt.

4.3.5 Personalisierung

Die Personalisierung gilt als eine der wichtigsten Eigenschaften oder Charakteristiken eines Portals. Gerade ein Wissensportal muß die persönlichen Präferenzen, Meinungen und Interessen der Benutzer berücksichtigen. Der Präfix „mein“ bzw. „*my*“ für alle möglichen Funktionen und Dienste ist ein typisches Zeichen der Personalisierung [MPR00]. Zu fordern sind daher eine Reihe von Funktionen:

M.1: **Annotation:** Jeder Benutzer muß die Möglichkeit haben, jedes Informationsartefakt mit privaten, nur für ihn sichtbaren Notizen annotieren zu können, um für ihn persönlich wichtige Bemerkungen oder Gedanken dazu festzuhalten. Die Annotation eines Objekts darf nicht abhängig vom Recht zur Änderung des Objekts selber sein; für einen Benutzer nicht veränderbare Objekte bleiben so für ihn annotierbar. Eine Gesamtübersicht über alle jemals getätigten Annotationen und Recherchemöglichkeiten darüber müssen zusätzlich vorhanden sein.

M.2: **Bewertungen:** Neben der – sehr freien – Annotation beliebiger Informationsartefakte muß jeder Benutzer diese bewerten können, um das persönliche Gefallen bzw. die Einschät-

zung der Relevanz und der Qualität des Objekts zu dokumentieren. Um aussagekräftige Angaben zu erhalten, müsste eine Bewertung nach streng festgelegten Kriterien auf verschiedenen Skalen wie Aktualität, Relevanz, Qualität (sowohl der Präsentation als auch inhaltlich) geschehen. Eine Unterscheidung der Bewertungen nach Zielgruppen (Relevanz für welche Gruppe etc.) wäre dann sinnvoll. Dies würde umfangreiche Eingaben für jedes zu bewertende Informationsartefakt bedingen, die die Benutzbarkeit und die Akzeptanz des Bewertungssystems gravierend schmälern würden. Ein Ausweg aus dieser Problematik wäre in Anreizsystemen zu sehen, die eine hohe Zahl von Bewertungen auf bestimmte Weise honoriert. Eine pragmatische Alternative ist die Zusammenfassung der gesamten Bewertung auf eine einfache Skala und der Verzicht auf die Definition komplexer Bewertungskriterien. Dies macht die Bewertung bei einer geeigneten Benutzungsoberfläche zu einer schnell zu erledigenden Aufgabe. Die Mindestanforderung ist die Bereitstellung einer Skala mit wenigen Werten (etwa 3 bis 7 Stufen von negativ über neutral bis positiv) zur gesammelten Bewertung eines jeden Objekts. Ferner ist die Gesamtübersicht über alle jemals bewerteten Objekte nötig, wobei insbesondere die Sortierung nach der Bewertung hilfreich sein kann, um die wichtigsten Dokumente wiederzufinden.

Den Bewertungsfunktionen kommt neben dem persönlichen Aspekt der wesentlich wichtigere der durch die Auswertung der Bewertungen aller Benutzer möglichen automatischen Wissensentdeckung zu, der in Abschnitt 4.3.7 eingehend beleuchtet wird.

M.3: Feedback: Ähnlich wie Annotationen und Bewertungen muß ein Feedback-Mechanismus vorhanden sein, über den jeder Benutzer dem oder den Autoren eines Dokuments oder beliebigen anderen Informationsartefakts persönliche Kritik, Fragen oder Anregungen übermitteln kann. Dabei muß zu wählen sein, ob diese lediglich dem Autor oder der gesamten Öffentlichkeit zugänglich sein sollen. Eine Unterscheidung der Beiträge nach Art (Frage, Verbesserungsvorschlag, Anmerkung etc.) ist dabei wünschenswert.

Auf der anderen Seite bedeutet der Mechanismus, daß jeder Autor für die von ihm verfaßten Dokumente eine Liste der eingegangenen Feedback-Beiträge vorfinden muß. Die jeweiligen Beiträge müssen bei der Anzeige eines Dokuments durch den Autor erscheinen. Dies bietet den Ansatzpunkt für Diskussionen und somit für kollaborative Aspekte, die in Abschnitt 4.3.8 weiter vertieft werden.

M.4: Sammelmappen: Zur Strukturierung der persönlichen Sicht auf das Informationsangebot des Portals und zur Unterstützung von Arbeitsprozessen muß das Portal für jeden Benutzer Sammelmappen bereitstellen, in der Referenzen auf beliebige Informationsartefakte gesammelt werden können. Sammelmappen sind als persönliche Ordner zu betrachten, in denen nach thematischen Gesichtspunkten z.B. für aktuelle Projekte relevante Arbeitsmaterialien zusammengetragen werden können (vgl. dazu auch [SSSW01]). Durch die Referenzsemantik der Mappen, die lediglich Verweise auf die an der ursprünglichen Stelle weiterhin existierenden Objekte enthalten, wird die Sicht anderer Benutzer nicht beeinflußt. Sammelmappen müssen als Objekte aufgefaßt werden können, die wiederum in Sammelmappen enthalten sein dürfen. Dadurch ergibt sich eine hierarchische Strukturierungsmöglichkeit mit eigenen Zugriffspfaden auf die Informationen. Die Sammelmappen können als servergestützte Lesezeichenverzeichnisse aufgefaßt werden.

Unerläßlich für das Funktionieren der Sammelmappen ist das Vorhandensein des in Abschnitt 4.2 geforderten uniformen Modells für alle Informationsartefakte, damit diese unterschiedslos verwendet werden können. Durch die Verwendung bidirektionaler Verknüpfungen für die Referenzen der Sammelmappen kann bei der Anzeige jedes Dokuments etc. sofort kenntlich gemacht werden, ob es sich in einer Sammelmappe des aktuellen Benutzers befindet.

M.5: Konfiguration der Oberfläche: Die älteste Form der Personalisierung ist die der benutzerspezifischen Anpassung der Oberfläche an die Vorlieben und Gewohnheiten des jeweiligen Benutzers. Zu personalisierende Einstellungen umfassen die Gestaltung der Oberfläche (etwa Farben, Zeichensätze etc.) und die verwendete Sprache der Oberfläche sowie das Erscheinungsbild der angebotenen Funktionen.

Der Trend vieler Portalsysteme geht dahin, funktionale Einheiten in getrennten Elementen wie eigene Fenster oder durch simulierte Fenster zu visualisieren (vgl. Abschnitt 3.4 und etwa [Bau01]). Diese Elemente werden z.B. *Portlets* oder *Tracks* genannt. Diese Einheiten sind häufig Fenstern des Betriebssystems nachempfunden und lassen sich in Größe, Form und Position frei arrangieren. Ohne den Anforderungen des Abschnitts 4.4 vorauszugreifen, gilt zumindest, daß der Benutzer seinen individuellen Arbeitsplatz und den Zugriff auf die individuell benötigten Arbeitsmittel (Funktionen und Informationen) frei gestalten können muß.

Die Personalisierung ist einer der entscheidenden Faktoren für die Akzeptanz und Benutzbarkeit eines Portals. Die Arbeitsweise aller Funktionen basiert in hohem Maße auf grundlegenden Modellierungsannahmen wie bidirektionalen Verknüpfungen und uniformen Objekt- und Dokumentenmodellen [RSG01]. Sehr wichtig ist, daß alle Annotationen und Feedback-Meldungen als spezielle Dokumentenarten betrachtet werden. Damit eröffnen sich beispielsweise orthogonale Recherchemöglichkeiten auch für diese. Durch bidirektionale Verknüpfungen kann die Konsistenz aller Informationen gewährleistet werden.

4.3.6 Wissenssuche und Wissenszustellung

suchen wissen

ich was suchen
ich nicht wissen was suchen
ich nicht wissen wie wissen was suchen
ich suchen wie wissen was suchen

ich wissen was suchen
ich suchen wie wissen was suchen
ich wissen ich suchen wie wissen was suchen
ich was wissen
– ERNST JANDL, die bearbeitung der mütze

Eine der wichtigsten Funktionen überhaupt in einem Wissensportal ist die Suchfunktion. Zu der traditionellen aktiven Rolle des Benutzers als Suchendem kommt die passive des automatisch relevante Neuigkeiten Empfangenden, wobei die Rolle des Systems jeweils andersherum passiv bzw. aktiv ist (siehe dazu auch das Modell der *Business Conversations* in Abschnitt 5.2.1 und [WHM00]). Die aktive Suche und das passive Empfangen (Wissenszustellung) werden als *pull*- und *push*-Technik bezeichnet. Häufig als Synonym zu *push*-Technik verwendet findet man den Begriff der Informationskanäle (*channels*). Beide Arten der Informationsversorgung müssen von einem Wissensportal unterstützt werden. Im einzelnen sind folgende Funktionen wichtig:

S.1: Recherche: Die klassische, aktive Suche nach benötigten Informationen muß diverse Möglichkeiten umfassen und orthogonal über alle vorhandenen Informationsartefakte aller Verzeichnisse möglich sein:

- Suche nach Metainformationen (Autor, Erstellungsdatum etc.): Diese muß teilweise gestützt durch das Angebot möglicher Werte (etwa der registrierten Benutzer) gesche-

hen können. Sowohl die bereits vordefinierten als auch die benutzerdefinierten Metaattribute (vgl. Anforderung D.4) müssen durchsuchbar sein.

- Suche in den Volltexten der Dokumente: Hier muß nach Wörtern und Wortkombinationen gesucht werden können. Dabei ist eine Suche, die automatisch Wortstämme bildet (*stemming*) und so auch gebeugte Wortformen findet, von großem Vorteil [Fra92]. Eine gewichtete Anzeige der Treffermenge nach (relativer) Häufigkeit der gesuchten Worte ist dabei hilfreich.
- Stichwortsuche: Diese ist entweder direkt der Suche nach Metainformationen zuzuordnen, indem dort nach verknüpften Begriffen gesucht werden kann (vgl. Anforderung B.3), oder ist als integrierte Anzeige der assoziierten Objekte eines Begriffs zu realisieren. Die freie Eingabe von Stichworten zur Suche kann dennoch nützlich sein; hier kann dann sowohl nach Begriffen und ihren Synonymen als auch im Volltext gesucht werden.
- Bei speziellen Arten von Dokumenten können spezielle Suchfunktionen nützlich sein. So kann bei XML-Dokumenten die Suche nach Werten zusätzlich durch die Angabe von XML-*Tags* eingeschränkt werden, also bereits semantisches Wissen in die Anfrage eingebracht werden. Eine Anfrage könnte dann gezielt nach bestimmten Informationen in einer XML-Datei suchen und so die Nachteile des fehlenden Kontextes der Volltextsuche vermeiden; beispielsweise wäre bei entsprechenden Dokumenten die gezielte Suche nach in Titeln oder Bildunterschriften verwendeten Begriffen möglich.
- Bei Stichwort- und Volltextsuche ist die Verwendung eines Thesaurus, der Synonyme bereitstellt, hilfreich. In erster Linie muß dieser auf die Synonyme, die in den Begriffen enthalten sind, zurückgreifen. Eine wichtige Anforderung ist hier Sprachunabhängigkeit, d.h. daß auch Objekte gefunden werden müssen, die die Stichwörter in anderen unterstützten Sprache enthalten.
- Unscharfe Suche: Techniken des *Fuzzy Retrieval* können neben der klassischen booleschen Suche mit scharfen, logischen Verknüpfungen verwendet werden, um bessere Ergebnisse zu erhalten [MS99, Büc99].

S.2: Suche nach Experten: Eine für das Management von implizitem Wissen wichtige Funktion ist die Expertensuche, die es gestattet, Personen mit Kenntnissen und Fähigkeiten zu einem bestimmten Thema zu finden. Im Rahmen der hier beschriebenen Anforderungen und des Modells der Vernetzung aller Informationsartefakte können zwei Wege beschritten werden: Die direkte Anzeige aller entspechenden Personen ausgehend von einem Begriff, um so die Experten zu dem dadurch beschriebenen Thema zu finden, oder eine explizite Suchfunktion im Bereich Personen mit der (evtl. durch die Taxonomie gestützten) Eingabe des Themas, die dann durch Auswertung der Verknüpfungen und ggf. der Selbstbewertungen (vgl. Anforderung P.4) eine Liste von Personen liefert. Ein Wissensportal sollte beide Funktionen besitzen.

S.3: Exploration: Mit in den Bereich des Suchens fällt das explorative Vorgehen durch Verfolgen von Assoziationen. Hierzu bieten sich die zahlreichen Verknüpfungen u.a. durch Verzeichnisse, Klassifikation, Kategorisierung, Autorenschaft, Bewertungen und Sammelmappen an. Eine spielerische Art des Erkundens dieser eng vernetzten Inhalte kann häufig zu neuen und überraschenden Erkenntnissen führen. Ein derart hypermedial orientierter Zugang zu allen Informationen ist daher eine wesentliche Anforderung. Gerade das Erforschen von Begriffstaxonomien ist ein zentraler Zugang zum Kennenlernen eines umfangreichen Wissensgebiets (ähnlich wie das „Surfen“ im Internet).

S.4: Personalisierung: In der Benutzungsoberfläche eines Wissensportals wird es typischerweise mehrere Stellen geben, an denen Suchfunktionen vorhanden sind, um spezifische

Anfragen über bestimmte Arten von Informationen abzusetzen, etwa Expertensuche oder Dokumentensuche. In vielen Fällen wird neben einer einfachen Suchmöglichkeit eine komplexere Abfrage sinnvoll sein, die die Eingabe einer größeren Zahl von verschiedenen Suchparametern gestattet. Mit zum Aspekt der Personalisierung gehören die folgenden Eigenschaften, die die Benutzung der Recherchefunktionen für den Benutzer komfortabler gestalten:

- Die letzten verwendeten Einstellungen (Parameter und Optionen) jeder Suchfunktion müssen für jeden Benutzer getrennt festgehalten und bei der nächsten Verwendung wieder vorgeschlagen werden.
- Die Möglichkeit, mehrere Sätze von Suchparametern zur erneuten Verwendung unter einem wählbaren Namen abzulegen und wieder aufzurufen, muß vorhanden sein. Dies erleichtert die routinemäßige Verwendung von öfter benötigten Anfragen erheblich.

Die Präsentation der Anfrageergebnisse und aller Listenausgaben insgesamt muß nach verschiedenen Kriterien zu ordnen sein, die sowohl beim Stellen der Anfrage als auch später noch angebbar sein müssen. Das jeweils letzte Sortierkriterium muß für alle Ergebnislisten vom System pro Benutzer festgehalten werden und bei der nächsten Verwendung automatisch benutzt werden.

S.5: **Persistente Suche:** Eine eng mit der Speicherung ganzer Sätze von Suchkriterien verbundene Funktion ist die der persistenten Suche. Einmal abgespeichert, kann ein benannter Satz an Kriterien als dynamische Kollektion von Informationsobjekten angesehen werden, deren Inhalt bei jedem Öffnen der Kollektion ermittelt wird, indem die Anfrage erneut ausgewertet wird.

Ein solcher Satz von Suchkriterien muß in die Verzeichnishierarchie oder in die persönlichen Verzeichnisse und Sammelmappen eines Benutzers derart eingefügt werden können, daß er wie ein gewöhnliches Verzeichnis bzw. eine Sammelmappe erscheint. Der Inhalt aber ergibt sich alleine aus der dahinterliegenden Anfrage. Der Inhalt eines solchen virtuellen Verzeichnisses kann natürlich nicht durch die sonst üblichen Operationen verändert werden, lediglich die Anfrage selbst muß später noch verändert werden können.

S.6: **Notifikation:** Eine auf der persistenten Suche aufbauende Funktion, die in den Bereich der (passiven) Informationszustellung fällt, ist die Notifikation (Benachrichtigung) über Veränderungen in den Suchergebnissen. Für jeden einzelnen der oben beschriebenen Sätze von persistenten Suchkriterien muß einstellbar sein, ob das Portalsystem – seinerseits aktiv – von sich aus auf Veränderungen in der Ergebnismenge insbesondere durch neu hinzugekommene oder einzelne geänderte Informationsartefakte reagieren soll. Die Zustellung der neuen Informationen kann dann durch verschiedene – asynchrone oder synchrone – Benachrichtigungsmechanismen geschehen:

- Per e-Mail (oder per SMS) an den Benutzer mit der Information, daß und wo genau eine Veränderung stattgefunden hat. Die e-Mail-Adresse muß dazu den Kontaktinformationen des persönlichen Profils (vgl. Anforderung P.3) zu entnehmen sein.
- Durch Auszeichnung der neu hinzugekommenen Objekte in den (virtuellen) Verzeichnissen und Sammelmappen durch bestimmte Merkmale.
- Durch Platzierung der neuen oder geänderten Objekte in zusätzlichen, dafür speziell bekanntgegebenen Sammelmappen.

Natürlich können auch Kombinationen aller angegebenen Mechanismen sinnvoll sein. Als Metapher für solche *push*-Dienste ist der Begriff des Abonnements gut geeignet, der etwa bei Mailinglisten bereits recht verbreitet ist.

Den Zustellungs- und Suchfunktionen kommt in einem Wissensportal eine zentrale Bedeutung zu. Gerade auf die passiven Funktionen zur Wissenszustellung ist – neben den immer noch genauso erforderlichen aktiven Suchfunktionen – bei der Verwaltung umfangreicher Informationsbestände nicht mehr zu verzichten. Eine nahtlose Integration der hier sogenannten virtuellen Verzeichnisse und Sammelmappen in die persönliche Sicht jedes Benutzers stellt dabei einen wichtigen Mehrwert und eine Schlüsselfunktion zur Wissensverteilung (vgl. Abschnitt 2.4.5) dar.

4.3.7 Wissensentdeckung

Zwar weiß ich viel, doch möcht' ich alles wissen.
– JOHANN WOLFGANG VON GOETHE (Faust I, Nacht, Wagner zu Faust)

Die Aufdeckung von Beziehungen zwischen Informationen, also die Wissensentdeckung (*Knowledge Discovery*) und die automatische Klassifikation von Informationen ist eine der fortgeschrittensten Techniken des Wissensmanagements. Von einem modernen Wissensportal sind solche Funktionen zu fordern, die dann die manuelle Klassifikation unterstützen bzw. ergänzen. Viele dieser Funktionen sind eng verwandt mit Recherchefunktionen und Volltextextraktion. Die Grenzziehung zwischen Klassifikationsfunktionen und Suchfunktion ist nicht immer eindeutig. Gemeinsam verwenden Klassifikations- und Erschließungsalgorithmen aus Dateien extrahierte Volltexte. Im einzelnen lassen sich folgende Anforderungen aufstellen, die alle als unterschiedlich ausgeprägte inhaltliche Erschließungsaufgaben betrachtet werden können [Büc01]:

E.1: Klassifikation: Als Klassifikation bzw. Textklassifikation wird die Einordnung eines Dokuments in eine vorgegebene Reihe von semantischen Kategorien bezeichnet. In dem hier verwendeten Kontext bedeutet dies, daß Informationsartefakte Begriffen zugeordnet werden müssen. Dies muß durch das inhaltliche Erschließen der Dokumente (im Gegensatz zum formalen Erschließen anhand von Metainformationen) geschehen. Die inhaltliche Erschließung durch Rechner und Algorithmen – im Gegensatz zu der durch Menschen – basiert auf der Merkmalsextraktion der Volltexte und Metainformationen der zu klassifizierenden Objekte. Ein Wissensportal muß sich das an folgenden Stellen zunutze machen:

- Beim Einstellen eines neuen oder geänderten Dokuments in die Verzeichnishierarchie des Portals kann die automatische Klassifikation genutzt werden, um es ohne weitere Eingriffe und Eingaben des Benutzers relevanten Begriffen zuzuordnen.
- Ähnlich der völlig automatischen Zuordnung kann das Ergebnis der Klassifikation dazu genutzt werden, dem Benutzer beim Einstellen eines Dokumentes einen qualifizierten Vorschlag von zuzuordnenden Begriffen zu machen, den er dann entsprechend verändern kann.
- Die nachträgliche, automatische Klassifikation umfangreicher, bestehender Bestände von Dokumenten, die zu einem früheren Zeitpunkt unerschlossen – etwa als Massenimport oder neue Quelle – in das Portalsystem integriert wurden.

Da das manuelle Klassifizieren von Dokumenten eine zeitaufwendige Arbeit ist, die viel Sorgfalt erfordert, muß die automatische Klassifikation zur Entlastung der Benutzer verwendet werden.

E.2: Kategorienbildung: Eine wesentlich umfangreichere Erschließungsaufgabe als die Klassifikation einzelner Dokumente ist die automatische Generierung von ganzen Begriffssystemen (Taxonomien). Diese Funktion bietet beispielsweise der LOTUS DISCOVERY SERVER an (vgl. Abschnitt 3.4.3). Die mindestens erforderliche Funktionalität ist, daß ein Kategorien

bildendes System die vorhandenen Dokumente analysieren kann, um daraus eine Menge von Kategorien zu bilden (häufig *Cluster* genannt), die Mengen von ähnlichen Dokumenten beschreiben. Zugleich müssen die Dokumente natürlich diesen Kategorien zugeordnet werden.

Diese Funktion kann zur initialen Erzeugung eines Begriffssystems benutzt werden, wenn bereits größere Mengen an Dokumenten vorliegen, aber noch keine Unternehmensterminologie existiert. Die manuelle Pflege und Verbesserung der so entstandenen Begriffe bleibt aber zumeist unerläßlich. Die Technik der automatischen Kategorienbildung kann grundsätzlich nutzbringend auch für andere Klassifikationsaspekte verwendet werden; die im folgenden beschriebene assoziative Suche basiert darauf.

E.3: Assoziative Suche: Die assoziative Suche ist eine Funktion, die es ermöglicht, nicht explizit angegebene oder nicht offensichtlich vorhandene Beziehungen zwischen Informationsartefakten zu entdecken. Sie findet zu einem gegebene Objekt oder sogar zu einem ad hoc eingegebene Freitext Objekte mit ähnlichem semantischen Inhalt. Dadurch wird die Navigation entlang ähnlicher Dokumente möglich, und so oft die Entdeckung neuer, bislang unbekannter und verborgener Zusammenhänge gefördert. Die assoziative Suche muß über verschiedene Arten von Informationsartefakten hinweg möglich sein.

An folgenden Stellen eines Wissensportals muß diese Funktionalität ausgenutzt werden:

- Bei der Anzeige eines Dokuments muß die Anzeige einer Liste von semantisch ähnlichen Dokumenten möglich sein. Diese Liste sollte dann nach dem Grad der Ähnlichkeit geordnet sein. Diese kann entweder direkt durch Vergleich der Merkmale der Dokumente oder indirekt über die Begriffe geschehen. Auch gemischte Bewertungen sind denkbar.
- Bei der Anzeige eines Dokuments muß gleichfalls eine Liste von assoziierten, semantisch zugehörigen Begriffen angezeigt werden können. Das sind normalerweise dann diejenigen Begriffe, die die automatische Kategorisierung ermittelt.
- Bei der Anzeige eines Begriffs müssen dazu passende, semantisch ähnliche Dokumente (wiederum sortiert nach Ähnlichkeit) angeboten werden, etwa unter dem Hinweis „siehe auch“. Dies entspricht der umgekehrten Beziehung von Dokumenten zu Begriffen.
- Bei einem Begriff muß eine Liste von Personen angezeigt werden können, die besondere Kenntnisse darüber haben. Diese Information kann indirekt durch Analyse der von Personen stammenden ähnlichen Dokumente gewonnen werden. Diese Assoziation kann zur Realisierung der zuvor geforderten Expertensuche verwendet werden (vgl. Anforderung S.2).

E.4: Empfehlungen: Eine weitere Möglichkeit ergibt sich aus der assoziativen Suche: Bei der Anzeige eines beliebigen Informationsartefaktes kann aus der semantischen Ähnlichkeit zu allen von dem Benutzer selbst verfassten Informationsartefakten der gleichen Art die mutmaßliche Relevanz des angezeigten Objekts für ihn berechnet und angezeigt werden. Eine weitergehende Berechnung von für einen Benutzer relevanten Objekten kann sich die von dem Benutzer selbst abgegebenen Bewertungen, Annotationen und Rückmeldungen (vgl. Abschnitt 4.3.5) zunutze machen, daraus persönliche Interessen ableiten und dadurch relevante Objekte finden. Dies wird als *Relevance Feedback* oder *Recommendation*, also Empfehlung, bezeichnet [Die01].

Zur Auswertung, welche Informationsartefakte für einen Benutzer relevant sind, ist eine Vielzahl von Strategien möglich: Neben expliziten Hinweisen wie Autorenschaft und vom Benutzer selbst vorgenommenen Bewertungen von Dokumenten und eigenen Kenntnissen

spielen implizite Hinweise wie Nutzungsfrequenz einzelner Dokumente, getätigte Annotationen und abgesetzte Suchanfragen eine mindestens ebenso wichtige Rolle. Diese Hinweise müssen alle gemeinsam durch eine Metrik verarbeitet werden, für die sich eine gute Einstellung nur durch empirische Beobachtungen gewinnen läßt.

Alle diese Funktionen zur Klassifikation müssen sich in hohem Maße der Verknüpfungsmöglichkeiten zwischen verschiedensten Informationsartefakten bedienen. Dies macht erneut die Bedeutung einer uniformen Behandlung der Objekte und zudem die Wichtigkeit der bidirektionalen Verknüpfungen deutlich.

4.3.8 Gemeinschaften

I wouldn't want to belong to any club that would accept me as a member.
– GROUCHO MARX

Zur Unterstützung der Gruppenarbeit muß ein Wissensportal eine Reihe von Eigenschaften und Funktionen aufweisen, die zum Teil in den Bereich der sogenannten *Awareness* fallen. Diese ist dabei definiert als „*understanding of the activities of others, which provides context for your own activity*“ [DB92]. Für die Gruppenunterstützung ergeben sich somit folgende Anforderungen an ein Wissensportal:

G.1: Ankündigungen: Es muß eine zentrale Stelle geben, an der Ankündigen allgemeiner Art gepflegt werden können (Schwarzes Brett). Den dort publizierbaren Artikeln müssen bei Bedarf Bilder oder andere Anlagen angefügt werden können. Falls die Notwendigkeit besteht, müssen weitere, themenspezifischere Bretter verwaltet werden können. Für die explizit als Ankündigungen etc. betrachteten Artikel darin muß ein geeignetes Rechtekonzept vorhanden sein, so daß evtl. nur ein eingeschränkter Personenkreis (z.B. Sekretariate) zum Publizieren berechtigt sind. Ein Betreff, Langtext, Autor und ein Verfallsdatum sind die mindestens notwendigen Attribute für die Artikel.

G.2: Diskussionsforen: Zu allen relevanten Themen müssen spezielle Diskussionsforen eröffnet werden können („*Knowledge Communities*“). Zu unterscheiden sind dabei offene Foren, die allen Benutzern zur Verfügung stehen, und geschlossene, die nur dazu berechtigten – also für die darin stattfindenden Diskussionen zugelassenen Benutzern – dienen. Durch Verwendung des in Anforderung P.2 postulierten Gruppen- und Rollenmodells lassen sich leicht die zum Initiieren und zur Teilnahme an Diskussionsforen berechtigten Personenkreise definieren. Die Möglichkeit, moderierte Foren einzurichten, bei denen jeder Beitrag vor der Veröffentlichung von einem Moderator geprüft wird, muß vorhanden sein. Zur Unterstützung der Moderationsprozesse sind die in Abschnitt 4.3.9 genannten Funktionen zu verwenden.

Die Beiträge selbst müssen sich ggf. mit Anhängen versehen lassen. Eine Anzeigemöglichkeit für jeden Benutzer, die alle von ihm verfassten Beiträge aller Foren anbietet und umfangreiche Suchfunktionen bereitstellt, ist ebenfalls nötig. Werden Beiträge als spezielle Arten von Dokumenten aufgefaßt, lassen sich durch die orthogonale Anwendbarkeit aller für Dokumente vorhandenen Recherche- und Klassifikationsfunktionen beträchtliche Mehrwerte für Wissensarbeiter schaffen. Als Beispiel sei nur die automatische Benachrichtigung (Notifikation) über neu entstandenen Diskussionen zu einem dem Benutzer wichtigen Thema in einem Forum, das er sonst nicht beachtet, zu nennen.

G.3: Feedback als Diskussion: Anknüpfend an die in Anforderung M.3 genannte Möglichkeit, Rückmeldungen zu jedem Dokument zu geben, muß die zusätzliche Möglichkeit geschaffen werden, jedes Dokument als Ausgangspunkt einer eigenen Diskussion zu sehen, also praktisch als eigenes Forum. In einem solchen Forum können dann Diskussionen zwischen Autor und Leser über den Inhalt ablaufen. Hierfür ist die Nutzung von Workflow-Funktionalität nötig.

G.4: Wissensgemeinschaften: Der Begriff der Wissensgemeinschaft (*Knowledge Community* oder Praxisgemeinschaft, *Communities of practice*) als eher informelle, nicht in der statischen Aufbaustruktur einer Organisation verankerte Gruppe von Personen mit ähnlichen Interessen [NRP00, Sch00a, Bor00] läßt sich gut durch das in Anforderung P.2 genannte Gruppenkonzept unterstützen. Darauf aufbauend sind für jede Gruppe spezielle unterstützende Funktionen zu fordern: Dazu gehören gruppenspezifische Nachrichtenverteiler etwa per e-Mail und eigens für die Gruppe bestehende Foren und Arbeitsräume (*Workspaces*) in Form von gemeinsam genutzten Verzeichnissen und Sammelmappen. Alle diese Funktionen sind bereits in den bisherigen Anforderungen aufgeführt, ein zentraler Zugang zu allen für die Gruppe relevanten Diensten ist aber als Portalfunktion im Kleinen notwendig. Ansonsten muß der organisatorische Prozeß außerhalb der informationstechnischen Facette nach [DDJ+98] das übrige zur Etablierung einer Gruppe leisten.

G.5: Direkte Kontakte: Zur Unterstützung der Zusammenarbeit in der Gruppe muß das Portal anzeigen können, wieviele und welche Benutzer gerade angemeldet sind und mit dem System arbeiten. Das direkte Senden einer Nachricht an einen bestimmten dieser Benutzer (e-Mail, SMS, Schnellnachrichten z.B. durch *Instant Messages* oder eigene, vom Portal bereitgestellte Dienste) muß möglich sein.

Auf diesen Funktionen aufbauend kann die Vermittlung einer Online-Konferenz (*Chat*) angeboten werden; entweder durch einen im Portal integrierten Dienst oder durch die Nutzung entsprechender Software von weiteren Herstellern. Dazu müssen ggf. entsprechende Kontaktinformationen im Profil jedes Benutzers vorhanden sein.

G.6: Integration: Um den Personen und Gruppen den Umgang mit den gewohnten Werkzeugen zu erleichtern und um einen bruchlosen Übergang zur Benutzung des Wissensportals zu schaffen, muß die bestehende Infrastruktur integriert werden können. Insbesondere zählen dazu:

- Integration externer e-Mail-Systeme. Zumindest der lesende Zugriff auf die Postfächer des aktuellen Benutzers sollte dem System unter Benutzung der Zugangsinformationen aus dem Personenprofil möglich sein. So ließe sich im Sinne einer integrierten Darstellung der für den Benutzer relevanten Informationen z.B. dessen aktueller Posteingang auf seiner persönlichen Startseite anzeigen. Die konsequente Erweiterung wäre, daß e-Mail direkt über das Portalsystem empfangen, verwaltet und geschrieben werden kann.

  Die e-Mail-Funktionalität muß folgendermaßen auf die bereits geschilderten Konzepte abgebildet werden: die verschiedenen Postkörbe auf Verzeichnisse, die innerhalb des persönlichen Verzeichnisses des Benutzers (vgl. Anforderung V.8) liegen, die einzelnen Nachrichten als Dokumente darin und Adressbücher als Gruppen und Personen.

- Integration externer Diskussionsforen. Da die *Newsgroups* des Internet über standardisierte Protokolle (NNTP, [KL86]) ausgetauscht werden, ist die Integration einfach zu bewerkstelligen. Wiederum ließen sich bestimmte Foren als allgemeine Verzeichnisse und die einzelnen Nachrichten als Dokumente darin abbilden.

- Integration weiterer (externer) Systeme wie Kalender. Gerade für Projektgruppen sind Werkzeuge für die Koordination ihrer Aktivitäten wichtig. Zumindest ein Gruppenkalender muß angeboten werden, der spezifisch für jede Gruppe zu pflegen ist.

Die Integration insbesondere von e-Mail und *News* in das Verzeichnis- und Dokumentenkonzept eröffnet weitreichende Vorteile, da diese dann von den orthogonal verfügbaren Diensten der Recherche und der Klassifikation genutzt werden können. Die Benutzung der Notifikationsdienste ist in diesem Zusammenhang äußerst wichtig.

Für die geforderten Gemeinschaftsfunktionen gilt wiederum, daß die uniforme Modellierung von Informationsartefakten und die Abbildung auf Verzeichnisse und Dokumente das Schlüsselkonzept darstellen, da Nachrichten etc. erst so erschlossen und nutzbar gemacht werden können.

4.3.9 Prozesse und Workflow

Die bisher beschriebenen Anforderungen decken im wesentlichen die produktorientierten Eigenschaften von Informationssystemen (vgl. Abschnitt 3.2.1) ab. Ein Wissensportal muß aber auch prozeßorientierte Eigenschaften zur Unterstützung von Workflows besitzen. Weitere Fähigkeiten könnten durch Anbindung existierender WFMS erreicht werden.

Da ein Wissensportal nicht unbedingt die volle Funktionalität eines ausgereiften Workflow-Management-Systems besitzen kann und muß, ist eine sinnvolle Reduzierung des Funktionsumfangs auf im Kontext des Wissensmanagements relevante Kernaufgaben nötig, also auf die in den vorangegangenen Abschnitten aufgeführten Anforderungen. Folgende Funktionen sind zu fordern:

W.1: **Administration:** Ein typischer zu unterstützender Prozess ist der Administrationsprozeß, der aus der Beantragung neuer Zugangsberechtigungen durch bislang dem System nicht bekannte Benutzer, der Prüfung des Antrages durch dafür bestimmte Personen und der anschließenden Erzeugung und Freigabe einer neuen Berechtigung im Falle eines positiven Bescheids nebst Benachrichtigung des Antragstellers besteht [Leh01].

W.2: **Publikationsprozesse:** Der Kernfunktionalität von heutigen Dokumenten-Management- und Content-Management-Systemen zuzurechnen sind Publikationsprozesse wie das kooperative Erarbeiten eines Dokuments mit anschließendem Begutachtungsprozeß durch einen oder mehrere Gutachter und schließlich der öffentlichen Freigabe des Erzeugnisses. Die produktorientierte Infrastruktur dafür steht bereits in Form von Personen mit Rollen und Rechten, Verzeichnissen und Dokumenten bereit; auch der Austausch von Gedanken über ein dem Dokument zugeordnetes Diskussionsforum ist möglich (vgl. Anforderung G.3). Zusätzlich dazu muß die gezielte Initiierung und Steuerung solcher Prozesse angeboten werden.

W.3: **Aufgaben:** Eine häufig nützliche Funktion zur Organisation der eigenen Arbeit sind Listen mit zu erledigenden Aufgaben. Eine gruppenweite Nutzung mit Aufgaben, die man nicht nur sich selbst, sondern auch anderen zuweisen kann, ist zudem für Gemeinschaftsarbeit hilfreich. Eine Aufgabenbeschreibung muß dafür zumindest die folgenden Attribute besitzen:

- Einen Betreff und eine ausführliche Beschreibung. Es muß ferner möglich sein, beliebige Anlagen (Bilder, Dokumente) anzufügen.

- Das Datum der Erstellung und das Fälligkeitsdatum; die Person, der die Bearbeitung der Aufgabe obliegt und die Person, die die Aufgabe vergeben hat.
- Den Status (offen, in Bearbeitung, erledigt etc.) und Priorität der Aufgabe.
- Den Aufwand (evtl. geschätzt) und Schwierigkeitsgrad der Aufgabe.
- Eine Liste von Teilschritten und (möglicherweise unterschiedlichen) Bearbeitern zu ihrer Durchführung.

Zur effektiven Nutzung der Aufgaben müssen u.a. Notifikationen über neu eingegangene, zugewiesene, weitergeleitete und erledigte Aufgaben verwendet werden. Zudem sind weitere Workflow-Funktionen zur Statusverfolgung und Erinnerung ausgeschriebener Aufgaben nötig. Weiterhin muß die Verfolgung der Historie der Aufgabe, also wer wann welche Teilschritte erledigt hat, sowie die Definition von (nebenläufig auszuführenden) Unteraufgaben möglich sein. Dies fällt mit in das Gebiet der *Awareness* [DB92].

W.4: Ad-hoc-Prozesse: Ähnlich wie Aufgaben sind ad-hoc definierte Prozesse ein wichtiges Mittel zur Arbeitsorganisation. Diese Prozesse bzw. ihre Beschreibungen müssen ähnlich wie die Aufgaben aufgebaut sein, wobei allerdings die Prozeßausführung (Initiierung, Überwachung etc.) von wesentlich größerer Bedeutung ist.

Alle diese Prozesse haben viele Gemeinsamkeiten, so daß eine generische Lösung zur Abwicklung sinnvoll ist [Leh01]. Die dazu modellierten Informationsartefakte (Aufgaben, Schritte, Anlagen) müssen wiederum Gebrauch von bidirektionalen Verknüpfungen machen und selbst als Dokumente, Sammelmappen oder Verzeichnisse erscheinen, um die Mehrwertdienste des Portals gewinnbringend dafür nutzen zu können.

4.4 Anforderungen an die Benutzungsoberfläche

Wer das Richtige sagt, braucht dabei nicht gut auszusehen.
– STEN NADOLNY

Die nicht funktionalen Anforderungen an die Benutzungsoberfläche eines Wissensportals fallen nicht so umfangreich wie die funktionalen aus und werden deshalb nicht in weitere Unterabschnitte gegliedert. Diese Anforderungen sind im wesentlichen durch Fragen des Anwendungskontextes und der Benutzbarkeit (also der Software-Ergonomie) bestimmt [EOO94]. Es fließen aber auch bereits stellenweise Systemanforderungen ein bzw. solche, die aus Gesichtspunkten der Entwicklung, Wartung und Anpassung sowie des Betriebes resultieren.

O.1: WWW-Zugang: Die Basisannahme für die Anforderungen an die Oberfläche ist die der Verwendung eines WWW-basierten Zugangs über Internet, Extranet oder Intranet. Das Portalsystem muß also die Oberfläche in Form von HTML unter eventueller Nutzung von weiteren Standards wie CSS und JAVASCRIPT ausliefern. Dabei müssen die gängigen Web-Browser auf den gängigen Betriebssystemen mit ihren Spezifika unterstützt werden.

O.2: Geräteunabhängigkeit: Grundsätzlich muß eine strikte Trennung zwischen Funktionalität und Gestaltung der Oberfläche erreicht werden. Neben der Basisanforderung der Unterstützung eines HTML-basierten Zugangs muß das Portalsystem grundsätzlich Geräteunabhängigkeit bieten, so daß es leicht für die Benutzungsoberflächen anderer Endgeräte anpaßbar ist. Neben HMTL etc. für Web-Browser kommt beispielsweise WAP bzw. WML (*Wireless Application Protocol* bzw. *Wiresless Markup Language*) für Mobiltelefone und

PDAs (persönliche digitale Assistenten) in Frage [WML00]. Wie in [Bar01] ausführlich beschrieben, sind dabei sogar umfangreiche Unterschiede zwischen einzelnen Gerätetypen vorhanden, die entsprechend berücksichtigt werden müssen. Die Unterscheidung und Ausnutzung der verschiedenartigen Fähigkeiten und Besonderheiten der diversen Web-Browser kann erforderlich sein und muß unterstützt werden.

Die Trennung von Funktionalität und Gestaltung muß die Abhängigkeiten zwischen eigentlicher Anwendungslogik des Portalsystems und Ausgabeformat soweit minimieren, daß die Anwendungslogik unterschiedslos für verschiedene Endgeräte verwendet werden kann.

O.3: Mehrsprachigkeit: Die Benutzungsoberfläche des Portals muß in unterschiedlichen Sprachen angeboten werden können. Dabei muß jeder Benutzer die ihm genehmste Sprache frei wählen können. Dies betrifft hier nicht die Inhalte und die Dokumentenmodelle; die aktuellen Dokumente etc. werden natürlich weiterhin typischerweise nur in einer Sprache vorliegen. Die (normalerweise) sprachunabhängigen Inhalte, also die aus den konzeptuellen Informationsartefakten generierten Informationen, werden gemäß der Trennung von Layout und Inhalt in Vorlagen eingesetzt, die ihrerseits sprachabhängig sind und gemäß der aktuellen Spracheinstellung des Benutzers aus einem Satz von Vorlagen der angebotenen Sprachen gewählt werden. Das Umschalten der Sprache muß jederzeit möglich sein. Die bevorzugte Sprache jedes Benutzers muß in seinem Profil vermerkt werden.

Die Mehrsprachigkeit bedeutet, daß das System gegebenenfalls in der Lage sein muß, Informationen in unüblichen Zeichensätzen (z.B. nah- oder fernöstlichen) korrekt zu verarbeiten. Die Unterstützung des UNICODE-Zeichensatzes [The00] ist dafür eine gute Grundlage.

O.4: Anpaßbarkeit: Die einfache Anpaßbarkeit des Oberfläche des Systems an die Bedürfnisse und die Grundsätze der Gestaltung in einer Organisation (*Coporate Design*) muß gewährleistet sein. Das Mittel dazu muß wiederum die Trennung von Funktionalität und Layout sein, die die Trennung der Aufgaben von Programmentwicklern und Web-Designern ermöglicht. Dadurch, daß die Anwendungslogik von der Gestaltung der Oberfläche entkoppelt wird, können Änderungen des Erscheinungsbilds ohne großen Aufwand und ohne Eingriffe in die Anwendungslogik auch nach der Installation des Systems von Personen ohne Programmierkenntnisse vorgenommen werden. Dies ist um so wichtiger, als daß es die hier relevanten vorherrschenden Spezialisierungsrichtungen in Entwickler oder Web-Designer (die jeweils oft die andere Richtung nicht besonders gut berücksichtigen) reflektiert.

O.5: Hilfe: Eine wichtige Anforderung in einem interaktiven System ist die Bereitstellung einer *online* verfügbaren Hilfe. Diese muß für jedes Gerät und in jeder Sprache vorhanden sein. Hier stellt sich die Frage, ob die angebotenen Hilfsinformationen Bestandteil der Oberfläche (der Vorlagen) oder als Informationsartefakte Teil des (redaktionell) gepflegten Inhalts des Wissensportals selbst sein sollten. Der zweite Weg hat den Vorteil, auch Suchfunktionen, Personalisierung etc. für die Hilfethemen verwenden zu können, besitzt aber den Nachteil, die Trennung zwischen Inhalten und Oberfläche partiell zu durchbrechen.

O.6: Integration: Neben den Oberflächen, die eventuell für verschiedene Endgeräte und (orthogonal dazu) in unterschiedlichen Sprachen angeboten werden, muß es weitere in andere Systeme integrierte Zugänge zu dem Portal geben, um die individuell verschiedenen Arbeitsweisen der Benutzer möglichst gut zu unterstützen. Dazu kommen eine Reihe von Möglichkeiten in Betracht:

- Das Angebot der Verzeichnisse und darin enthaltener Dokumente als virtuelles Netzlaufwerk, das dann wie andere (entfernte) Dateisysteme betrachtet werden kann. Dateien können so direkt in das Portalsystem importiert und exportiert werden. Dies

erleichtert die Bearbeitung von Dateien erheblich, da die zusätzlichen Schritte des Herunterladens und Heraufladens der Dateien über eine web-basierte Oberfläche so entfallen. Problematisch bleibt dabei die Erfassung zusätzlicher Metaattribute. Sowohl das HYPERWAVE-System als auch der MS SHAREPOINT SERVER bieten diese Eigenschaft an. Einen ähnlichen Ansatz verfolgt das ORACLE INTERNET FILESYSTEM (iFS), das ebenfalls Datenbankinhalte sowohl web-basiert als auch als Netzlaufwerk zur Verfügung stellt [Ora01].

- Module oder Erweiterungen für gängige Büroanwendungen, die es ermöglichen, die bearbeiteten Dokumente direkt und ohne Umweg über die web-basierte Oberfläche in das Portalsystem publizieren zu können. Dabei müssen die benötigten Metaattribute abgefragt werden können.
- Die Unterstützung offener Standards und Schnittstellen, die die Ansprache von Dokumenten-Management-Systemen etc. direkt aus Anwendungen zum Ziel haben. Hier sind z.B. WEBDAV und ODMA zu nennen [GWF+99, ODM97].

Die Anforderungen der Geräte- und Sprachunabhängigkeit werden großen Einfluß auf den Entwurf haben. Außerdem beeinflußt eine gut gestaltete Oberfläche entscheidend die Akzeptanz und somit den Erfolg eines Wissensportals. Viele Hinweise dafür finden sich etwa in [Bau01, Wal01c, BLW01]. Gestaltungsgrundsätze der Softwareergonomie für web-basierte Anwendungen, speziell für Wissensportale, werden aber im Rahmen dieser Arbeit nicht weiter erarbeitet oder vertieft.

4.5 Systemanforderungen

Man kann nie so kompliziert denken, wie es letztendlich kommt.
– WILLY BRANDT

Die Bedürfnisse von Wissensmanagementsystemen in der Praxis enden nicht bei den fachlichen Anforderungen, sondern werfen vielmehr meist im Rahmen einer konkreten Projektdurchführung genauso vielfältige sekundäre qualitative und quantitative Betriebs- und Integrationsprobleme auf, die aus der praktischen Einbettung in die bestehende informationstechnische Infrastruktur resultieren. Diese hier grob mit dem Terminus „Systemmanagement“ umschriebene Gruppe von daraus entstehenden Anforderungen läßt sich in eine Reihe von Aspekten untergliedern.

Y.1: Einsatzszenarien: Der Betrieb eines Wissensportals ist in vielen verschiedenen Einsatzszenarien denkbar, die von dem Einsatz in sehr kleinen Organisationen mit wenigen Benutzern an einem Ort bis hin zu weltweit auf zahlreiche Städte verteilten Konzernen mit Tausenden von Benutzern reichen. Qualitative Anforderungen, die aus der möglichen Komplexität solcher Szenarien entstehen, sind:

- Das Portalsystem muß rasch für verschiedene Projekte umkonfiguriert werden können, um auf der Basis bestehender Lösungen unter der Weiterbenutzung bereits existierender Teillösungen neue, inkrementelle Verbesserungen für neue fachliche Anforderungen realisieren zu können. Die unter dem Begriff des *Customizing* firmierende Anpassungsfähigkeit des zunächst generischen, unternehmensneutralen Systems [Wit00] muß insbesondere sicherstellen, daß projektspezifische Anpassungen, eventuelle Weiterentwicklungen und Anpassungen in neuen Versionen des Basis-Systems selbst funktionsfähig bleiben.

- Die Mandantenfähigkeit eines Systems als die Fähigkeit, die logischen Systeme mehrere verschiedener Organisationen unabhängig voneinander auf demselben physikalischen System zu betreiben, ist eine Anforderung, die zum einen der Komplexität großer Einführungsprojekte Rechnung trägt, indem die Implementierung neuer Einstellungen, Funktionalitäten und Versionen in geschützten Testumgebungen (speziellen Mandanten) ermöglicht wird, und zum anderen das Szenario des Rechenzentrumsbetriebs, wo ein Dienstleister für verschiedene Kunden Systeme betreibt, abdeckt. Als Rechenzentrumsbetrieb kann das Szenario aufgefaßt werden, in dem von einer Organisation mehrere voneinander unabhängige Projekte bzw. Systeme betrieben werden. Ein Portalsystem muß also auch diesen Anforderungen gerecht werden, um sehr große Projekte handhaben zu können.
- Die Modularität des Portalsystems muß so ausgelegt sein, daß eine schrittweise Einführung bei sehr großen Projekten mit vielen Benutzern und umfangreichen Informationsbeständen möglich ist, ohne den Produktivbetrieb nennenswert zu beeinträchtigen. Die Migration alter Systeme muß gegebenenfalls im laufenden Betrieb stattfinden können.

Weitere Anforderungen, die mit den oben genannten eng zusammenhängen, betreffen die Skalierbarkeit des Systems und werden separat aufgeführt.

Y.2: Skalierbarkeit: Die eben genannte Reichhaltigkeit an Einsatzszenarien erfordert die Anpaßbarkeit des Portalsystems an unterschiedlich dimensionierte quantitative Anforderungen. Diese als Skalierbarkeit bezeichnete Eigenschaft eines Systems betrifft die folgenden Aspekte:

- Lastverteilung: Immer wenn aufgrund des Umfangs des Informationsbestandes oder der Anzahl der Benutzer und der daraus resultierenden Aktivität auf dem Server ein einzelner Rechner an Kapazitätsgrenzen (Zeit, Platz) stößt, muß die erzeugte Last verteilt werden. Dazu müssen entweder einzelne Komponenten und damit Aufgaben des Portalservers auf verschiedene Rechner verteilt werden können, oder aber mehrere Instanzen wichtiger Komponenten parallel auf verschiedenen Rechnern ablaufen können, die sich dann die Last der gleichartigen Aufgabe teilen (*Clustering*). In der Regel sind beide Lösungen abhängig von den Eigenarten des Systems in sinnvoller Weise kombinierbar. Dabei spielt das Lastprofil, also die Art der Benutzung des Systems, eine wichtige Rolle.
- Räumliche Verteilung: Im Gegensatz zur Lastverteilung, die die verteilten Komponenten eng beieinander aufstellt (*Clustering*), kann es aus organisatorischen Gründen sinnvoll sein, redundante Komponenten weit voneinander entfernt zu positionieren, etwa um international verteilten Organisationen an ihren jeweiligen Standorten schnellen Zugriff auf jeweils lokale Systeme zu bieten. Dies macht bei der verteilten Haltung eines einzigen logisch zusammengehörigen Datenbestandes die Replikation der auf allen Seiten vorgenommenen Änderungen nötig. Im Falle unterschiedlicher logischer Datenbestände, die aber Zugriff aufeinander haben (etwa durch serverübergreifende Verknüpfungen) wird eine möglichst transparente Kommunikation zwischen den Systemen erforderlich.
- Kapazität: Die Größe der Organisation (Mitarbeiteranzahl) und der Informationsbestände bedingen Anforderungen an den Speicherplatz und die Rechenleistung der verwendeten Systeme. Die Anzahl der Mitarbeiter drückt sich sowohl in der Anzahl der registrierten Benutzer (Personen) als auch in der anzunehmenden Anzahl von gleichzeitig mit dem System arbeitenden Benutzern aus. Die registrierten Benutzer haben eher Einfluß auf die benötigte Menge an persistentem Speicherplatz, die der

gleichzeitig aktiven Benutzer auf die verfügbar zu haltende Rechenleistung der Server. Die erforderliche Speicherkapazität des Systems bemißt sich überwiegend durch die Menge an verwalteten Informationsartefakten aller Art. Die Größe der zugrundeliegenden Datenbanken und die Rechnerausstattung muß dementsprechend durch Metriken kalkuliert werden können.

- Konfigurierbarkeit: Der Betrieb eines Wissensportals muß in mehreren verschiedenen Modi möglich sein, die zwischen Einzelplatzinstallationen zum Zwecke der Demonstration, der Entwicklungs-, Einführungs- und Testphase auf mittelgroßen Systemen und dem Produktivbetrieb im täglichen Einsatz großer und größter Organisationen variieren. Hier ist besonders die einfache Anpaßbarkeit an die jeweilige Konfiguration zu fordern. Da komplexe Systeme auch komplexer Einstellungen bedürfen, ist die übersichtliche und zentrale Konfigurierbarkeit des Systems eine äußerst wichtige Anforderung.

Die Skalierbarkeit bestimmt ganz wesentlich die Architektur und den Entwurf eines Portalserversystems. Hinzu kommen verschiedene operationale Anforderungen, die im folgenden genannt werden.

Y.3: Datensicherheit: Der Wert der in einem Wissensportal aggregierten Informationen ist beträchtlich, handelt es sich doch um das intellektuelle Kapital einer Organisation. Dieses muß entsprechend gegen Verlust gesichert sein, während gleichzeitig die Verfügbarkeit garantiert sein muß. Geleistet wird dies durch die Abdeckung folgender Anforderungen:

- Durchführung regelmäßiger Datensicherungen (*Backup*), die auch während des Betriebes möglich sein müssen;
- Herstellung von Ausfallsicherheit durch redundante Auslegung von Systemen (gespiegelte Datenbestände, redundante Hardware);
- Erhaltung der Datenkonsistenz im Fehlerfall (etwa bei Systemabstürzen) durch Wiederherstellung eines gesicherten Zustandes (*Recovery*);
- Laufende Archivierung gesicherter Zustände zum Zwecke der Dokumentation, insbesondere der rechtssicheren Ablage beweiskräftiger Dokumente soweit erforderlich (Revisionssicherheit).

Weite Teile dieser Anforderungen lassen sich durch die Verwendung von in typischen Infrastrukturen ohnehin vorhandenen Mitteln (Datenbanken, Backup-Systeme, unterbrechungsfreie Stromversorgung etc.) leicht erfüllen.

Y.4: Zugriffssicherheit und Übertragungssicherheit: Neben der Datensicherheit (Sicherung gegen Verlust, Herstellen der Verfügbarkeit) ist die Zugriffssicherheit sowie die Übertragungssicherheit im Sinne des Datenschutzes, der informationellen Selbstbestimmung und der Vertraulichkeit persönlicher und unternehmensrelevanter Informationen ein weiterer wichtiger Aspekt des Themas Sicherheit.

- Bereits in Anforderung P.2 wurden Personen, Gruppen und Rollen als Einheiten der Rechtevergabe aufgeführt. Die Grundlage aller Maßnahmen für die Zugriffssicherheit ist die Existenz eines feingranularen rollenbasierten Rechtekonzepts, über dessen Verwendung genau definiert werden kann, welche Personen Zugang zum System überhaupt und zu welchen Teilen und Funktionalitäten besitzen.
- Gerade mit den Informationen der persönlichen Profile, insbesondere den privaten Anmerkungen und Bewertungen sowie den Benutzungsstatistiken, muß sehr sorgfältig umgegangen werden. Schon der Anschein einer Überwachung der Produktivität der Benutzer ist sowohl rechtlich problematisch als auch ein potentielles Motivationshindernis.

- Um umfangreiche Benutzergruppen administrieren zu können, muß das Berechtigungssystem des Wissensportals in bestehende Systeme zur Benutzerverwaltung integrierbar sein. Dazu muß es Verzeichnisdienste wie LDAP, NIS oder ACTIVE DIRECTORY ansprechen und automatisch zur Authentifizierung der anzumeldenden Benutzer verwenden. Die Integration in die Anmeldungsprozeduren der von den jeweiligen Benutzern verwendeten Rechnern und Betriebssystemen ist wünschenswert, da damit die Authentifizierung und Autorisation nur einmal beim Anmelden an den Rechner geleistet werden muß und fortan bei der Benutzung weiterer Systeme entfallen kann (*single sign on*).
- Die Übertragungssicherheit ist besonders bei einem web-basierten Portalsystem ein kritischer Punkt, wenn das System auch von außerhalb der Organisation über das Internet genutzt werden soll. Moderne Sicherungsverfahren wie das gesicherte HTTP (S-HTTP) müssen dann zum Einsatz kommen [RS99].
- Um die Zugriffssicherheit zu verbessern und die Möglichkeiten zu gezielten Angriffen und Ausspähungen von Informationen zu verringern, muß ein Portalserver zum Zugriff berechtigte Rechner und Netzwerkdomänen verwalten und respektieren. Bei wie in Abbildung 3.7 dargestellten Szenarien, in denen ein Server gleichermaßen Intranet, Extranet und Internet bedient, ist eine Unterscheidung der Zugangsrechte aufgrund der anfragenden Netzwerkdomäne eine zusätzliche Sicherungsmaßnahme, die beispielsweise das Ausspähen von internen Benutzerkennungen und deren (mißbräuchliche) Verwendung über einen externen Zugang wirksam verhindert. Der Betrieb mehrerer virtueller Web-Server unter verschiedenen Adressen, die durch *Firewalls* unterschiedlich abgeschirmt werden, ist eine weitere flankierende Maßnahme. Ein Portalserver muß also in dieser Hinsicht umfangreiche Konfigurationsmöglichkeiten besitzen.
- Eine hauptsächlich im Interesse der Hersteller von Portalservern liegende Anforderung ist die Durchsetzung von technisch orientierten Lizenzbedingungen wie maximaler Benutzeranzahl, Datenvolumen, Anzahl von (gleichzeitigen) Verbindungen oder Skalierung durch redundante Auslegung und parallelen Betrieb von mehreren Servern.

Die Beachtung der Sicherheitsrichtlinien und Zugriffseinschränkungen stellt insbesondere eine für die Implementierungsphase wichtige und sorgfältig durchzuführende Aufgabe dar. Werden Portalsysteme im kommerziellen Umfeld für das Angebot von Informationsartefakten zum Kauf oder zur Miete (als Beispiel für Information als Wirtschaftsgut) verwendet, so wird die Sicherstellung der Zugriffsrechte zu einer zentralen fachlichen Anforderung.

Y.5: Plattformen: Aus den unterschiedlichen Einsatzszenarien resultiert die Notwendigkeit, eine möglichst große Zahl von Plattformen zu unterstützen. Zu unterscheiden ist dabei zunächst nach Plattformen für Client und Server und dann nach Betriebssystemen.

Da als *Frontend* eines web-basierten Portals naturgemäß Web-Browser zum Einsatz kommen, reduziert sich die Frage der Plattformen auf der Seite der Clients hauptsächlich auf die der Web-Browser. Die erhältlichen und gängigen Web-Browser unterscheiden sich quer durch alle Betriebssysteme zum Teil erheblich in der Art und Weise der Interpretation der verwendeten Standards. Um die Benutzbarkeit der Oberfläche mit allen Systemen sicherzustellen, muß das System soweit Unterstützung abieten, daß diese Unterschiede ohne hohen Aufwand berücksichtigt werden können.

Die Ablauffähigkeit eines Portalservers sollte auf möglichst vielen Plattformen, also Hardwarearchitekturen und Betriebssystemen, möglich sein. Wichtige Betriebssysteme wie MI-

CROSOFT WINDOWS und diverse Unix-Varianten wie SUN SOLARIS sind als Mindestanforderung zu nennen, um der Wirklichkeit der verwendeten IT-Landschaften Rechnung zu tragen.

Y.6: Interoperabilität: Unter dem Begriff Interoperabilität wird der Tatsache Rechnung getragen, daß ein Wissensportal in konkreten Anwendungsprojekten fast immer in bestehende informationstechnische Infrastrukturen eingebettet werden muß und zum großen Teil von diesen abhängt. Die Umsetzung des in Abschnitt 4.3.1 geforderten generalisierten Dokumentenbegriffs bedeutet auf der technischen Ebene unter Umständen die Integration einer Vielzahl von bestehenden Systemen, um die darin abgelegte Information dem Portal zu erschließen. Auf der anderen Seite kann ein Portalserver nicht als abgeschlossenes, monolithisches System entworfen und realisiert werden, sondern muß sich ab einer gewissen Ebene und für bestimmte Funktionen auf bestehende Lösungen abstützen können, beispielsweise zur Sicherung der Persistenz auf Datenbanken oder zur Kommunikation auf e-Mail-Server. Auch der häufig nötige Datenaustausch mit anderen Systemen zu unvorhergesehenen Zwecken erfordert die Bereitstellung von Informationen in bestimmten Austauschformaten. Die Interoperabilität ist also in folgende Bereiche zu unterteilen:

1. Integration von Systemen: Wichtige Systeme, deren Informationen ein Wissensportal integrieren und die es deswegen gezielt ansprechen können muß, sind:
   - Datenbanken: Hier sind insbesondere relationale Datenbanken, die über SQL ansprechbar sind, wichtig. Relevante Produkte sind etwa ORACLE, DB2, MS SQL SERVER, ADABAS oder INFORMIX, aber auch das freie Datenbanksystem MYSQL.
   - Kommunikationsserver: Diese umfassen sowohl e-Mail-Server als auch Produkte zur Unterstützung der Gruppenarbeit wie LOTUS NOTES, MS OUTLOOK oder MS EXCHANGE.
   - Externe WWW-Quellen: Hiermit sind insbesondere Web-Server und *News*-Server gemeint, die ein Portalserver (bzw. seine Erschließungskomponente) in der Rolle eines Clients ansprechen können muß.
   - Interne Quellen: Die wichtigsten internen Quellen sind Dateisysteme, die typischerweise von dedizierten Datei-Servern angeboten werden. Diese müssen angesprochen werden können; wichtiger aber ist, daß eine Vielzahl von Dokumentenformaten zur tiefen semantischen Erschließung erkannt werden können. Darunter fallen HTML, XML, die gebräuchlichsten Formate der gängigen *Office*-Anwendungen, PDF, diverse Bildformate und viele mehr.
   - Spezielle Systeme: Gerade in dem Szenario, das in Abschnitt 3.3.4 entwickelt wurde, ist die Einbindung bestehende Systeme für ERP, SCM, CRM etc. nötig. Dies kann nur über spezielle Adapter und Anpassungen geschehen, die für konkrete Projekte und deren Anforderungen entwickelt werden müssen. Schnittstellen zu Systemen wie SAP R/3 sind hier besonders wichtig.
   - Nachrichten: Weitere spezielle Quellen können Nachrichtenströme (*Newsfeed*) von Nachrichtenagenturen sein, die ständig aktualisierte Meldungen zu weiten Themenbereichen anbieten. Die Integration und Klassifikation dieser Meldungen (siehe dazu etwa [Mor00]) kann in einem Portal nützlich sein; diese Meldungen ließen sich als Dokumente einbinden. Genauso ließen sich Informationen über Börsenkurse einbinden.

   Ein wichtiger Standard zur Integration ist in dem *Information and Content Exchange Protocol* (ICE) zu sehen [WWW98b].
2. Nutzung von Systemen: Zur Erfüllung bestimmter Aufgaben muß ein Portalserver auf weitere Dienste zurückgreifen können. Dazu gehören:

- Sicherung der Persistenz aller vom Portalserver erzeugten und verwalteten Daten (Metainformationen aller Informationsartefakte und diese evtl. auch selbst, Betriebsdaten etc.). Dazu bietet sich die Verwendung relationaler oder objektorientierter Datenbanken an, da bereits ausgereifte, professionelle kommerzielle Produkte existieren, die allen oben genannten Anforderungen hinsichtlich Skalierbarkeit und Datensicherheit umfassend gerecht werden. Über generische Schnittstellen wie ODBC oder JDBC lassen sich alle wichtigen Datenbankprodukte ohne größere Probleme anbinden, so daß in diesem Punkt weitgehende Produktunabhängigkeit zu erzielen ist.
- Nutzung der Kommunikationsinfrastruktur für die aktiven Dienste des Wissensportals wie Wissenszustellung. Die Anbindung an bestehende Systeme ist über die Verwendung standardisierter Protokolle wie NNTP, SMTP, POP und IMAP möglich.
- Verwendung von Dateisystemen zur Ablage von Massendaten außerhalb der o.g. Datenbanksysteme. Die Anbindung sollte normalerweise unproblematisch sein.
- Nutzung von Verzeichnisdiensten wie LDAP, NIS oder WINDOWS NT-Benutzerverzeichnissen. Diese Funktionalität wurde in den Anforderungen P.5 und Y.4 bereits genannt.

3. Austauschformate: Sowohl zur Integration der Informationen aus fremden Systemen als auch zum manuellen, fallweisen Austausch von Informationen mit anderen Systemen ist es nötig, daß ein Portalserver wichtige Austauschformate für die in ihm enthaltenen Informationen unterstützt. Der Austausch wird im letzteren Falle ad hoc und per Hand von berechtigten Benutzern eingeleitet. Diese Möglichkeit trägt der Anforderung nach Unterstützung der traditionellen Arbeitsweise Rechnung (vgl. Anforderung D.9). Eine häufig vorkommende und praxisnahe Situation ist beispielsweise, daß bestimmte Informationen extrahiert werden und zur weiteren Bearbeitung und zur Analyse in ein Tabellenkalkulationsprogramm übertragen werden.

   Zu fordern ist also zum einen, daß alle relevanten Daten, die ein Portalserver aggregiert (also sowohl die vollständigen Dokumente inklusive aller Metainformationen als auch alle weiteren Informationsartefakte, Betriebsdaten etc.) über einen einfachen Mechanismus in leicht weiterverarbeitbaren Formaten exportiert werden können, und zum anderen die Unterstützung von für bestimmte Zwecke verbreiteten Austauschformaten für das Exportieren und Importieren. Für die erste Forderung bietet sich ein Export der Daten in roher, tabellarischer Textform sowie neuerdings in speziellen XML-Dialekten an. Ein wichtiges Formate für den Austausch von Wissensnetzen sind mittlerweile die *Topic Maps* [Top01]; siehe dazu auch Anforderung B.6 sowie ferner [WWW98b].

4. Schnittstellen: Diese erfüllen grundsätzlich Aufgaben mit der gleichen Intention wie die Austauschformate; hier liegt der Fokus aber in der Ansprechbarkeit durch Software und nicht durch manuelle Aktionen. Dazu muß ein Portalserver Software-Schnittstellen besitzen, die von weiterer Software angesprochen werden kann. Neben der für Web-Server originären Schnittstelle durch das Protokoll HTTP sind eine Reihe von weiteren zu fordern: Einerseits müssen z.B. wichtige auf der HTTP-Schnittstelle aufbauende Standards unterstützt werden (beispielsweise WEBDAV, das das verteilte Publizieren ermöglicht [GWF+99, Web01a] und somit den entfernten Zugriff erlaubt), andererseits muß durch reine Software-Schnittstellen der Zugriff auf die gesamte Funktionalität eines Portalservers ermöglicht werden. Zur Realisierung einer Schnittstelle für Anwendungssoftware (*Application Programming Interface*, API) bieten sich diverse (sich nicht ausschließende) grundsätzliche Möglichkeiten wie CORBA, COM, *Java Enterprise Beans*, SOAP oder direkte Bibliotheksschnittstellen für bei-

spielsweise JAVA, C oder C++ an, auf deren genaue Bedeutung und Funktionsweise hier aber nicht weiter eingegangen werden kann. Grundsätzlich muß ein Portalserver aber zumindest eine Art von offener, wohldokumentierter Schnittstelle haben, die die Erstellung von Erweiterungen für Dritte gestattet.

5. Synchronisation: Ähnlich wie bei der Interoperabilität eines Portalservers mit fremden Systemen kann der Server mit anderen Servern der gleichen Art Informationen austauschen. Die in jeweils zwei Portalservern vorhandenen Informationen können dabei miteinander abgeglichen (synchronisiert) werden, so daß beide danach identische Informationsbestände besitzen. Dies ist nicht nur im Falle der räumlich weiten Verteilung mehrerer Portalserver zur Lastverteilung nötig, sondern kann auch zur Verwendung von lokale Kopien eines stationären Portals gebracht werden. Die Kopien können dann etwa auf mobilen Rechner ohne Netzwerkzugang (also ohne Zugang zum ursprünglichen Portal) zunächst unabhängig benutzt und verändert werden, um die Inhalte bei Gelegenheit wieder abzugleichen. Bei der Erstellung lokaler Kopien und dem späteren Abgleich ist eine Strategie nötig, um Änderungen und Konflikte wirkungsvoll erkennen und (evtl. manuell) auflösen zu können. Ferner ist u.U. nötig, den Umfang von Kopien gezielt einzuschränken, um überhaupt transportable Datenmengen zu erhalten.

Auf Fragen der Interoperabilität gibt es viele Antworten, so daß keine wirklich allgemeingültigen Anforderungen gestellt werden können. Die oben genannten fassen lediglich die am häufigsten in der Praxis beobachteten bzw. die am häufigsten in realen Systemen implementierten Fähigkeiten zusammen. Letztlich entscheidet auch die geschäftliche Strategie der jeweiligen Produkthersteller über die jeweils angebotene Ausprägung der Interoperabilität. Zudem bedingt der jeweilige Einsatzzweck und die im praktischen Einsatz vorhandene Umgebung die weiteren genauen Anforderungen, die deshalb stark divergieren können.

Die – immer auf organisatorischen Randbedingungen beruhenden – Systemanforderungen fallen teilweise sehr technisch aus und konnten deswegen in diesem Rahmen nur angerissen werden. Ein pragmatischer Systementwurf muß das technische und organisatorische Umfeld stets berücksichtigen, um erfolgreich zu sein.

4.6 Zusammenfassung

In den letzten drei Abschnitten wurden umfassende Anforderungen an ein Wissensportal in Form eines detaillierten und strukturierten Katalogs formuliert, die aus den Grundlagen des Wissensmanagements in Kapitel 2 und den Untersuchungen der Systemklassen und Produkte in Kapitel 3 abgeleitet wurden. Durch die Aufstellung von neun fachlichen Anforderungsgruppen, einer die Oberfläche und einer das Systemmanagement betreffenden weiteren Anforderungsgruppe ist die Erfüllung der anfangs in Abschnitt 4.1 aufgestellten Zielvorgaben erreichbar:

- Die Unterstützung des Wissenszyklus zur Erfassung, Aufbereitung und Nutzung expliziten und impliziten Wissens wird durch die produktorientierten Konzepte der Personen, Dokumente, Verzeichnisse und Begriffe sowie die prozeßorientierten der Personalisierung, Suche, Zustellung, Entdeckung, Gemeinschaften und Prozesse geleistet.
- Der generalisierte Dokumentenbegriff und die uniforme Modellierung aller Informationsartefakte gestatten die Erfüllung des Ziels der Integration strukturierter, semistrukturierter und unstrukturierter Informationen aus beliebigen Quellen.

- Die Konzepte der Verzeichnisse, der Begriffe und die damit mögliche tiefe Erschließung bieten multidimensionale Leitsysteme zum vorhandenen expliziten und impliziten Wissen an.
- Die Ziele der Verfügbarkeit, Konsistenz, Aktualität, Relevanz, Bedienbarkeit, Wartbarkeit, Erweiterbarkeit und Skalierbarkeit sowie die Integration bestehender heterogener Systemlandschaften werden durch umfangreiche Anforderungen an das Systemmanagement abgedeckt.

Aus den Anforderungen ergeben sich einige Einsichten, die sich konsequent an die des Kapitels 3 in Abschnitt 3.5 anschließen.

1. Die uniforme Modellierung aller integrierter Informationsartefakte durch ein generalisiertes Dokumentenmodell ist das Schlüsselkonzept zur Erschließung aller semantischer Beziehungen. Nur durch die Gleichbehandlung aller Objekte sind auch alle gleichermaßen nützlich, weil sie erst dadurch den Basisoperationen wie Klassifikation und Suche zugänglich werden.
2. Die Vermeidung konzeptueller Brüche wie z.B. durch die explizite Modellierung von Personen als einem zu Dokumenten, Verzeichnissen und Begriffen gleichberechtigten Konzept gestattet die Erschließung impliziten Wissens und die tiefe Erschließung aller semantischer Beziehungen durch die gezielte Einbringung des konzeptuellen (Welt-)Wissens in das Modell.
3. Als technische Grundvoraussetzung für fast alle Klassifikations- und Kategorisierungsaufgaben, Personalisierungsfunktionen und für die Verwaltung aller weiterer Beziehungen ist eine bidirektionale Verweisverwaltung (*Link Management*) zwingend erforderlich, um Konsistenz und Aktualität zu wahren. Eine Verknüpfung ist ebenfalls als selbständiges Informationsartefakt zu modellieren, um personalisierte Sichten der Verknüpfungen aufbauen zu können. Eine Trennung von Dokumenteninhalt, Metainformationen und Beziehungen erweist sich als nötig.
4. Die Anforderungen lassen sich gut in verschiedene, prägnant zu bezeichnende Anforderungsgruppen differenzieren. Dennoch wird immer wieder ein ständiger enger Zusammenhang zwischen vielen der Einzelanforderungen sichtbar, der beinahe jede Anforderung auf alle anderen wirken läßt und teilweise komplementäre Sichtweisen auf Anforderungen anderer Gruppen eröffnet.

Diese Einsichten bestimmen die im folgenden Kapitel vorgestellten Modelle und die in Kapitel 6 präsentierte Architektur und den Systementwurf eines auf Grundlage des Anforderungskatalogs und der Modelle des nächsten Kapitels realisierten Wissensportals maßgeblich.

Kapitel 5

Integrierte Dokumenten- und Prozeßmodelle

Modelle sollten sich bemühen, dem Porträt ähnlich zu sehen.
– SALVADOR DALI

Ein guter Systementwurf für ein derartig komplexes System wie es in Kapitel 4 gefordert wird, bedarf über den Architekturentwurf hinaus (der gemeinhin als oberste Strukturierungsebene des softwaretechnischen Entwurfs gilt) der Bereitstellung eines den Entwurf leitenden Modells auf einer Ebene, die zwischen den fachlichen, benutzerorientierten Anforderungen, die durch ein konzeptuelle Modell ausgedrückt werden, dem eigentlichen softwaretechnischen Entwurf und seiner Implementierung auf der einen Seite und den verschiedenen zu integrierenden Systemen und Modellen auf der anderen Seite vermittelt. Dieses eigentlich als Metamodell zu bezeichnende Modell federt so die Modell- und Medienbrüche wirkungsvoll ab und schließt die in Kapitel 3 identifizierte Lücke des problemadäquaten Dokumentenmodells, das zur bruchlosen Integration verschiedener Informationsartefakte aus heterogenen Systemlandschaften in einem Wissensportal nötig ist.

Eine beherrschende Frage, die sich bereits durch die letzten Kapitel hindurchzog, war die nach den hinter den Systemen liegenden Modellen, insbesondere nach Dokumentenmodellen. Da die vorkommenden Informationsartefakte (Dokumente, Verzeichnisse, Personen, Begriffe etc.) naturgemäß im Vordergrund des Interesses bei Untersuchungen von Wissensmanagement-Systemen (und auch CMS und DMS) stehen, ergaben sich diesbezüglich bereits einige wichtige Ergebnisse (vgl. z.B. Abschnitt 4.6). Diese werden im folgenden Abschnitt 5.1 weiter diskutiert und ausgebaut.

Aufgrund des beabsichtigten Einsatzzweckes als Wissensportal auf der einen Seite und der geforderten Möglichkeiten zur Integration von Systemen, der pragmatischen Notwendigkeit der Verteilung bzw. der ohnehin bei einem Client/Server-System inhärenten Verteilung auf der anderen Seite, kann das zu entwerfende System in vielen Aspekten als kooperatives Informationssystem aufgefaßt werden. Die Erkenntnisse, die bei der Modellierung solcher Systeme und der Untersuchung von zugehörigen Dokumenten- und Prozeßmodellen gewonnen wurden, sollen hier angewendet werden. Gerade die in [WHM00, Hup00, MWH99, HMW98, Weg98, Joh97] formulierten Ergebnisse werden sowohl das Dokumentenmodell als auch die Prozeßsicht nachhaltig beeinflussen. Das neben dem Dokumentenmodell ebenso wichtige abstrakte Prozeßmodell wird in Abschnitt 5.2 behandelt.

5.1 Dokumentenmodell

Eines der Kernkonzepte aller untersuchten Systemklassen (vgl. Abschnitt 3.3) und Systeme (vgl. Abschnitt 3.4) ist das jeweilige Modell für die verwalteten Informationsartefakte, kurz Dokumentenmodell genannt. Die Bezeichnung Dokumentenmodell ist deswegen zutreffend, weil die Informationsartefakte zumeist Dokument genannt wurden und die historische Entwicklung bei unstrukturierten Modellen, die das Dokument als kleinste Einheit sehen, ihren Ausgangspunkt hat. Ziel dieses Abschnittes ist es nun, aus der logischen Fortentwicklung der bereits ansatzweise beleuchteten Modelle ein Dokumentenmodell zu entwickeln, das einerseits den in Abschnitt 4.1 aufgestellten Zielsetzungen und den nachfolgend formulierten Anforderungen gerecht wird und andererseits bereits mögliche Entwurfs- und Implementierungsaspekte ausreichend berücksichtigt.

Die wichtigsten das Dokumentenmodell betreffende Ziele und Anforderungen sind nachfolgend nochmals kurz aufgeführt:

1. Integration der Informationsartefakte heterogener Systemlandschaften;
2. Integration von strukturierten, semistrukturierten und unstrukturierten Informationen;
3. Uniforme Behandlung aller Informationsartefakte;
4. Trennung von Struktur, Inhalt, Layout und Funktionalität;
5. Konzeptionelle Trennung der Daten, Metadaten und Beziehungen der modellierten fachlichen Domäne;
6. Dynamische Erweiterbarkeit der Dokumententypen und Möglichkeit der Definition neuer Dokumententypen.

Die Lösung dieser Aufgabe ist in dem Spannungsfeld zwischen objektorientierten, relationalen und dokumentenorientierten Modellen angesiedelt. Um den zur Verfügung stehenden Lösungsraum besser zu verstehen, werden die wichtigsten Eigenschaften, Konzepte und Qualitäten der einzelnen den Raum begrenzenden Modelle zunächst summarisch aufgeführt. Die Kriterien zur Bewertung der für die obigen Anforderungen relevanten Qualitäten sind dabei:

- **Abstraktion:** Der Grad des durch das Modell möglichen Abstraktionsniveaus. Das beschreibt zugleich die Integrationsfähigkeit des Modells bezüglich der Anforderungen 1, 2 und 3.
- **Separation:** Die Fähigkeit, innerhalb des Modells die verschiedenen konzeptionellen Ebenen zu trennen. Dies trägt den Anforderungen 4 und 5 Rechnung.
- **Dynamik:** Die Möglichkeiten der dynamischen Veränderbarkeit der Dokumenttypen insbesondere zur Laufzeit des Systems. Dies betrifft die letzte Anforderung 6.

Auf der Grundlage der Bewertungen wird daraufhin ein integratives Dokumentenmodell für ein Wissensportal erstellt.

5.1.1 Relationales Datenmodell

Das relationale Datenmodell [Dat95, Cod70] ist wegen seiner Einfachheit und guten mathematischen Handhabbarkeit nicht nur in der Forschung beliebt, sondern auch in der kommerziellen Praxis durch relationale Datenbanksysteme mittlerweile weit verbreitet. Die Kernkonzepte sind die der Relation (Tabelle) und des Tupels. Mathematisch gesehen sind die Relationen über dem Kreuzprodukt diverser Domänen aufgebaut, die die zulässigen Wertebereiche (Typen) der Tupel-Elemente beschreiben. Alle Tupel einer Tabelle (die Zeilen) haben dabei zwingend dieselbe Struktur (die Spalten). Die Identifizierung der einzelnen Tupel geschieht wertbasiert und über Schlüsselattribute. Die relationale Modellierung geht üblicherweise von der Repräsentation eines Entitätstyps pro Relation aus. Beziehungen zwischen Relationen werden durch die Konzepte der Primärschlüssel und Fremdschlüssel ausgedrückt, also durch assoziative Identifikation (im Gegensatz zu referentieller Identifikation etwa im objektorientierten Modell). Durch prädikatenlogische Integritätsbedingungen wird die Konsistenz der Informationen garantiert. Die Struktur aller Relationen wird durch ein Schema festgelegt. Weitere Modellierungskonstrukte existieren nicht, wenn man von neueren Entwicklungen wie objektrelationalen Modellen bzw. Erweiterungen absieht. Die Modellelemente sind in dem Metamodell-Diagramm in Abbildung 5.1 wiedergegeben.

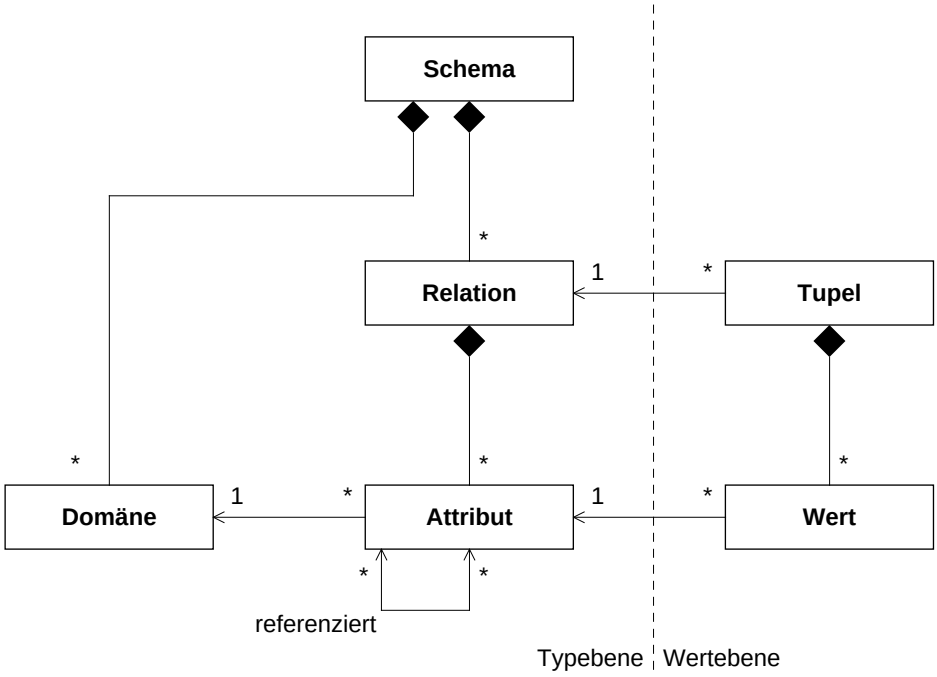

Abbildung 5.1: Metamodell der relationalen Konzepte

Die Datenbanksprache SQL hat sich zum de-facto-Standard zur Benutzung relationaler Datenbanksysteme entwickelt und umfaßt Konstrukte zur Schemadefinition, zur Schemaänderung, zur deskriptiven Formulierung von Anfragen und zur Modifikation der Datenbank [SM98, KE97, EN89, LS87].

Die Abstraktionsmöglichkeiten sind durch die einfache Struktur von regulär aufgebauten Tabellen und die wertbasierte Identifizierung auf relativ niedrigem Niveau. Dies erfordert zur Darstellung komplexer Beziehungsgeflechte zahlreiche Hilfstabellen und die unnatürliche, relativ aufwendige Abbildung komplexer Datenstrukturen auf flache, unstrukturierte Attributmengen.

Auf der anderen Seite muß konzediert werden, daß damit dennoch die große Masse an Modellierungsaufgaben gut lösbar ist, da insbesondere in betrieblichen Informationssystemen, die zu den Hauptnutzern relationaler Datenbanksysteme gehören, große Problembereiche recht regulär aufgebaut sind und komplexe Datenstrukturen eher selten vorkommen.

Die Separation der Ebenen ist in relationalen Datenbanken einerseits dadurch gut vorhanden, daß Struktur (also Schema) und Daten (also Tabellen) unterschiedliche Konzepte darstellen, die auch durch verschiedene Sprachmittel in SQL ausgedrückt und verwendet werden können; andererseits fehlt durch die reine Datenzentrierung des Modelle ohnehin die Ebene der Darstellung. Eine Trennung zwischen Daten und Metadaten ist nicht vorgesehen. Beziehungen zwischen Tupeln können (und müssen z.T. auch bei Beziehungen höherer Kardinalität) durch Hilfstabellen von den eigentlichen Daten getrennt werden.

Die Dynamik üblicher SQL-basierter relationaler Datenbanksysteme ist sehr gut; die Schemadefinition und -veränderung ist zur Laufzeit möglich und die Schemainformationen lassen sich zur Laufzeit reflektiv aus bestimmten Systemtabellen extrahieren.

Des weiteren liegen die Qualitäten der real existierenden relationalen Datenbanksysteme in der Fähigkeit, große und sehr große Datenmengen und Änderungsfrequenzen bei gleichzeitiger Sicherstellung der Wiederherstellbarkeit im Fehlerfalle und der Wahrung der Konsistenz beim nebenläufigen Zugriff bewältigen zu können.

5.1.2 Objektorientiertes Modell

Das objektorientierte Paradigma als Programmiersprachenparadigma ist bereits relativ alt. Neuer hingegen sind die darauf basierende objektorientierte Analyse, Modellierung und Entwurf [Mey88, RBP+91, JCJÖ92, Boo94]. Die mittlerweile als de-facto-Standard zu verstehende *Unified Modelling Language* (UML) hat zu einer weiteren Vereinheitlichung der Terminologie und Konzepte geführt [JBR99, RJB99, BRJ99]. Die wichtigsten Konzepte objektorientierter Datenmodelle sind Typkonstruktoren und komplexe Objekte, die rekursiv aus einfachen Basisdatentypen aufgebaut werden, Klassen und Vererbung, die zur Beschreibung von Generalisierungen und Spezialisierungen einer Taxonomie dienen, Methodenredefinition und späte Bindung, die das Subsumptionsprinzip realisieren und die Erweiterbarkeit bestehender Klassenhierarchien erlauben, Objektidentität, die strikte Referenzsemantik einführt, Assoziationen zwischen Klassen und ihren Exemplaren, die die Referenzsemantik ausnutzen, sowie die Kapselung durch Methoden, die den Zustand verbergen, die Funktionalität eines Objektes ausmachen und zugleich eine Schnittstelle definieren.

Die Abstraktionsmöglichkeiten objektorientierter Modelle sind sehr hoch. Insbesondere der Mechanismus der Vererbung sowie der Bündelung der beschreibenden Attribute und der auf den Objekten einer Klasse möglichen Operationen, also von Zustand und Verhalten, erlaubt eine der realen Welt vergleichsweise nahe Modellierung. Die Mechanismen der Aggregation und Assoziation erlauben zudem eine wirklichkeitsnähere Modellierung der Zustände als dies beispielsweise im relationalen Modell gelingen kann. Die wichtigsten Eigenschaften sind in Abbildung 5.2 in einem Metamodell dargestellt.

Die Separation insbesondere von Inhalt und Funktionalität ist in objektorientierten Modellen inhärenterweise gerade nicht gegeben. Der Aspekt der Darstellung und der Metadaten entfällt ebenso wie im relationalen Modell. Die Struktur und die Dynamik sind Eigenschaften, die von dem konkreten System abhängen. Die Struktur der objektorientiert modellierten Informationsobjekte ist in Programmiersprachen entweder durch statische Typisierung bereits vor der Laufzeit unveränderbar festgelegt (etwa in C++, JAVA oder TYCOON-2), oder aber in dynamisch typisier-

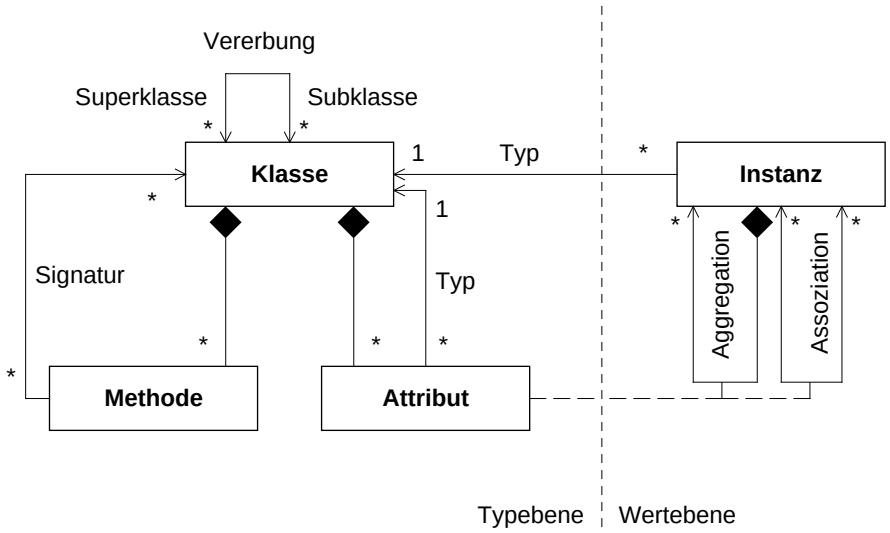

Abbildung 5.2: Metamodell der objektorientierten Konzepte

ten Sprachen mit dynamischen Datentypen erst zur Laufzeit (z.B. in JAVASCRIPT). Auf jeden Fall ist eine strikte Trennung von Inhalten (Werten) vorhanden, wobei aber in den meisten modernen Sprache die Typen reflektiv inspizierbar sind. Die Änderbarkeit zur Laufzeit ist in den meisten statisch typisierten Sprachen nicht gegeben. Eine Ausnahme stellt das TYCOON- [Mat93] bzw. TYCOON-2-System dar [GW98, Wah98, GM95], in dem der Übersetzer, alle Typinformationen und die übersetzten Klassen orthogonal als Objekte erster Ordnung in einem gemeinsamen persistenten Objektspeicher vorliegen und so im Prinzip auch zur Laufzeit manipulierbar sind; eine Eigenschaft, die sich Übersetzer und Typprüfer bereits zunutze machen [Wei98, Ern98, Wie97].

5.1.3 Dokumentenorientierte Modelle

Im Gegensatz zu den weitgehend unstrittig definierten relationalen und objektorientierten Modellen stellt sich die Situation bei dokumentenorientierten Modellen grundlegend anders dar. Eine einheitliche Wahrnehmung des Begriffs „dokumentenorientiert“ ist nicht festzustellen; bereits der Begriff wird nicht einmal immer verwendet. Die hier eingenommene Sichtweise knüpft an drei häufiger vorgefundene Wahrnehmungen der Dokumentenorientierung an:

1. Als dokumentenorientiert werden oft die graphischen Benutzungsoberflächen zur Handhabung von Dateisystemen und von bestimmten Bereichen in Betriebssystemen bezeichnet. Die Metapher eines Dokuments, das in einer Datei gespeichert ist und mit Werkzeugen wie Textverarbeitung etc. bearbeitet werden kann, hat sich weit verbreitet und wird von den Benutzern offenbar verstanden und angenommen. Diese Sichtweise ist also eine sehr anwendernahe.

2. Eine zusätzlich mehr technisch geprägte Sicht wurde durch das oft dokumentenorientiert genannte Konzept von LOTUS NOTES eingeführt, wo als zentrales Artefakt das Dokument als eine durch Felder flach strukturierte Einheit von Informationen angesehen wird. Diese Modellierung ist insofern richtungsweisend, als daß erstmals ein strukturiertes Dokumentenmodell gebraucht wurde, aber die Benutzungsmetapher des Dokuments verwendet wurde. Die Konzepte von NOTES lassen sich relativ gut auf das relationale Modell abbil-

den, mit dem wichtigen Unterschied, daß eine explizite Schemadefinition nicht stattfindet [Car99].

3. Die bereits in Abschnitt 3.2.1 untersuchten semistrukturierten Informationen besitzen stark konzeptuell geprägte Sichtweisen. Während strukturierte Informationen im wesentlichen durch relationale und objektorientierte Modelle abgedeckt sind, finden sich im Bereich der dokumentenorientierten Modelle vor allen Dingen semistrukturierte und unstrukturierte Konzepte wieder. Wichtige Beiträge liefern hier in letzter Zeit besonders die Anstrengungen zur Definition der *Extensible Markup Language* XML und ihrer konkreten Ausprägungen. Der Gedanke des Austauschs von Dokumenten steht dabei oft im Vordergrund des Interesses.

Gänzlich unstrukturierte Dokumentenmodelle – monolithische Dateien ohne (bekannte) Struktur – werden im folgenden nicht weiter betrachtet. Die wichtigste Gemeinsamkeit aller dokumentenorientierten Modelle liegt darin, daß es sich um eine rein datenzentrierte Sichtweise handelt, etwa im Gegensatz zu den objektorientierten Modellen, die auch die Funktionalität in die Klassen integrieren. Ähnlich wie relationale Modelle haben Dokumente (vergleichbar mit Datensätzen) eine Reihe von flachen Attributen. Der XML-orientierte Dokumentenbegriff hingegen sieht zusätzlich den rekursiven Aufbau von beliebigen Strukturen (Wiederholungen, geschachtelte Substrukturen) vor. Dazu kann jedes Attribut wiederum weitere Attribute aggregieren. Der Aufbau muß darüber hinaus nicht zwingend einem fest vorgegebenem Schema entsprechen (Wohlgeformtheit), wodurch sich das Dokument bezüglich seiner Struktur selbst definiert. Alternativ kann es aber strikte Gültigkeit bezüglich einer vorgegebenen Grammatik bzw. Dokumententypdefinition besitzen [WWW98a, Tol99]. Weiterhin lassen sich die Definitionen von Dokumententypen wiederum als Dokumente auffassen, die ihrerseits einer speziellen dafür konstruierten Dokumentendefinition gehorchen. XML-Schemata stellen eine Möglichkeit dar, die Definition wieder als XML-Dokument zu repräsentieren [WWW99]. Die soweit erläuterten Zusammenhänge sind in Abbildung 5.3 dargestellt.

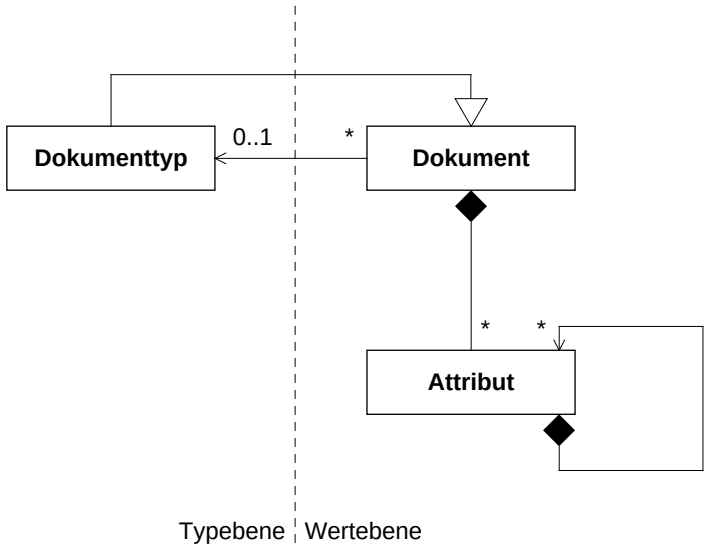

Abbildung 5.3: Metamodell der dokumentorientierten Konzepte

Die Abstraktionsmächtigkeit dieses Modells liegt über dem des relationalen Modells und unter dem objektorientierten, da einerseits komplexere Strukturen abbildbar sind, andererseits aber

Modellierungsmechanismen wie Vererbung und Assoziationen fehlen. Ferner fehlt der Aspekt der Funktionalität ganz. Die Separation von Struktur und Inhalt ist durch die Möglichkeit der expliziten Dokumenttypdefinition gegeben; die Präsentation ist üblicherweise (jedenfalls im Falle von XML-Dokumenten) strikt vom Inhalt getrennt und wird durch separate Formatierungsanweisungen für die jeweiligen Attribute geleistet, im Falle von XML-Dokumenten nämlich durch die *Extensible Stylesheet Language* (XSL) und die *XSL Transformations* (XSLT) [Dea99, Cla99]. Eine Trennung von Inhalt und Metadaten ist fester Bestandteil von XML.

Die Antwort auf die Frage nach der Dynamik des Modells hängt von der jeweiligen Ausführungsumgebung ab, in der die Dokumente verarbeitet werden. Auch wenn die Interpretation des Konzepts dokumentenorientierter Modelle die textuelle Speicherung eines Dokuments in einer Datei nahelegt, ist dies nicht die einzige mögliche Form: ebenso können Dokumente und ihre Attribute durch geeignete Zerlegung in (speziellen) Datenbanken gespeichert werden, transient oder persistent als Objekte in Softwaresystemen existieren oder durch direkte Übertragung zwischen Rechnern und Softwaresystemen in flüchtigen Repräsentationen vorliegen. In Abhängigkeit von den dort möglichen Manipulationsmöglichkeiten ist die dynamische Erzeugung und Veränderung von Dokumenttypdefinitionen möglich.

5.1.4 Ein Dokumentenmodell für ein Wissensportal

Bevor nun aus den wichtigsten Eigenschaften der drei zuvor diskutierten Modelle eine Synthese in Form eines Dokumentenmodells für ein Wissensportal gebildet wird, wird in einem kurzen Exkurs ein Dokumentenmodell (bzw. dort Inhaltsmodell genanntes Datenmodell) vorgestellt, das für die Erstellung kooperativer Internet-Informationssysteme im Rahmen von [Hup00, Weg98, Joh97] entwickelt und u.a. in [WHM00, MWH99, HMW98, Mat98, Mat97] beschrieben wurde.[1] Dieses Modell wird hier vorgestellt, da es interessante Eigenschaften besitzt, die gewinnbringend in das zu entwickelnde Dokumentenmodell integriert werden können, und weil das zu entwickelnde Portalsystem ebenfalls als kooperatives Informationssystem aufgefaßt werden kann.

Ziel war die integrierte Beschreibung und Nutzung eines Prozeß- und Dokumentenmodells zum Aufbau von interaktiven, medien- und aktorenunabhängigen Kommunikationsdiensten auf der Grundlage eines der Sprechakttheorie entlehnten Konversationsmodells. Von Interesse ist hier zunächst lediglich der Anteil des Dokumentenmodells. Zur Beschreibung der ausgetauschten Informationen wurde ein strukturiertes Inhaltsmodell entwickelt, das den Aufbau der zum Zeitpunkt der Konversation ausgetauschten Inhalte definiert. Das Modell unterscheidet strikt zwischen der Ebene der Inhaltsbeschreibungen (Typen) und der der Inhaltsexemplare (Dokumente), die durch jeweils eigene Klassen in Entwürfen und prototypischen Implementierungen realisiert wurden. Damit sind die Beschreibungen Objekte erster Ordnung im System und können zur Laufzeit erstellt und modifiziert werden. Der konzeptuelle Aufbau der Inhaltsbeschreibungen ist in Abbildung 5.4 illustriert.

Alle Inhaltsspezifikationen sind Subklasse der Klasse `ContentSpec`. Es sind verschiedene Subklassen vorhanden, die folgende Strukturierungs- und Abstraktionsmöglichkeiten anbieten:

- Atomare Basisdatentypen (Wahrheitswert, Ganzzahl, Fließkommazahl, Zeichenkette, Datum, Währung etc.);
- Auswahldatentypen, die genau einen Typ aggregieren und den Wertebereich der Inhaltsexemplare zur Laufzeit durch die Vorgabe einer beliebigen Menge von Werten dieses Typs

[1] Forschungsprojekt *Business Conversations* am Arbeitsbereich Softwaresysteme der Technischen Universität Hamburg-Harburg [Tec00].

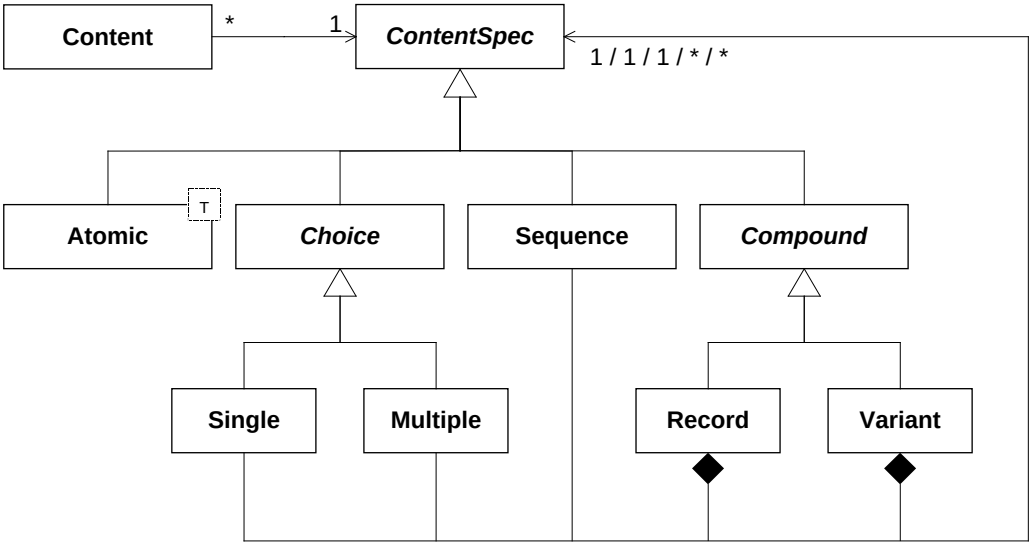

Abbildung 5.4: Inhaltsspezifikationen im Modell der *Business Conversations*

einschränken, aus der dann entweder genau einer oder beliebig viele als aktueller Wert des Exemplars gewählt werden können. Betrachtet man nur die Werte, so sind Exemplare dieses Typs äquivalent zu dem aggregierten Typ bzw. einer Sequenz davon. Die Intention dieser Konstruktion ist die Bereitstellung von für interaktive Dienste nützlichen Typen, die die unmittelbare Umsetzung in Elemente einer Benutzungsoberflächen erleichtern. Ein Exemplar des Typs Einfachauswahl läßt sich beispielsweise leicht durch sogenannte *Radiobuttons* oder *Popup*-Menüs darstellen, die die vorgegebenen Werte anbieten;

- Sequenztypen als Massendatencontainer (*Array*, Liste), die eine gleichförmige – möglicherweise leere – geordnete Menge von Exemplaren desselben Typs transportieren. Dies ist der einfachste Typkonstruktor des Modells;
- zusammengesetzte Typen, die zum rekursiven Aufbau von heterogenen, strukturierten Elementen dienen. Hier sind sowohl einfache Aggregate (*Records*, Tupel) als auch variante Strukturen [Wir86, S. 34] möglich, bei denen zur Laufzeit nur genau eine der definierten Varianten aktiv ist.

Die Modellierungsmächtigkeit dieses Ansatzes ist hoch; alle Strukturen sowohl des relationalen Modells als auch der untersuchten dokumentenorientierten Modelle lassen sich damit abbilden. Aufgrund des dialogorientierten Aufbaus der Dokumente wurden die – als Modellkonstrukt nicht vorhandenen – Assoziationen in den relevanten Systemen nicht benötigt. Die Separation von Struktur, Inhalt und Layout der Dokumente ist vollständig gegeben und wurde für die Entwicklung eines web-basierten Internet-Informationssystems erfolgreich eingesetzt [Rip98, Weg98]. Gerade die Trennung von Inhaltsspezifikation und Inhaltsexemplaren ermöglichte dabei eine extrem hohe Dynamik, die beispielsweise die Erstellung der Inhaltsspezifikationen zur Laufzeit aus HTML-Formularen gestattete oder die Verwendung von dynamischen Spezifikationskontrollen [Hup98].

Eine interessante Beobachtung ergibt sich, wenn man die Systeme analysiert, die dieses Modell eingesetzt haben oder darauf basieren (etwa [Hup00, Stö99, Car99, Rip98, Weg98, Hup98, Zad98, Kru97, Joh97, Ric97]): In den seltensten Fällen wird die Abstraktionsmächtigkeit des Modells voll ausgenutzt; fast alle jemals benutzten Spezifikationen sind lediglich flache Aggregationen von

atomaren Basisdatentypen. Selten werden Auswahlstrukturen zur Verbesserung der Benutzbarkeit sowie Sequenzen flach strukturierter oder atomarer Elemente etwa zur Übermittlung von Ergebnissen verwendet. Tiefer strukturierte Gebilde sind nicht anzutreffen. Diese somit als einigermaßen typisch erkannten Benutzungsmodalitäten werden beim Konzipieren des Dokumentenmodells berücksichtigt werden.

Nach diesem Exkurs wird jetzt die Entwicklung eines Dokumentenmodells vorgestellt, das die zu Beginn in Abschnitt 5.1 genannten Ziele und die Anforderungen aus Kapitel 4 möglichst gut erfüllt, indem die jeweils besten Eigenschaften der drei untersuchten Modelle und des Inhaltsmodells der *Business Conversations* kombiniert werden.

	RDM	OODM	DDM	BC
Abstraktion / Integration	-	++	o	+
Separation	n.a. +	n.a. o	++ +	++ o
Dynamik	+	o	+	++

Tabelle 5.1: Gegenüberstellung der Modell-Qualitäten

Eine Gegenüberstellung der untersuchten Qualitäten in Tabelle 5.1 ergibt ein uneinheitliches Bild:[2] Während das objektorientierte Datenmodell (OODM) bei der Abstraktionsfähigkeit klar am besten abschneidet, ist die Rangfolge in den anderen beiden Qualitäten unklarer: das Modell der *Business Conversations* (BC) bzw. das dokumentenorientierte Datenmodell (DDM) schneidet jeweils am besten ab, gefolgt von DDM bzw. BC, relationalen Datenmodell (RDM) und OODM, wobei die Unterschiede allerdings gering ausfallen.

Im Folgenden werden die Eigenschaften des zu erstellenden Dokumentenmodells, die besonders wertvoll und deshalb für die Erreichung der Anforderungen wesentlich sind, im einzelnen diskutiert. Die Eigenschaften des für jede Qualität jeweils am besten geeigneten Modells werden dabei integriert.

1. Die Basis soll aufgrund seiner Einfachheit das dokumentenorientierte Modell sein, wie es in Abbildung 5.3 dargestellt ist. Jedes Dokument muß allerdings zwingend einen Dokumententyp besitzen, dem es genau entspricht, so daß ad hoc definierte Dokumente nicht existieren dürfen. Diese Verschärfung erlaubt – bei geeignetem Systementwurf – trotzdem die weitgehende dynamische Veränderbarkeit und Neudefinition von Typen[3], während zugleich generische, reflektive Dienste ermöglicht werden.

2. Die rekursive Aggregation von Attributen ist ein fortschrittliches Konzept zur Strukturierung der Typen, das zumindest aus der objektorientierten Modellierung nicht mehr wegzudenken ist. Die Art der Nutzung in dokumentenorientierten Modellen ist allerdings grundlegend anders und zielt insbesondere auf semistrukturierte, nicht regulär und nicht gleichförmig aufgebaute Informationen. Der streng reguläre Aufbau der relationalen Strukturen

[2] Die Separation wird nochmals in Trennung von Struktur, Inhalt und Layout sowie Trennung von Inhalt und Metainformationen differenziert, da die Ergebnisse sonst schwerer vergleichbar sind.

[3] Im folgenden wird der Kürze halber auch von Typen statt Dokumenttypen gesprochen.

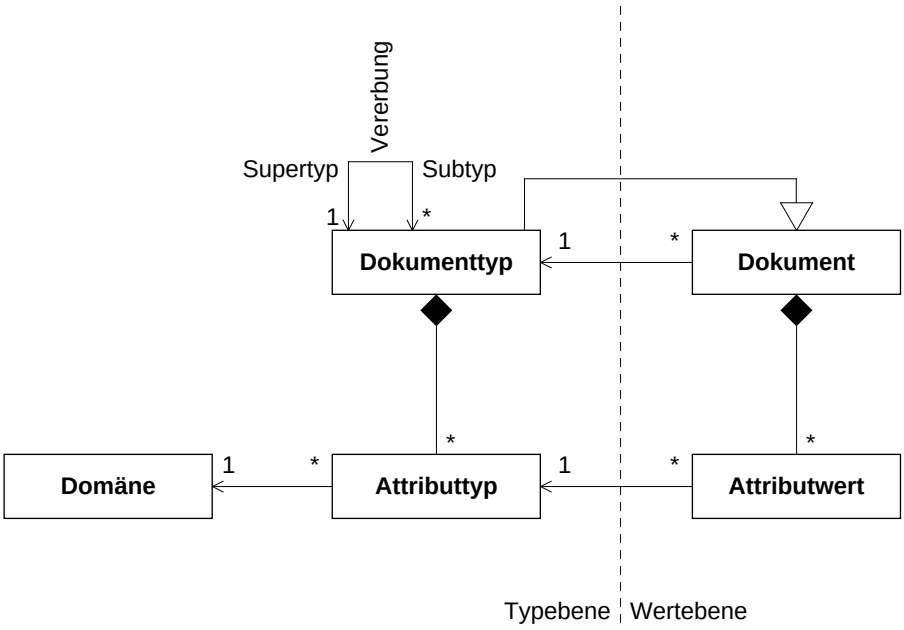

Abbildung 5.5: Ein Dokumentenmodell für ein Wissensportal

dagegen erlaubt den einfacheren Aufbau von Diensten und die einfachere Abbildung auf und Integration von heterogenen Systemen. Im zu definierenden Modell werden also nur reguläre, flach strukturierte Attribute ähnlich dem relationalen Modell gestattet. Zudem müssen fest vorgegebene Domänen als Wertebereiche der Attribute verwendet werden. Ein Dokumententyp definiert demnach eine Liste von Attributen bestimmter Domänen, die die Struktur jedes Dokuments genau vorgeben.

3. Die aus dem objektorientierten Datenmodell bekannte Vererbung ist ein gewinnbringender Mechanismus, um bestehende Typen leicht erweitern und neue erstellen zu können. Dies trägt der zentralen Forderung nach Unterstützung dynamischer Typen Rechnung. Obwohl grundsätzlich möglich, ist die Mehrfachvererbung, d.h. die Definition neuer Typen auf der Basis mehrerer anderer bestehender Typen, nicht zwingend erforderlich, da die Komplexität des Modells durch die benötigten Auflösungsstrategie für die in diesem Falle möglichen Konflikte (doppelte oder inkompatible Definitionen in verschiedenen Supertypen) stark erhöht wird. Die Einfachvererbung muß allerdings unterstützt werden.

4. Die Referenzsemantik objektorientierter Modelle ist für monolithische Systeme bzw. für solche gut geeignet, die alle zu adressierenden Informationsartefakte in einem gemeinsamen Speicher (Datenbank, Hauptspeicher oder persistenter Objektspeicher) bereithält, und weist dann gegenüber der wertbasierten Assoziation erhebliche Vorteile auf (schnelle Navigierbarkeit, Integrität etc.). Diese Vorteile verschwinden, sobald heterogene Systeme mit diversen, unverbundenen Speichern integriert werden sollen, in denen und über deren Grenzen hinweg keine Referenzierung möglich ist. Hier bleibt die wertbasierte Identifikation als Ausweg. Ähnlich wie im relationalen Modell muß das Dokumentenmodell also wertbasierte Identifikation erlauben. Die Konsistenzsicherung bleibt dabei speziellen Subsystemen vorbehalten und ist im Systementwurf genauer zu definieren.

5. Die Dokumenttypen sollen wiederum als Dokumente modelliert werden. Dazu müssen sie einen speziellen, fest vom System vorgegebenen Dokumenttyp besitzen, der wiederum

natürlich als Dokument aufgefaßt werden kann. Durch diese Modellierung lassen sich generische Dienste, die die Verwaltung von Dokumenten zur Aufgabe haben, gleichfalls für die dynamische Erstellung und Veränderung von Typen verwenden.

Die soweit beschriebenen Eigenschaften des Dokumentenmodells für ein Wissensportal sind in Abbildung 5.5 illustriert. Alle im Rahmen der Analyse bestehender Systeme und Systemklassen sowie bei der Aufstellung der Anforderungen beschriebenen Informationsartefakte oder Objekte werden durch die Dokumente des definierten Modells repräsentiert. Um für die folgenden Diskussionen eine sprachlich unkompliziertere Abgrenzung zu anderen dort benutzten Begriffen erreichen zu können, definieren wir für den weiteren Gebrauch den Begriff des *Information Assets* (kurz *Asset* genannt) als durch Dokumente des eben definierten Modells repräsentierte Informationsartefakte aller Art. Dieser Begriff bringt die Tatsache zum Ausdruck, daß es sich bei dem beschriebenen Objekt um eine Einheit von bestimmtem (hohem) Wert, hier nämlich Information, handelt.

5.2 Prozeßmodelle für kooperative Informationssysteme

Neben dem Dokumentenmodell, das der produktorientierten Sichtweise Rechnung trägt, benötigt ein Wissensportalsystem auch ein Modell, das den prozeßorientierten Anforderungen Rechnung trägt. Dieses wird in diesem Abschnitt entwickelt.

Die zahlreichen von einem Wissensportal geforderten Eigenschaften zur Unterstützung von Kooperation und Kollaboration und die Integrationsfähigkeit in und von heterogenen Systemlandschaften lassen die Betrachtung des Portalsystems als kooperatives Informationssystem zu. Ein kooperatives Informationssystem wird dabei folgendermaßen definiert [DDJ+97]:

> *„The paradigm for the next generation of information systems will involve large numbers of information systems distributed over large, complex computer / communication networks. [...] Such information systems will manage or have access to large amounts of information and computing services. They will support individual or collaborative human work. Computation will be conducted concurrently over the network by software systems that range from conventional to advanced application systems including expert systems, and multiagent planning systems. Information and services will be available in many forms through legacy and new information repositories that support a host of information services. Communication among component systems will be done in a centralized or distributed fashion, using communication protocols that range from conventional ones to those based on distributed AI. We call such next generation information systems cooperative information systems.“*

Da die geforderte Funktionalität des Wissensportals trotz des web-basierten Ansatzes in vielen Punkten weit über die einfachen Anfrage/Antwort-orientierten Protokolle des WWW hinausgehen, werden die Erkenntnisse, die über die Modellierung von kooperativen Informationssystemen und insbesondere den darin ablaufenden Prozessen existieren, hier aufgegriffen und bilden später die Grundlage des Systementwurfs. Dazu wird im folgenden zunächst das Modell der *Business Conversations* (nebst wichtigen Teilen der Realisierung) näher vorgestellt, dessen Inhaltsmodell bereits in Abschnitt 5.1.4 aufgeführt wurde, sowie darauf aufbauend analog zum Dokumentenmodell ein Prozeßmodell für ein Wissensportal definiert.

5.2.1 Das Modell der Business Conversations

Die folgende Darstellung des Modells der *Business Conversations* orientiert sich stark an [Mat98, Mat97, Joh97, Ric97, Weg98, Rip98, Hup98, HMW98, MWH99, Hup00, WHM00], also den Arbeiten, in denen es entwickelt und präsentiert wurde. Eine ausführliche Darstellung findet sich insbesondere in [Weg98].

Das Leitbild des Modells basiert auf der fundamentalen Erkenntnis der Sprechakttheorie, daß die „Sprache nicht nur dazu dient, die Welt zu beschreiben“ [GHS91], sondern auch dazu, sie zu verändern, indem sie Tatsachen schafft. Diese Theorie wurde zunächst von AUSTIN entwickelt und später von SEARLE fortentwickelt [Aus62, Sea69]. Dabei wurden unterschiedliche Sprechakte identifiziert wie assertative, direktive, kommissive, expressive oder deklarative Akte. Auf diesem eher sprachtheoretisch orientierten Schema setzten später Arbeiten im Umfeld der *Computer Supported Cooperative Work* (CSCW) auf. Als für das Modell relevanter Ansatz ist an erster Stelle der *Language/Action*-Ansatz von WINOGRAD und FLORES zu nennen [WF86, Win87, FGHW88, Win88]. Hier sind zwei Arten von Sprechakten als wichtig erkannt worden: *Conversations for Possibilities* (CfP) und *Conversations for Actions* (CfA). Der zugrundeliegende Gedanke ist, daß genau zwei Partner im Rahmen einer Konversation jeweils wechselseitig eine Folge von Sprechakten vollführen, die aus diversen Aufforderungen und Versprechen zur Erreichung eines gemeinsamen Ziels bestehen.

Die CfA bildet die Basis des Kooperationsmodells der *Business Conversations*. Es werden hierzu in genau fixierter Reihenfolge von den beiden beteiligten Aktoren gemäß eines vorgegebenen Schemas performative Sprechakte vollzogen, die die Konversationen strukturieren und die Erreichung eines Ziels koordinieren. Mit der gemeinsamen Ausführung der Konversationsschritte entsteht eine Historie, vor deren Hintergrund die jeweils nächsten Akte erzeugt und interpretiert werden. Dieses Leitbild ermöglicht eine weitreichende Medien- und Aktorenunabhängigkeit dadurch, daß das Interaktionsmuster universell bezüglich der Art der Akteure (Software oder Menschen) und der von ihnen verwendeten Kommunikationsmedien und Protokolle ist. Insofern ist die Kommunikation mit menschlichen oder maschinellen Akteuren ununterscheidbar.

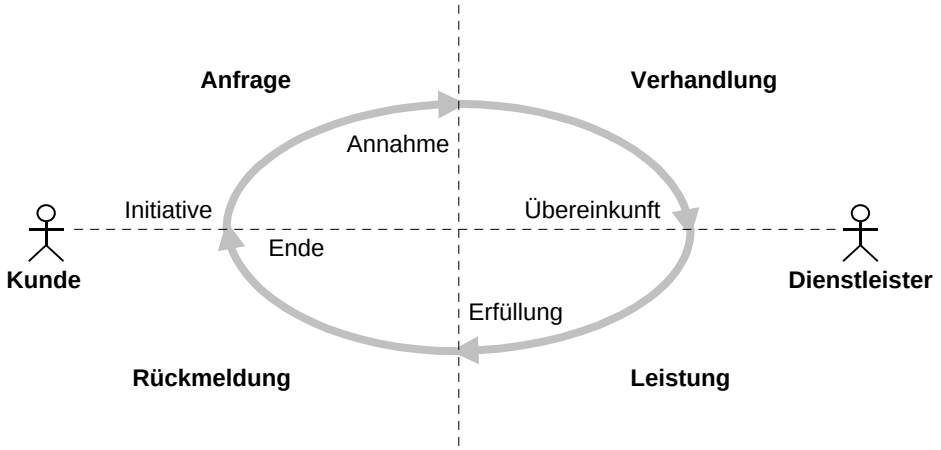

Abbildung 5.6: Interaktionsphasen des Modells der *Business Conversations*

Die Rollenverteilung im Modell der *Business Conversations* ist asymmetrisch konstruiert: Der eine Aktor agiert in der Rolle eines Kunden, der andere in der eines Dienstleisters. Die geführte Konversation untergliedert sich dabei nach dem in Abbildung 5.6 gezeigten Schema in vier aufeinanderfolgende Phasen:

1. In der *Anfragephase* eröffnet der Kunde durch Anfrage beim Dienstleister die Konversation. Die Anfrage ist unverbindlich und kann entweder mit der Annahme der Konversation enden oder aber der Ablehnung. Die Initiative kann nur vom Kunden ausgehen.

2. In der anschließenden *Übereinkunftsphase* erfolgt die Verhandlung über die Modalitäten der zu erbringenden Dienstleistung, die entweder mit der Übereinkunft oder dem Scheitern enden kann.

3. Die *Leistungsphase* umfaßt die Erbringung der eigentlichen Leistung durch den Dienstleister unter entsprechender Mitwirkung des Kunden. Sie endet mit der Erfüllung der zuvor ausgehandelten Leistung.

4. Schließlich dient die *Rückmeldungsphase* dem Kunden dazu, seine Zufriedenheit mit der erbrachten Leistung bezüglich seiner Anforderungen und Erwartungen zum Ausdruck zu bringen. Mit ihr endet zugleich die gesamte Konversation.

Der Abschluß aller vier Phasen bedeutet das Erreichen des gemeinsamen Ziels. Der vierphasige Zyklus kann durch Weglassen von Phasen (z.B. der Rückmeldungsphase) oder Zusammenfassung (etwa von Anfrage und Verhandlung) vereinfacht werden, sofern die Beziehung der Akteure dies erlaubt. Zudem ist es jederzeit möglich, die Konversation abzubrechen. Neben technischen Ausfällen können Übereinkunft und Erbringung der Leistung scheitern und so die vorzeitige Beendigung der Konversation erzwingen.

Das universale Interaktionsmuster gestattet die uniforme Modellierung verschiedenster Arten von Akteuren wie ganzer Organisationen, Teilbereichen daraus, Behörden, Einzelpersonen oder einzelner Informationssysteme. Dabei lassen sich die zahlreichen gleichzeitig zwischen allen vorhandenen Akteuren stattfindenden Konversationen in ein umfangreiches Geflecht binärer Kunde/Dienstleister-Beziehungen zerlegen, in denen jeweils pro Konversation jeder Akteur in einer festen, nicht veränderlichen Rolle verbleibt. Zusammen mit der Medien- und Aktorenunabhängigkeit gestattet das zusätzliche Vorhandensein formaler Spezifikationen der möglichen Konversationen darüber hinaus, Konversationen in mehreren verschiedenen Modalitäten zu führen:

- **Anwendungskopplung** bedeutet, daß beide Partner einer Konversation maschinelle Akteure sind, also Softwaresysteme. Dies ermöglicht die synchrone oder asynchrone Kopplung der Anwendungen.

- **Benutzungsschnittstellen** können zur Interaktion von menschlichen Akteuren mit maschinellen genutzt werden. Das Vorhandensein einer formalen Spezifikation der erlaubten Konversationen gestattet die Interpretation derselben durch ein die Kundenrolle einnehmendes Softwaresystem (auch „generischer Kunde“ genannt), das dann daraus dynamisch eine Benutzungsoberfläche erstellen kann und seinerseits als Dienstleister gegenüber den menschlichen Akteuren auftritt. Dies hält die eigentliche Konversation universell, da der primäre mit dem generischen Kunden kooperierende Dienstleister nicht zu erkennen vermag und braucht, ob der Partner maschineller oder menschlicher Natur ist. Wesentlich ist, daß der Mensch als Initiator der Konversation auftritt, was ja grundsätzlich der Kundenrolle vorbehalten ist.

- **Workflow-Management** kann durch die komplementäre Interpretation des obigen Falles unterstützt werden: Hier tritt ein Softwaresystem als Kunde auf, das Anfragen (etwa nach zu erledigenden Aufgaben) an einen menschlichen Dienstleister sendet. Dazu muß es in Analogie zum generischen Kunden einen Softwareagenten als generischen Dienstleister in Anspruch nehmen.

- **Kollaboration** zwischen menschlichen Akteuren bedeutet in diesem Kontext, daß die formal beschreibbaren Konversationen zwischen Menschen entweder manuell oder von einem unterstützenden Softwaresystem überwacht werden, um die Arbeitsstrukturierung und die Einhaltung bestimmter Regeln und Vorgehensweisen sicherzustellen. Dieser Fall kann durch Auflösung einer binären Konversation in mehrere kaskadierte Beziehungen der obigen beiden Arten abgebildet werden.

Durch diese losen, autonomieerhaltenden Kopplungsmöglichkeiten ist die Integration komplexer bestehender Infrastrukturen und Prozesse möglich. Dazu kann die Zerlegung nach außen geschlossen erscheinender Dienste in weitere Kunde/Dienstleister-Beziehungen verwendet werden: Der über eine primäre Konversation angesprochene Dienstleister kann in der Funktion eines Vermittlers mehrere weitere sekundäre Konversationen mit dritten Dienstleistern aufnehmen, in denen er als Kunde agiert. Gleichfalls kann ein einzelner Dienstleister in der Funktion eines Koordinators mehrere Konversationen mit Kunden gleichzeitig führen, und als Kunde eine einzelne Konversation mit einem dritten Dienstleister führen, um den Ablauf aller primärer Konversationen zu koordinieren. Die sekundären Konversationen können dabei je nach Aufgabe synchron oder asynchron zu den primären geführt werden.

Die bereits angesprochene formale Spezifikation von Konversationen wird durch eine strikte Trennung der zwei Ebenen der Konversationsspezifikation und der konkreten Konversationen erreicht. Eine Konversationsspezifikation legt die Struktur und den Ablauf von Konversationen fest; sie entspricht der Typebene von Programmiersprachen. Konversationsexemplare sind die konkreten, zwischen zwei Aktoren ablaufenden Konversationen. Jede Konversation besitzt als Typ genau eine Konversationsspezifikation. Eine Konversationsspezifikation gliedert die Konversation in einzelne Dialogschritte. In jedem Schritt wird vom Dienstleister ein sogenannter Dialog, der als Dokument oder Formular betrachtet werden kann, zusammen mit einer Menge von möglichen Anfragen an den Kunden gesendet, der daraufhin den Inhalt des Dialogs verändert und ihn unter Angabe genau einer der möglichen Anfragen an den Dienstleister zurückgibt. Dieser generiert in Abhängigkeit von der gewählten Anfrage und dem veränderten Dialoginhalt einen Folgedialog und beginnt den Zyklus erneut. Dies geschieht solange, bis die Konversation von beiden Seiten als beendet betrachtet wird. Formal lassen sich diese Transitionen folgendermaßen als Funktionen notieren:

$$\begin{array}{rrcl} \text{Dienstleister:} & (d'_n, r_n) & \mapsto & d_{n+1} \in r_n \\ \text{Kunde:} & d_n & \mapsto & (d'_n, r_n) \end{array}$$

Die d_i stellen dabei die Dialoge und die r_i die Anfragen dar, wobei i der Index des jeweiligen Dialogschritts ist. Eine Anfrage r_i besteht formal aus der Menge der möglichen Folgedialoge d_{i+1}. Die Spezifikation der möglichen Dialogfolgen läßt sich als nichtdeterministischer endlicher Automat (NFA) und der Ablauf der Konversation als Interpretation bzw. Simulation desselben betrachten, wobei die Dialoge die Zustände bilden und die Anfragen die Übergänge markieren. Der Nichtdeterminismus kommt in der Möglichkeit mehrerer verschiedener Folgedialoge für eine Anfrage zum Ausdruck (siehe Abbildung 5.7). Die Entscheidung der Zustandsübergänge erfolgt verteilt: die Auswahl der Anfrage obliegt dabei dem Kunden, während die Auswahl des daraufhin gewählten Folgedialogs dem Dienstleister zukommt.

Eine Konversationsspezifikation definiert also neben ihrem Namen

- alle in der Konversation möglichen Dialoge, die durch ihren Namen und eine strukturierte Beschreibung ihres Inhalts durch ein Inhaltsmodell (siehe Abschnitt 5.1.4) näher definiert werden,

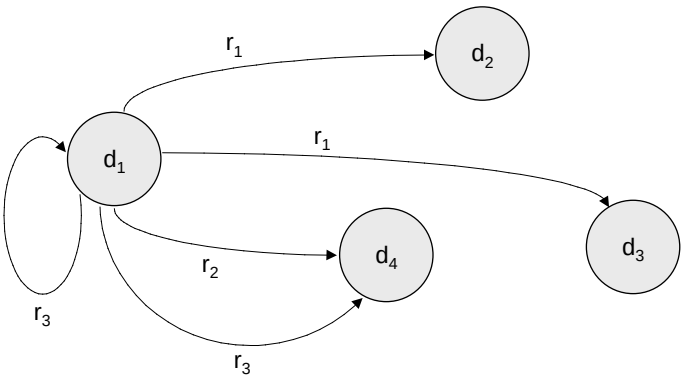

Abbildung 5.7: Konversationsspezifikation als nichtdeterministischer endlicher Automat

- eine Menge von Anfragen für jeden Dialog, die aus jeweils einem Namen und der Menge der möglichen zugelassenen Folgedialoge besteht, und
- einen ausgezeichneten Dialog als initialen Dialog, mit dem jede Konversation beginnt. Diesen versendet der Dienstleister auf die initiierende Anfrage eines Kunden als erstes.

Daneben existieren als weitere Dialoge mit besonderer Bedeutung die finalen Dialoge. Dies sind Dialoge, die keine Anfragen besitzen und somit keine weiteren Folgedialoge; ihr Erreichen innerhalb einer Konversation beendet die Konversation. In Begriffen der Automatentheorie handelt es sich hierbei um die Endzustände des Automaten. Das Inhaltsmodell der Dialoge besteht aus einem einfachen Typsystem, das den rekursiven Aufbau komplexer Aggregate ermöglicht. Es wurde bereits im Vorgriff in Abschnitt 5.1.4 beschrieben. Auch für das Inhaltsmodell gilt, daß strikt zwischen Spezifikation und Exemplar unterschieden wird.

Ein Aktor kann in diesem Modell gleichzeitig mehrere Konversationen mit anderen Aktoren führen, wobei seine Rolle (Kunde oder Dienstleister) pro Konversation fixiert ist. Ein Aktor besitzt also unter Umständen eine Reihe von gleichzeitig aktiven Rollen in unterschiedlichen Konversationen. Dabei ist jede Konversation unveränderlich fest an eine Konversationsspezifikation gebunden, die bei der Eröffnung zwischen Kunde und Dienstleister ausgehandelt wird. Dies geschieht mit der initialen Anfrage des Kunden.

Der softwaretechnische Entwurf und die *Tycoon Business Conversations* (TBC) genannte Realisierung, die in [Joh97] und [Weg98] durchgeführt wurden, sehen die dynamische Konfiguration einer Rolle zur Laufzeit durch die Konversationsspezifikation vor. Um dies zu ermöglichen, wurde der Ansatz gewählt, alle Spezifikationen als Objekte erster Ordnung in der Implementierungssprache zu erzeugen und nicht als zu Klassen modellieren, deren Exemplare die Konversationen sind. So sind sowohl Spezifikationen als auch Exemplare gleichberechtigt Objekte erster Ordnung, die sich dynamisch zur Laufzeit erzeugen, inspizieren, manipulieren und wieder zerstören lassen. Damit wurde einerseits die dynamische Konfigurierbarkeit von Aktoren und Rollen gewährleistet, und andererseits die Voraussetzung für generische und reflektive Dienste sowie den Bau und die Nutzung von Meta-Diensten geschaffen. Das TBC-System wurde als Rahmenwerk (*Framework*) konzipiert, so daß an speziellen Stellen die aufgabenspezifische Anwendungslogik konkreter Anwendungen eingehängt werden kann. Dazu wurde ein regelbasierter Ansatz gewählt, bei dem die Rollenobjekte (Kunde bzw. Dienstleister) zur Laufzeit an rollenspezifische Sätze von Regeln gebunden werden, die letztlich genau die oben genannten Transitionen der Zustandsmaschine implementieren. Die gebundenen Regel-Objekte müssen dafür Exemplare von Subklassen bestimmter rollenspezifischer Regeln sein und Operationen mit bestimmten Signatu-

ren implementieren. Alle bisher beschriebenen Modellkonzepte sind in Abbildung 5.8 in einem Klassendiagramm illustriert.

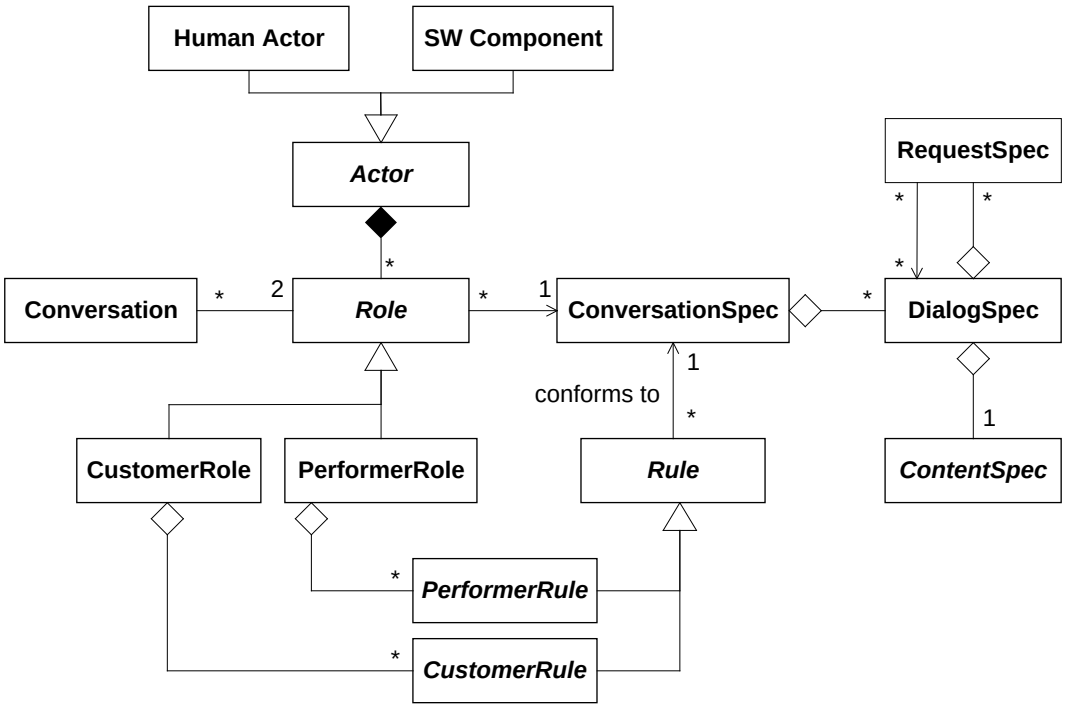

Abbildung 5.8: Modellkonzepte der *Business Conversations*

Die Sicherstellung der Konsistenz der an die Rollen gebundenen Regelsätze stellt natürlich ein Problem dar, da die Regeln – obwohl statisch typisiert – erst dynamisch zur Laufzeit gebunden werden. Es kann so beispielsweise nicht garantiert werden, daß alle benötigten Regeln existieren und – als schwerer wiegendes Problem – daß sie korrekten Gebrauch von Dialog-Inhalten des durch sie behandelten Dialogschritts machen. Der in [Hup98] vorgestellte Ansatz eines dynamischen Spezifikationsprüfers, der unter Verwendung der Konversationsspezifikation die gebundenen Regeln auf korrekte Verwendung der Inhalte und der Konversationshistorie überprüft, stellt einen Ausweg dar. Er erfordert allerdings das Vorhandensein umfangreicher Möglichkeiten zur Reflektion in der verwendeten Programmiersprache und machte sich die direkte Ansprechbarkeit von Typprüfer und Compiler in TYCOON-2 zunutze.

Das Modell der *Business Conversations* wurde als Grundlage verschiedener Systeme verwendet, um Internet-Informationssysteme zu realisieren [WHM00, MWH99]. In [Rip98] wurde die prototypische Realisierung eines Hotelreservierungs-Systems beschrieben, das Gebrauch vom TBC-System macht. Dazu wurde ein generischer Kunden-Softwareagent entwickelt, der beliebige Konversationsspezifikationen in web-basierte Benutzungsoberflächen zu konvertieren vermag und so als Web-Server agierend die oben beschriebene Benutzungsschnittstelle zu komplexen Systemen bereitstellt. In [Stö99] wurde ein Intranet-basiertes Informationssystem zur Erfassung und Verfolgung von Fehlermeldungen in einem Softwareunternehmen konzipiert und entwickelt, das schwerpunktmäßig die Workflow-Eigenschaften und Koordination der Aktivitäten unterstützt. In [Car99] wurde eine Reimplementierung des Modells mit dem LOTUS NOTES DOMINO-Server und einer Anbindung von SAP R/3-Systemen beschrieben, um ein generisches *Online*-Verkaufssystem zu realisieren.

5.2.2 Ein Prozeßmodell für ein Wissensportal

Die in Kapitel 4 aufgestellten Anforderungen an das Wissensportal-System umfassen Funktionen, die in vielen Punkten weit über die einfachen Anfrage/Antwort-orientierten Protokolle des WWW hinausgehen. Hierzu zählen insbesondere asynchrone Funktionen wie die Unterstützung von Workflow, Wissenszustellung, Notifikationsdienste und Gemeinschaftsfunktionen, aber auch intern benötigte Kopplungsmöglichkeiten an bestehende Informationssysteme, deren Informationsbestände in das Portalsystem integriert werden sollen.

Von dem eben beschriebenen Modell der *Business Conversations* können als Prozeßmodell des Wissensportals einige Eigenschaften übernommen werden, während andere nicht sinnvoll umzusetzen sind:

- Der grundsätzliche Interaktionszyklus des BC-Modells ist gut geeignet, um alle in sämtlichen Anforderungen vorkommenden Interaktionen zwischen Benutzern und dem Portalsystem zu beschreiben. Die Sichtweise der Akteure als Kunde und Dienstleister ist für alle Szenarien zutreffend. Während beispielsweise in allen Interaktionen, die über die webbasierte Oberfläche abgewickelt werden, der Benutzer vorteilhafterweise als Kunde modelliert wird, ist dies bei aktiven Diensten des Portals wie etwa Notifikation gerade andersherum: das System tritt als Kunde auf, der sich aktiv an den Benutzer wendet.

- Die Unterstützung der oben genannten Interaktions-Modalitäten wie Benutzungsschnittstelle, Workflow etc. läßt sich damit gut erreichen. Allerdings wird die im Modell mögliche Zerlegung komplexer Systeme (etwa durch Vermittler und Koordinatoren) für ein einzelnes System nicht erforderlich sein.

- Das Inhaltsmodell, das bereits in Abschnitt 5.1.4 vorgestellt wurde, fand dort auch bereits Eingang in das Dokumentenmodell des Wissensportals.

- Strikte Konversationsspezifikationen für sämtliche möglichen Navigationspfade (Klickpfade) in einem web-basierten Portal zu erstellen und zu verwenden ist wenig sinnvoll, da aufgrund der zu erwartenden hohen – teilweise auch dynamischen – Vernetzung zum einem die Aussagekraft einer Vernetzung, die jeden Dialog mit nahezu jedem anderen verbindet, gering ist, und zum anderen durch die hohe Dynamik des Portals (Personalisierung, Suchergebnisse) ebenfalls ad hoc für den einmaligen Gebrauch erstellte Spezifikationen keinen Vorteil bieten. Konversationsspezifikationen zu erstellen und zu verwenden hat dennoch für einige ausgewählte Funktionen einen Vorteil, nämlich für die Definition und die Abwicklung von allen schematischen Prozessen und Workflows (vgl. die Anforderungen in Abschnitt 4.3.9). Die Modellierung aller Spezifikationsobjekte (Konversationsspezifikationen, Dialogspezifikationen, Anfragespezifikationen und Inhaltsspezifikationen) als *Asset* ist dabei eine wichtige Anforderung.

- Die Sichtweise des Portalsystems als Aktor, der in verschiedenen Rollen diverse Konversationen führt, paßt gut zu den geschilderten aktiven und passiven Aufgaben. Während hier einige Rollen wie die des normalen Web-Servers ohne Konversationsspezifikation auskommen, müssen beispielsweise die aktiven Kundenrollen zur Ausführung von Prozessen oder zur Notifikation solche verwenden.

- Der für die Realisierung der TBC gewählte regelbasierte Ansatz verspricht auch für einen Portalserver mit den genannten Anforderungen ein sinnvolles Konzept zu sein, da sich zum einem eine klare Trennung der softwaretechnischen Verantwortlichkeiten erzielen läßt und zum anderen die einfache Erweiterbarkeit um neue Funktionalität als eine wichtige Anforderung erreicht wird.

Zusammen mit dem Dokumentenmodell bildet dieses Prozeßmodell die abstrakte Grundlage des im nächsten Kapitel vorgestellten Entwurfs eines Wissensportals. Weitergehende Ansätze zur Beschreibung der Prozesse in einem Portal werden in [ZHMS01] beschrieben.

Kapitel 6

Objektorientierter Entwurf eines Wissensportalsystems

There are two ways of constructing a software design:
One way is to make it so simple that there are obviously no deficiencies
and the other way is to make it so complicated that there are no obvious deficiencies.
– C.A.R. HOARE

In diesem Kapitel wird der objektorientierte Entwurf eines Wissensportals basierend auf den Anforderungen aus Kapitel 4 und den Modellen aus Kapitel 5 beschrieben. Nach einer Diskussion und Begründung der wichtigsten Entwurfsentscheidungen in Abschnitt 6.1 wird zunächst die Systemarchitektur in Abschnitt 6.2 vorgestellt und motiviert. Daran schließen sich in den Abschnitten 6.3, 6.4, 6.5 und 6.6 die Darstellungen der Strukturentwürfe und Verhaltensentwürfe der Komponenten der verschiedenen Schichten des Gesamtsystems an.

Das hier angewandte objektorientierte Vorgehensmodell (etwa nach [JBR99, JCJÖ92]) sieht eine Trennung von Analyse- und Entwurfsphase vor. Die Analyse kann durch die in den vorigen Kapiteln vorgenommenen Untersuchungen als abgedeckt betrachtet werden. Die Ergebnisse des objektorientierten Entwurfs werden hier mit den modernen Notationsmitteln der *Unified Modeling Language* (UML) dargestellt [RJB99, BRJ99, FS99, MW00].

6.1 Entwurfsentscheidungen

Zu Beginn eines softwaretechnischen Entwurfs sind diverse grundlegende Entscheidungen zu treffen, die im wesentlichen die verwendeten Sprachen und Systeme sowie die Konstruktion des Gesamtsystems betreffen. Im folgenden werden diese Entscheidungen für das Wissensportalsystem dargestellt und begründet.

6.1.1 Konstruktion

Die wichtigsten Realisierungsentscheidungen sind die über die grundsätzliche Art des Baus des in Frage stehenden Systems. Es bieten sich als die Extreme des vorhandenen Lösungsspielraumes zwei verschiedene Vorgehensweisen an:

1. Die Konstruktion des gewünschten Systems aus möglichst vielen bereits existierenden – auch kommerziell angebotenen – Software-Komponenten. Das Ziel ist dabei, die Wiederverwendung zu maximieren und den Anteil der erforderlichen Neuentwicklungen von Softwarekomponenten zu minimieren. Die Integration der Komponenten wird dadurch zur Hauptaufgabe, daß die Komponenten aneinander angepaßt werden müssen. Die dahinter stehende Annahme ist die, daß die Wiederverwendung erprobter Komponenten durch die Nutzung standardisierter Schnittstellen erheblichen Entwicklungsaufwand einspart und im Verhältnis dazu nur geringen Anpassungsaufwand erzeugt.
2. Die Konstruktion des gewünschten System durch möglichst weitgehende Implementierung der Kernfunktionalität in einem neuen System. Dies umfaßt einen gründlichen und umfangreichen Entwurf und die entsprechende sorgsame Implementierung. Das Ziel ist hier, die Bruchlosigkeit des Entwurfs zu maximieren, dadurch die zugrundeliegenden Konzepte nahtlos umzusetzen und nur diejenigen bereits existierenden Komponenten hinzuzufügen, deren Integration weniger aufwendig als die Neuimplementierung ist. Die Annahme ist hier, daß ein gut fundierter Entwurf eine über weite Strecken einfache Implementierung erlaubt und die dadurch höheren Kosten durch Einsparungen von Anpassungsarbeiten sowie geringerem Aufwand für Wartung und Erweiterung von schlechter zueinander passende Komponentenschnittstellen mindestens aufgewogen werden.

Die Überlegenheit des einen oder des anderen Ansatzes läßt sich nicht grundsätzlich feststellen, da die Art des zu konstruierenden Systems einen starken Einfluß auf die potentiell entstehenden Entwicklungs- oder Anpassungskosten hat. Festzustellen ist, daß das Bestreben, komponentenorientierte Konstruktionen zu verwenden, stark ausgeprägt ist. Die Hoffnung, die Defizite der traditionellen Softwareentwicklung (Fehleranfälligkeit, unvorhergesehene Kostensteigerungen) dadurch zu beheben, motiviert seit geraumer Zeit die Forschung in dieser Richtung erheblich (siehe z.B. [Kre99]) und hat für eine weitgehende Akzeptanz der Komponententechnologie geführt [Gri98, Pre97].

Wir vertreten in dieser Arbeit die These, daß der zweiter Ansatz für die Konstruktion eines wie beschrieben beschaffenen Wissensportals geeigneter ist. Dies ist zugleich eine der zentralen Behauptungen dieses Kapitels. Zahlreiche empirische Beobachtungen untermauern neben den in Kapitel 7 gezeigten Anwendungen die Evidenz dieser Behauptung. Im folgenden werden diese systematisch näher erläutert.

- Zunächst ist der zur Verfügung stehende Lösungsspielraum bezüglich benötigter und zur Verfügung stehender Komponenten zu analysieren. Da es sich im weiteren Sinne bei dem Portal um ein Internet-Informationssystem handelt, sind primär folgende Aufgaben zu lösen, die an den Schichten der dafür üblicherweise eingesetzten n-Schichtenarchitektur orientiert sind [BG98, Tur99, Loe98, Wil99]. Diese Schichten sind in Abbildung 6.1 illustriert. Die neben der client-seitigen Präsentationsschicht auf dem Server befindlichen drei oder mehr Schichten haben folgende Aufgaben:

  1. Betrieb eines Web-Servers, also einer Komponente, die das HTTP-Protokoll implementiert und mit den Web-Browsern kommuniziert. Hier kommen auch Verteilungsmechanismen im Falle hoher Skalierbarkeit zur Lastverteilung zum Einsatz.
  2. Betrieb der eigentlichen Anwendungslogik des Portals. Dazu gehören zahlreiche Teilaufgaben, die sich aus den einzelnen Anforderungen ergeben. Die Verteilung dieser Aufgaben auf Subkomponenten ist dabei möglich. Zudem ist hier die Gliederung in mehrere Schichten, die sogenannten *middle tiers*, möglich.
  3. Sicherstellung der Persistenz aller langlebigen Nutz- und Betriebsdaten durch geeignete Speicherungsmechanismen.

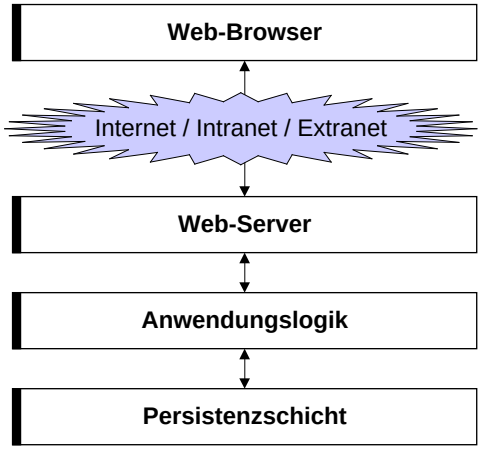

Abbildung 6.1: Typische n-Schichtenarchitektur von Internet-Informationssystemen

Die kritischen Stellen sind dabei in der Kopplung der drei Schichten zu sehen. Besonders problematisch ist die Verbindung vom Web-Server zur Anwendungslogik, da hier eine hohe Integrationsleistung erbracht werden muß: Bei der geforderten Trennung von Inhalt und Layout der Informationen wird es zur Aufgabe dieser Verbindung, beides zur Auslieferung – in dem die Trennung nicht oder nur unzureichend unterstützendem HTML – wieder zu vereinigen. Ansätze zur Einbettung von Funktionalität in die Gestaltungsvorlagen, die dies bewerkstelligen, gibt es einige; zu nennen sind insbesondere *Java Server Pages* (JSP), *Active Server Pages* (ASP) und der *PHP Hypertext Preprocessor* (PHP) [Sun01c, ASP01, PHP01]. Allen diesen Lösungen ist jedoch gemein, daß sie wiederum die Trennung von Anwendungslogik und Gestaltung unterlaufen. Die Anbindung der Persistenzschicht stellt häufig ebenfalls eine problematische Schnittstelle dar, da hier beispielsweise der oft beschriebene *Impedance Mismatch* zwischen Anwendungsmodell und Datenbankmodell zum Tragen kommt.

Denkbar ist auch, für spezielle Belange der eigentlichen Anwendungslogik auf bestehende Komponenten zurückzugreifen. Die Möglichkeiten dafür hängen stark von der geforderten Funktionalität ab.

- Die mittlerweile vorliegenden Erfahrungen mit den später in den Einsatzbeispielen (Kapitel 7) beschriebenen Systemen lassen sich gut anhand eines Beispiels charakterisieren: Zur Durchführung der Volltextindizierung wurde in verschiedenen prototypischen und produktiven Systemen als hinzugefügte Komponente die dazu vorhandene Fähigkeit der ORACLE 8I-Datenbank genutzt. Aufgrund zahlreicher verborgener Fehlerquellen, undokumentierter Eigenschaften und nicht nachvollziehbarer Fehler erforderte die Anpassung und Inbetriebnahme der Volltext-Funktionen des Systems einen um mehrere Größenordnungen höheren Aufwand als zuvor dafür eingeplant. Dies macht deutlich, daß der Aufwand zur Systemintegration leicht völlig falsch eingeschätzt werden kann.

- Die Konfigurierbarkeit von Standardkomponenten suggeriert, daß die einfache und schnelle Inbetriebnahme ohne tiefgehende Kenntnisse der intern verwendeten Arbeitsweise möglich ist. Dies ist zumindest das Versprechen der Komponententechnologie. Betrachtet man aber u.a. das eben aufgeführte Beispiel, so wird deutlich, daß sich die Situation auch ganz anders entwickeln kann, so daß zunächst unter hohem Aufwand die Komponente verstanden werden muß und dann erst angepaßt werden kann. Ob in einer solchen Situation der Einsatz einer Komponente wirklich günstiger ist, ist überaus fraglich.

> Zudem stellt sich das grundsätzliche Problem der hohen technischen Abhängigkeit der Komponenten untereinander gerade im Falle des Einsatzes von Komponenten unterschiedlicher Hersteller. Befriedigende – insbesondere praktisch einsetzbare – Antworten auf diese Fragen sind noch nicht auszumachen [Sch00c].

Neben diesen kritischen Hinweisen ist das zweite gewichtige Argument für die Begehung des zweiten Entwicklungsweges die komplexe Natur des zugrundeliegenden Dokumenten- und Prozeßmodells zusammen mit einigen speziellen Anforderungen. Die uniforme Modellierung der *Information Assets* und und ihrer komplexen Beziehungen untereinander durch zahlreiche bidirektionale Verknüpfungen und die geforderte orthogonale Anwendbarkeit generischer Funktionen zur Klassifikation und Suche erfordern eine äußerst hohe Integrationsdichte innerhalb des funktionalen Kerns des Portalserversystems. Diese enge Kopplung innerhalb einer heterogenen Komponentenstruktur zu erreichen stellt sich als überaus schwieriges Problem dar, das extrem hohen Integrationsaufwands bedarf, so daß der Aufwand, die Kernfunktionalität geschlossen neu zu entwerfen und zu implementieren, geringer erscheint. Dabei bleibt immer noch abzuwägen, welchen Umfang diese Kernfunktionalität haben soll und ab wo sich wieder die Nutzung bestehender Komponenten und Systeme lohnt.

Diese Probleme lassen sich insbesondere nicht durch die relativ häufig verwendete *Open Source*-Lösung aus der mittlerweile als „LAMP“ titulierten Kombination der Komponenten LINUX als Betriebssystem, APACHE als Web-Server, MYSQL als Datenbank und PHP als Programmiersprache bzw. System zur Integration von Anwendungslogik, Web-Server und Datenbank lösen [Apa01, MyS01, PHP01], da damit weder eine mehrschichtige Serverarchitektur (die etwa für die geforderte Integration nötig ist) noch die Trennung von Funktionalität und Gestaltung erreichbar ist. Daß grundsätzlich bestimmte Anwendungen, auch Portale mit CMS-Funktionalität, damit realisiert werden können, soll nicht bestritten werden; siehe etwa [SOF00].

Zusammengenommen rechtfertigen diese Gründe die Konstruktion des Kerns des Portalserversystems als eigenständiges, neu zu implementierendes System. Zum Kern zählen dabei die ersten beiden Schichten, also der Web-Server und die Anwendungslogik. Die Gründe dafür sind, daß

- der prinzipielle Aufwand zur Entwicklung eines Web-Servers aufgrund der Einfachheit des Protokolls HTTP nicht besonders hoch ist;
- aufgrund der eingeschränkten Anforderungen für dieses konkrete System längst nicht alle Fähigkeiten universell einsetzbarer Web-Server wie beispielsweise APACHE oder von generellen Anwendungsservern benötigt werden;
- nur so eine optimale (bruchlose und effiziente) Kopplung zwischen Web-Server und Anwendungslogik erreicht werden kann. Zudem kann dabei trotzdem die geforderte Trennung von Funktionalität, Gestaltungsvorlagen und Inhalten durchgehalten werden.

Nicht zum Kern zu zählen ist die darunter liegende Persistenzschicht, da die Realisierung leistungsfähiger Datenbanken, die sowohl die nötige Sicherheit als auch die Performanz für einen produktiven Einsatz bieten, weit außerhalb eines solchen Vorhabens liegen. Hier kommt es dann auf eine Anbindung an, die möglichst effizient und nahtlos ist.

Diese grundlegenden Entwurfsentscheidungen finden ihren direkten Niederschlag in der in Abschnitt 6.2 vorgestellten und diskutieren Architektur des Systems.

6.1.2 Programmiersprache und Plattform

Zwei weitere wesentliche Entwurfsentscheidungen sind die der verwendeten Programmiersprache und der unterstützten Plattformen, d.h., auf welchen Hardwarekonfigurationen und welchen Betriebssystemen das zu entwerfende System später betrieben werden können soll. Diese sind grundsätzlich nicht unabhängig voneinander zu treffen, da viele Entwicklungsumgebungen bestimmter Programmiersprachen nur eine eingeschränkte Menge von Plattformen unterstützen, also selbst auf ihnen benutzt werden können oder die Entwicklung für bestimmte Zielplattformen erlauben.

Für ein Client/Server-System stellt sich die Frage gleich doppelt, nämlich getrennt für Client und Server. Da es sich bei dem Wissensportal primär um ein web-basiertes System handelt, ist die Frage des Clients automatisch geklärt, da hier ein bereits vorhandener Web-Browser verwendet werden soll. Die Serverseite bleibt also als einziges von Belang. Entscheidend für eine Serveranwendung im kommerziellen Umfeld ist, daß möglichst viele dort benutzte Plattformen unterstützt werden. Die Wahl der Programmiersprache sollte sich daran messen lassen. Grundsätzlich ist die Wahl einer plattformunabhängigen Programmiersprache oder zumindest einer leicht portablen Sprache, die auf vielen Plattformen implementiert ist, vorteilhaft.

Als Implementierungssprache für das Wissensportal wird die plattformunabhängige Programmiersprache JAVA gewählt [GJS96], da diese einerseits auf vielen gängigen Plattformen (u.a. MICROSOFT WINDOWS, SUN SOLARIS etc.) verfügbar ist, und andererseits ein starker Trend besteht, derartige Systeme in JAVA zu realisieren. Dies bedeutet nämlich, daß für viele nicht selber zu realisierende Teil-Aufgaben bereits *Frameworks*, Klassenbibliotheken und Komponenten verfügbar sind, und für die Entwicklung selbst entsprechende Werkzeuge (Entwicklungsumgebungen etc.) bereitstehen. Die Wahl von JAVA entspricht somit dem aktuellen Stand der Entwicklung und vermeidet die Entwicklung eines auf veralteter Software aufsetzenden Systems.

6.2 Architektur

Nachdem alle wesentlichen Entwurfsentscheidungen bekannt sind, wird nun die Systemarchitektur des Wissensportalservers vorgestellt. Auf dieser Ebene ist im wesentlichen eine geschichtete Architektur des Gesamtsystems zu entwerfen, die den vielfältigen Anforderungen und den Entwurfsentscheidungen Rechnung trägt. Zum besseren Verständnis der gewählten Lösung werden die Kernanforderungen, die direkten Einfluß auf die Architektur haben, nochmals konzentriert wiedergegeben (vgl. dazu auch die Basiskonzepte in Abschnitt 4.2):

1. Enge Kopplung der klassischen Schichten eines Internet-Informationssystems, regelbasierte Anbindung der Anwendungslogik und medien- und aktorenunabhängiges Interaktionskonzept;
2. Uniformes, dynamisches und reflektives Modell der *Information Assets* mit bidirektionaler Verweisverwaltung und transparenter Integration verschiedener Informationssysteme und -quellen;
3. Nutzung der vier konzeptuellen Hauptmetaphern von Personen, Dingen, Orten und Begriffen;
4. Trennung von Struktur, Inhalt, Layout und Funktionalität.

Um diesen Anforderungen und Zielen gerecht zu werden, wurde die in Abbildung 6.2 dargestellte Architektur entworfen. Die Schichten, ihre Aufgaben und ihr Zusammenwirken werden

im folgenden überblicksartig erläutert und in den nachfolgenden Abschnitten dann detailliert diskutiert.

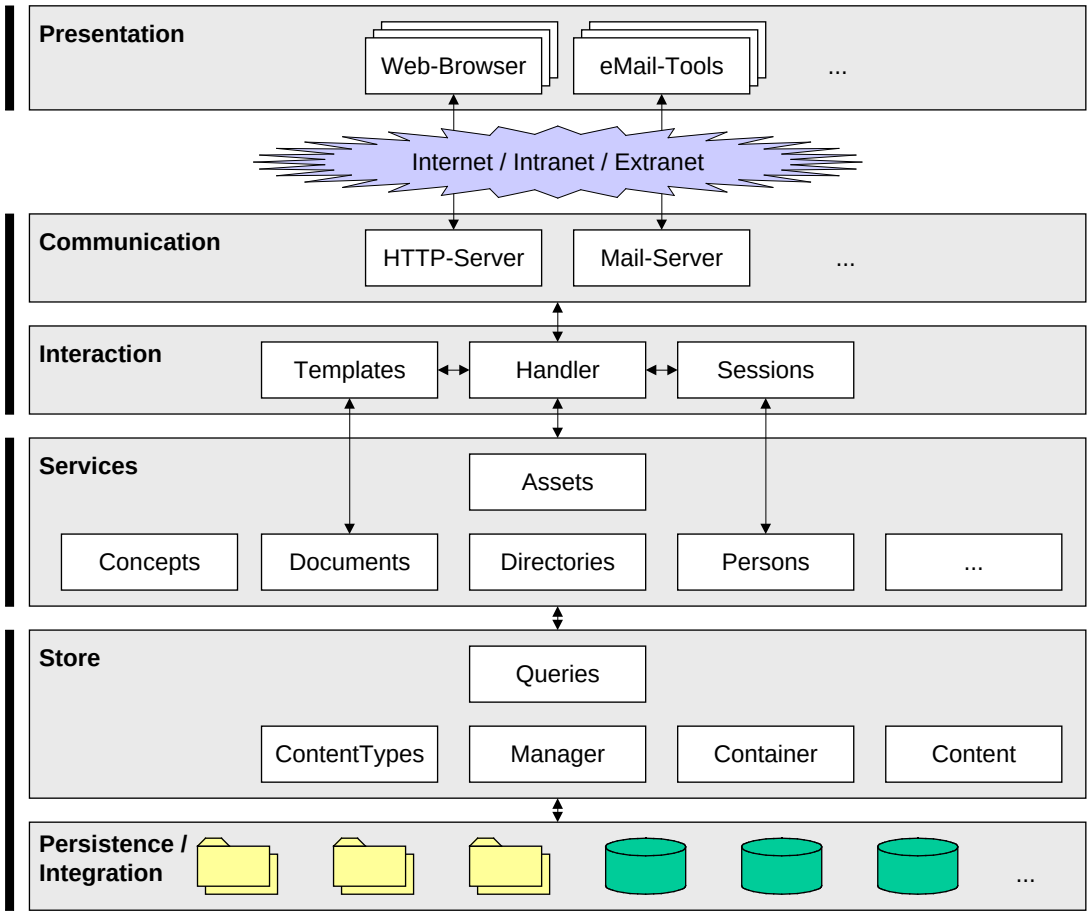

Abbildung 6.2: Entwurf der Schichtenarchitektur

Die klassische n-Schichtenarchitektur für Internet-Informationssysteme (vgl. Abbildung 6.1 sowie Abbildung 6.2) findet sich hier grundsätzlich wieder; allerdings sind einige dieser Schichten nochmals in mehrere Unterschichten gegliedert. Sie werden von oben nach unten erläutert:

1. **Presentation:** Dies ist die Präsentationsschicht (*Graphical User Interface*, GUI); wie bei Internet-basierten Systemen üblich wird als Benutzungsoberfläche primär ein Web-Browser und für bestimmte Aufgaben eine e-Mail-Anwendung benutzt. Auch spezielle mobile Geräte wie WAP-fähige Mobiltelefone oder *Personal Digital Assistents* (PDA) können unterstützt werden. Im Sinne der geforderten Medien- und Aktorenunabhängigkeit kann potentiell eine Vielzahl von Endgeräten bedient werden. Für die geforderten Portalfunktionen sind zunächst Web-Browser und e-Mail zwingend zu unterstützen. Sie kommunizieren über Internet-Protokolle (HTTP, SMTP, POP, FTP, WebDAV, SMB etc.) mit der Kommunikationsschicht.

2. **Communication:** Diese Schicht dient der Kommunikation mit der client-seitigen Präsentationsschicht und der Umsetzung der jeweiligen Protokolle und Interaktionsmuster auf ein generalisiertes Interaktionsschema der darunterliegenden Schicht. Hier sind insbesondere ein Web-Server zur Kommunikation über HTTP sowie ein e-Mail-Server zur Generierung

und zum Empfang von e-Mail erforderlich. Weiterhin können auf dieser Schicht bereits Maßnahmen zur Lastverteilung realisiert werden, falls dazu mehrere Instanzen der darunterliegenden Schichten existieren.

3. **Interaction:** Diese Schicht realisiert die medien- und aktorenunabhängige Abstraktion des Prozeß- und Interaktionsmodells. Der Kommunikationsschicht obliegt die Übersetzung der medienabhängigen Protokolle und Interaktionsformen auf ein generelles, auf dieser Schicht verstandenes Schema. Dazu existiert als grundlegender Mechanismus die Regel (*Handler*), die im Sinne des aus dem Modell der *Business Conversations* übernommenen regelbasierten Interaktionsansatzes Anfragen aller Art in den Rollen von Dienstleister oder Kunde bearbeitet bzw. stellt. Zur Generierung der Antworten auf Anfragen wird ein vorlagenbasierter Ansatz gewählt, der es unabhängig von der Funktionalität der Regeln erlaubt, Gestaltungsvorlagen (*Templates*) mit dem aktuell benötigten Inhalt zu füllen und über die Kommunikationsschicht auszuliefern. Dieses Konzept der *Templates* kommt der Forderung nach Trennung von Struktur, Inhalt, Layout und Funktionalität nach. In den Regeln wird lediglich die Umsetzung der Präsentation und Funktionalität der Dienste geleistet. Zudem wird in dieser Schicht eine Sitzungsverwaltung (*Sessions*) realisiert, die insbesondere von dem Web-Server der darüberliegenden Schicht benötigt wird, um die Mängel des zustandslosen Protokolls HTTP auszugleichen, die aber auch von den Regeln als Kontext benötigt wird. Die Gestaltungsvorlagen und Sitzungsinformationen werden teilweise auf die semantischen Objekte (*Assets*) der Dokumente und Personen der darunterliegenden Dienstschicht abgebildet.

4. **Services:** Diese Schicht enthält die eigentliche Anwendungslogik des Systems durch objektorientierte Realisierungen der *Assets* der semantischen Objekte wie Begriffen, Dokumenten, Verzeichnissen, Personen und Verknüpfungen sowie aller weiteren Klassen. Diese Dienste werden von allen Regeln der darüberliegenden Schicht in Anspruch genommen. Zur Sicherstellung der Persistenz der erzeugten *Assets* wird transparent für die darüberliegenden Schichten die nächst tiefere Schicht in Anspruch genommen.

5. **Store:** Diese Schicht stellt eine einheitliche Abstraktionsebene dar, die einerseits die transparente Nutzung verschiedenartiger und verschiedener Persistenzmechanismen wie Datenbanken oder Dateisysteme erlaubt, und andererseits zur Integration bestehender Anwendungen wie Content-Management-Systeme, betrieblicher Informationssysteme oder sonstiger beliebiger Systeme dienen kann. Jede konkrete Anbindung basiert auf einer *Manager* genannten Komponente, die Container und Content-Objekte verwaltet.

6. **Persistence/Integration:** Diese letzte Schicht enthält die konkreten, von der darüberliegenden *Store*-Schicht verwendeten Persistenzdienste und integrierten Systeme, die üblicherweise von Drittherstellern stammen. Hier sind relationale oder objektorientierte Datenbanken, Dateisysteme, Content-Management- und Dokumenten-Management-Systeme, bestehende Verzeichnisdienste (LDAP, NIS etc.), *News*-Server, externe e-Mail-Server, externe Web-Server, *News-Ticker* und anderes mehr denkbar.

In den folgenden Abschnitten werden diese Schichten von unten nach oben detailliert beschrieben. Diese *bottom-up* Reihenfolge wurde gewählt, um das Verständnis jeder Schicht zu erleichtern, da dann jeweils die von ihr benutzte direkt darunterliegende Schicht bekannt ist. Die Darstellung der obersten und der untersten Schicht (Präsentation und Persistenz) erübrigt sich, da es sich um vorhandene, hier nicht entworfene Systeme handelt. Einzelheiten der Präsentationsschicht, d.h. der Oberfläche, sind ferner in den Einsatzbeispielen in Kapitel 7 zu finden.

6.3 Die Speicherungsschicht

Reality is merely an illusion, albeit a very persistent one.
– ALBERT EINSTEIN

Die Speicherungsschicht (auch *Store*-Schicht genannt) dient der darüberliegenden Dienstschicht zur Sicherstellung der Persistenz und zur Integration von Informationsquellen. Die wesentlichen Entwurfsziele in dieser Schicht sind:

- Bereitstellung eines Inhalts- bzw. Dokumentenmodells, das von den konkreten Modellen der verschiedenartigen darunter verwendeten Systemen abstrahiert, sich aber dennoch möglichst einfach auf deren Konzepte abbilden läßt und die einfache Abbildung der *Assets* auf das Modell zulässt. Der *Impedance Mismatch* zwischen den Schichten soll durch diese Abstraktion stark gemildert werden.
- Dazu muß eine generische Schnittstelle angeboten werden, über die unterschiedliche Systeme einheitlich angesprochen werden können. Der Austausch eines konkret verwendeten Systems muß möglich sein, ohne daß die benutzenden Schichten einen Unterschied bemerken. Die Schnittstelle muß Operationen zum Erzeugen, Ändern, Löschen und zum Suchen der verwalteten Objekte anbieten.
- Bereitstellung von effizienten Realisierungen zur Anbindung von relationalen Datenbanken und Dateisystemen. Diese Minimalimplementierungen sind zumindest für den Betrieb des Gesamtsystems nötig.

Eine ähnlich Diskussion wird in [Zül98, S. 465 ff.] bezüglich der effizienten Abbildung objektorientierter Anwendungskonzepte auf konventionelle (relationale) Systeme geführt.

6.3.1 Inhaltsmodell der Speicherungsschicht

Das Inhaltsmodell gliedert ein zu integrierendes System in drei Ebenen: Auf der obersten Ebene wird ein System als ganzes durch einen Manager repräsentiert, der eine Menge von persistenten Containern verwaltet, die in einem hierarchischen Namensraum angesiedelt sind. Die Container dienen der Strukturierung der enthaltenen Informationen. Jeder Container aggregiert dazu eine – eventuell leere – Menge von persistenten Inhaltsobjekten. Die Inhaltsobjekte gehorchen dabei einem Inhaltstyp, der ihre Struktur durch eine Reihe von flachen Attributen verschiedener atomarer Datentypen beschreibt. Bezüglich der Typisierung sind dabei zwei Arten von Containern zu unterscheiden:

- **Homogene Container** erzwingen, daß ausschließlich Inhaltsobjekte desselben Typs von ihnen aggregiert werden. Die Erzeugung von Inhaltsobjekten unterschiedlichen Typs ist nicht möglich. Der Inhaltstyp ist praktisch fest an den Container gebunden.
- **Inhomogene Container** erlauben die Speicherung von Inhaltsobjekten unterschiedlichen Typs. Jeder beliebige Typ von Inhaltsobjekt kann in dem Container erzeugt werden. Der Container besitzt dadurch praktisch keinen Typ; diese sind an die verschiedenen Inhaltsobjekte gebunden.

Ein Manager eines konkreten Speicherungssystem muß nicht zwingend sowohl homogene als auch inhomogene Container anbieten; das Angebot inhomogener Container bleibt optional. Homogene Container müssen immer unterstützt werden.

Jedes Inhaltsobjekt hat stets einen festen Inhaltstyp. Sobald ein Inhaltsobjekt erzeugt worden ist, ist sein Typ unveränderlich. Das bedeutet, daß auch in inhomogenen Containern keine Attribute zu bestehenden Inhaltsobjekten hinzugefügt werden können. Das Typsystem der Inhaltstypen ist als flache, geordnete Aggregation von Attributen aufgebaut. Der Typ als ganzes trägt neben seinem Namen keine weiteren Eigenschaften. Jedes Attribut wird durch seinen Namen, einen atomaren Basisdatentyp sowie weitere vom Datentyp abhängige Eigenschaften definiert. Folgende Basisdatentypen existieren:

- **Wahrheitswert:** Der Wert kann nur „wahr“ oder „falsch“ sein (**`true`** und **`false`**).
- **Ganzzahl:** Ganze Zahlen mit dem durch 32 Bit im Zweierkomplement darstellbaren Wertebereich (`Integer` in JAVA).
- **Zeichenkette:** Zeichenketten mit bestimmter, fest durch den Typ vorgegebener Maximallänge. Eine größte Maximallänge von 2048 Zeichen darf dabei nicht überschritten werden. Die Einführung dieses Typs ist ein Zugeständnis an die Existenz relationaler Datenbanken, in denen ein solcher Typ effizient implementiert und überaus gebräuchlich ist. Da Zeichenketten in JAVA nicht längenbeschränkt sind, wirft dies natürlich Probleme auf, wenn Attributwerte gespeichert werden sollen, die zu lang sind. Hier ist dann insbesondere ein einheitliches Verhalten aller Implementierungen erforderlich.
- **Zeitstempel:** Dieser Datentyp integriert eine Datums- und Uhrzeitangabe, wie es auch die Klasse `Date` in JAVA tut. Er spezifiziert damit einen Zeitpunkt. Durch die in `Date` vorhandene Schnittstelle wird es relativ leicht ermöglicht, vom Datum oder der Uhrzeit abzusehen, wenn diese irrelevant ist. Konzeptuell schöner wäre die Auftrennung in zwei Typen Datum und Uhrzeit; da diese Trennung aber in JAVA nicht vorhanden ist, hätte es bei jeder Benutzung zahlreicher umständlicher Konvertierungsoperationen bedurft, so daß hier der pragmatische Weg gewählt wurde, der eine einfache Benutzung gestattet.
- **Text:** Dieser Datentyp enthält gleichfalls Zeichenketten, aber mit (praktisch) unbeschränkter Länge. Auch die Existenz dieses Typs ist ein Zugeständnis an Datenbanken, die neuerdings solche Datentypen als *Character Large OBject* (CLOB) anbieten. Abgesehen von der Längenbeschränkung verhält dieser Typ sich wie der der Zeichenkette.
- **Objekt:** Dieser Datentyp dient zur Speicherung von – aus seiner Sicht – unstrukturierten Bytefolgen (praktisch) beliebiger Länge. Es findet sein Pendant in den in Datenbanken vorhandenen *Binary Large OBjects* (BLOBs). Hiermit können beispielsweise Bilder, Multimediadateien oder ähnliches verwaltet werden.

Neben diesen benutzerdefinierten Attributen besitzt jedes Inhaltsobjekt zwei immer vorhandene und automatisch verwaltete Metaattribute:

1. **Identifikator:** Eine innerhalb des Containers eindeutige Identifikation (ID) in Form einer längenbeschränkten Zeichenkette. Diese dient zur wertbasierten Identifikation des Inhaltsobjektes und ist nach dem Erzeugen nicht mehr änderbar.
2. **Zeitstempel:** Der Zeitpunkt der letzten Veränderung des Inhaltsobjektes. Dieses Attribut ist vom oben genannten Typ des Zeitstempels.

Die Attributtypen Zeichenkette, Zeitstempel, Text und Objekt haben als erlaubten und voreingestellten Wert den **`null`**-Wert, die Ganzzahl den numerischen Wert 0 und Wahrheitswerte **`false`**. Dies entspricht genau den Initialisierungskonventionen von JAVA. Die Werte der zuerst genannten Attributtypen können jederzeit wieder auf **`null`** gesetzt werden.

Die bisher genannten Konzepte sind in Abbildung 6.3 in einem Klassendiagramm illustriert. Die Inhaltstypen werden durch Exemplare der Klasse `ContentType` repräsentiert, die Exemplare von konkreten Subklassen von `Attribute` aggregieren. Die drei abstrakten Klassen `Manager`, `Container` und `Content` werden durch Subklassen für konkrete Systeme realisiert. Die möglichen Eigenschaften der Attribute sind hier als UML-Attribute der Subklassen von `Attribute` dargestellt. Insbesondere läßt sich für alle „kleinen“ Datentypen der Wunsch nach Indexierung zum schnellen Zugriff ausdrücken. Die Zeichenkettentypen lassen sich ferner volltextindexieren.

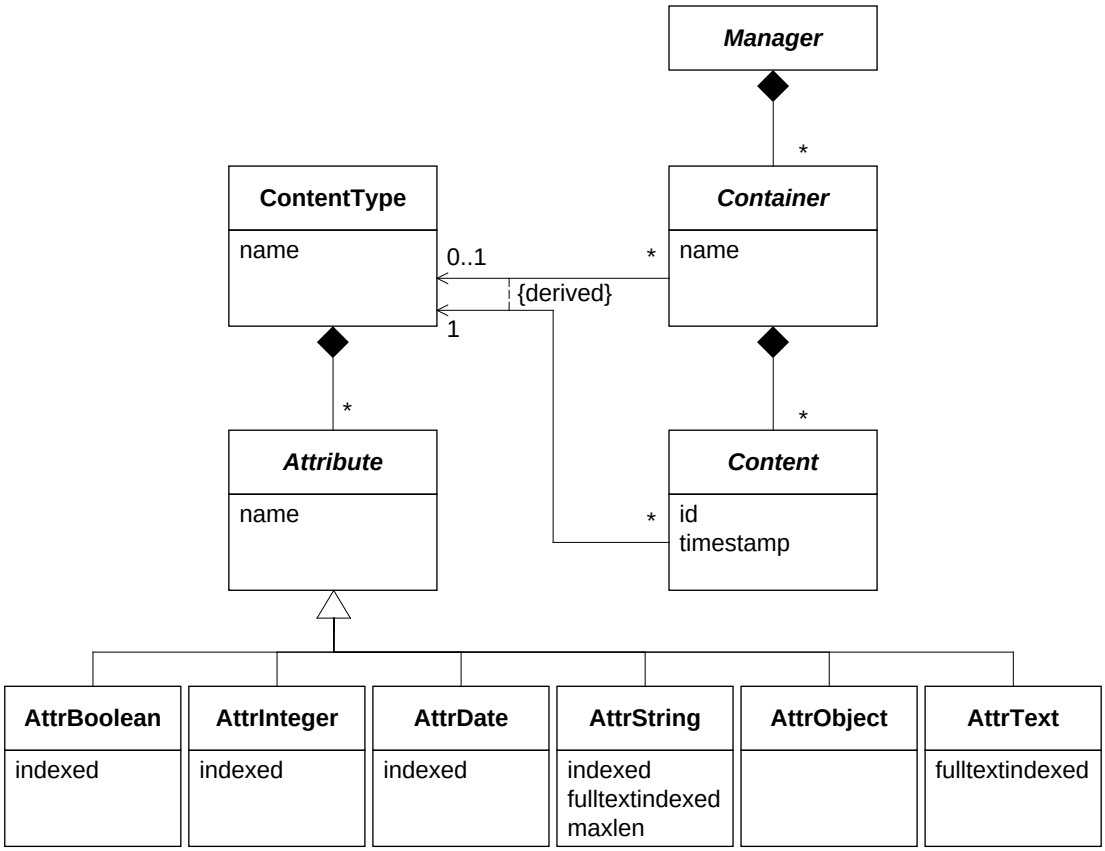

Abbildung 6.3: Inhaltsmodell der Speicherungsschicht

Dieses Inhaltsmodell mit einer flachen Attributstruktur ist bewußt als möglichst enge Annäherung an das relationale Datenmodell (vgl. Abschnitt 5.1.1) konstruiert, da die Nutzung relationaler Datenbanken immer eines der wahrscheinlichsten Nutzungsszenarien sein wird. Die gute Passung ermöglicht so eine effiziente Umsetzung des Inhaltsmodells auf das relationale Modell. Aber auch die Konzepte anderer Systeme wie Dateisysteme, objektorientierter Datenbanken oder von Verzeichnisdiensten lassen sich gut abbilden. Auf der anderen Seite entspricht das Inhaltsmodell bereits dem darüberliegenden Modell der *Information Assets* weitgehend, so daß hier eine effiziente Übersetzung möglich bleibt.

Auf die auf den ersten Blick naheliegende Einführung spezieller (referentieller) Assoziationstypen in das Inhaltsmodell wurde bewußt verzichtet. Naheliegend ist die Einführung zwar, da beispielsweise eine Abbildung auf die wertbasierte Assoziation im relationalen Modell dank der Identifikatoren aller Inhaltsobjekte einfach und unter gleichzeitiger Sicherstellung der referentiellen Integrität möglich wäre, aber da auf der darüberliegenden Schicht der Dienste die Inhalte mehrerer Manager transparent integriert werden können müssen (etwa für die Klassifikation der

Inhalte heterogener Informationsquellen), scheidet diese Lösung aus. Der Aufbau von Assoziationen wird stattdessen entweder durch explizites Nachbauen der Primär- und Fremdschlüsselbeziehungen des relationalen Modells im Inhaltsmodell gelöst, oder aber durch Dienste auf der höheren Schicht.

6.3.2 Schnittstellen der Speicherungsschicht

Die Basisidee des Entwurfs des Inhaltsmodells ist, daß jedes zu verwendende System von einem Exemplar des Typs `Manager` verwaltet wird. Dies ist eine abstrakte Schnittstelle, die durch ein JAVA-Interface spezifiziert wird. Für jede zu verwendende Art von System wird ein eigener Manager als Implementierung der Schnittstelle realisiert. Mehrere Manager – auch, aber nicht ausschließlich derselben Art – müssen gleichzeitig verwendbar sein.

Die Verantwortlichkeiten sind folgendermaßen organisiert und drücken sich durch die Schnittstellen aus: Der Manager (bzw. seine Exemplare im objektorientierten Sinne) ist für die Identifikation, Aufzählung (als Iterationsabstraktion), Erzeugung und Zerstörung der einzelnen Container zuständig. Jeder Container ist ebenso für die Identifikation, Aufzählung und Suche, Erzeugung und Zerstörung der enthaltenen Inhaltsobjekte verantwortlich. Jedes Inhaltsobjekt bietet hauptsächlich Operation zur Abfrage und Manipulation der Attributwerte an.

Ein Auszug der Schnittstelle des Managers mit den wichtigsten Methoden (in UML-Notation der Methodensignaturen) lautet:

```
getContainer(name:String) :Container
defineContainer(name:String, type:ContentType)
createContainer(name:String, type:ContentType) :Container
createContainer(name:String) :Container
deleteContainer(name:String)
getContainers() :Iterator<Container>
getContainerNames() :Iterator<String>
allContainersAreHomogeneous() :Boolean
close()
```

Die Identifikation von Containern geschieht grundsätzlich über ihren Namen. Das hierarchische Namensschema wird über qualifizierte Namen analog zu Pfadnamen in Dateisystemen oder Paketnamen in JAVA ausgedrückt. Ein Container könnte also z.B. `STS/Persons/Staff` heißen. Die Identifikation der Container geschieht ausschließlich über diese Namen; ein Container hat also keine Kenntnis über logisch in ihm enthaltene Subcontainer.[1] Diese – beispielsweise vom klassischen *Composite*-Muster nach [GHJV95] abweichende – Konstruktion wurde so gewählt, um zum einen die Verantwortlichkeiten für die Containerverwaltung auf den Manager zu konzentrieren, und zum anderen erleichtert es die Realisierung von Managern, die diese hierarchischen Strukturen intern auf flache abbilden müssen (etwa im Falle von relationalen Datenbanken). Die beiden überladenen `createContainer()`-Operationen dienen zum Anlegen von homogenen bzw. inhomogenen Containern; im zweiten Fall entfällt so gerade der Typ. Ein Manager, der dies nicht unterstützt, muß darauf mit Laufzeitfehlern reagieren.

Die Operation `defineContainer()` dient zur Schemaevolution in homogenen Containern. In realen Einsatzszenarien, besonders aber während der Weiterentwicklung des Systems kommen ständig neue Anforderungen, revidierte Ansichten und entdeckte Fehler und Unzulänglichkeiten

[1] Dies entspricht interessanterweise genau den Importregeln für Pakete und den daraus resultierenden Sichtbarkeiten von Klassen in JAVA.

zum Tragen, die die einst definierten Inhaltstypen verändern. Da in der Regel selbst bei Testsystemen in der Entwicklung das Löschen aller vorhandenen Daten und Neuinstantiieren der Inhalte nicht akzeptabel ist, müssen die vorhandenen persistenten Inhaltsobjekte dynamisch an veränderte Inhaltstypen anpaßbar sein. Dies leistet die genannte Operation, die die Zusicherung trifft, daß nach dem erfolgreichen Aufruf der Manager einen Container des angegebenen Namens mit dem angegebenen homogenen Inhaltstyp besitzt, unabhängig davon, ob und mit welchem Typ der Container zuvor existierte. Gleichzeitig werden eventuell existierende Inhaltsobjekte nach speziellen Kompatibilitätsregeln umgeformt. Da selbstverständlich Typredefinitionen möglich sind, die zu nicht umformbaren Inhaltsobjekten führen, kann dieser Aufruf mit einer Ausnahme scheitern. Zur Feststellung von Typänderungen muß der Begriff der Typgleichheit definiert werden:

> **Definition**: Die Typgleichheit zweier Inhaltstypen ist genau dann gegeben, wenn:
>
> - beide Inhaltstypen den gleichen Namen haben,
> - sie die gleiche Anzahl von Attributen besitzen und
> - alle Attributtypen paarweise typgleich in Bezug auf die definierte lineare Ordnung sind.
>
> Zwei Attributtypen sind typgleich, wenn sie Exemplare der gleichen Klasse sind, den gleichen Namen tragen und ihre Eigenschaften gleich sind.

Die Schemaevolution funktioniert wie folgt: Wenn der bereits bestehende und der neu angegebene Typ als nicht gleich erkannt wurden, werden folgende Schritte mit dem Ziel durchgeführt, die Gleichheit zu erreichen:

1. Wenn ein Attribut gegenüber der alten Definition des Inhaltstyps neu hinzugekommen ist, so wird es an der richtigen Position hinzugefügt. Alle existierenden Inhaltsobjekte erhalten als Attributwert den voreingestellten Wert (**`null`**, 0 oder **`false`**).
2. Wenn ein Attribut in dem neuen Typ gegenüber dem alten fehlt, so wird es auch im alten entfernt. Die entsprechenden Attributwerte aller existierenden Inhaltsobjekte werden entfernt.
3. Wenn sich lediglich der Typ eines Attributs verändert (dabei werden die Attribute über ihren Namen und ihre Position einander zugeordnet), wird zunächst überprüft, ob es sich um eine legale Typänderung handelt. Wenn nicht, wird eine Ausnahme ausgelöst. Legale Typänderungen sind genau:
   (a) Die Änderung der Eigenschaft `indexed` eines Attributtyps.
   (b) Die Änderung der Maximallänge (`maxlen`) eines Zeichenkettenattributtyps auf einen höheren Wert als zuvor.
   (c) Die Änderung des primitiven Datentyps eines Attributtyps. Dabei sind nur bestimmte Änderungen zulässig:
      - von Wahrheitswert nach Ganzzahl,
      - von Ganzzahl nach Zeichenkette,
      - von Zeitstempel nach Zeichenkette,
      - von Zeichenkette nach Text,
      - von Text nach Objekt, und
      - durch iterative Anwendung der obigen Regeln erreichbare Änderungen.

Nachdem eine legale Typänderung eines Attributs entdeckt wurde, werden die Attributwerte aller existierender Inhaltsobjekte des Containers so transformiert, daß sie dem neuen Typ entsprechen. Die oben definierten Regeln wurden gerade so gewählt, daß die Transformation stets ohne Fehler und ohne Informationsverlust möglich ist.

Das Umbenennen von Attributen kann mit diesem Verfahren nicht entdeckt werden. Stattdessen äußert sich eine Umbenennung als aufeinanderfolgende Löschung und Einfügung von zwei Attributen. Dabei gehen alle alten Attributwerte der existierenden Inhaltsobjekte verloren.

Ähnlich wie der Manager für Container ist der Container für die Verwaltung der Inhaltsobjekte (`Content`) verantwortlich. Ein Auszug seiner Schnittstelle lautet folgendermaßen:

```
getContentType() :ContentType
getContent(id:String) :Content
createContent(id:String, type:ContentType) :Content
createContent(id:String) :Content
deleteContent(id:String)
getContentIds() :Iterator<String>
getContents() :Iterator<Content>
queryContent(query:Query) :Iterator<Content>
queryContentIds(query:Query) :Iterator<Id>
isHomogeneous() :Boolean
```

Nur im Falle homogener Container liefert `getContentType()` einen Inhaltstyp zurück und analog zum Erzeugen von Containern dient die überladene Operation `createContent()` zum Anlegen von Inhaltsobjekten in homogenen bzw. inhomogenen Containern. Es lassen sich außerdem über Iteratoren alle Identifikatoren und alle Inhaltsobjekte auslesen.

Die Suche nach bestimmten Inhaltsobjekten ist über die Operationen `queryContent()` etc. möglich. Die Selektionskriterien werden dabei über ein Objekt des Typs `Query` übermittelt. Um eine hohe Ausdrucksmächtigkeit der Selektionsprädikate zu erreichen, wurde ein komplexes Modell zum Formulieren von geschachtelten konjunktiven und disjunktiven Termen entworfen, das stark an die Möglichkeiten der Auswahlklauseln in SQL angelehnt ist. Anders als dort wird die Klausel aber nicht textuell übermittelt, sondern als explizit konstruierter Ableitungsbaum des Ausdrucks. Um diesen adäquat repräsentieren zu können, wurde die in Abbildung 6.4 gezeigte Klassenhierarchie entworfen. Von der abstrakten Basisklasse `Query` ausgehend sind als Subklassen Operatoren (Konjunktion, Disjunktion, Negation) und Bedingungen, die Bezug auf ein Attribut nehmen, definiert. Durch die rekursive Aggregation von weiteren Anfrageobjekten durch die Operatoren lassen sich so beliebige logische Ausdrücke konstruieren. Die Bedingungen geben neben ihrer Klasse, die die Vergleichsoperation definiert, einen Wert und das dagegen zu prüfende Attribut an. Statt des Wertes kann ein weiteres Attribut angegeben werden, so daß – der relationalen Terminologie folgend – multirelationale Anfragen auch über Containergrenzen (allerdings desselben Managers) hinweg möglich werden.

Eine besondere Bedeutung hat die Bedingung `QueryContains`, über die Volltextsuchen spezifiziert werden können. Hier gibt es wiederum logische Operatoren zum Aufbau komplexer Bedingungen, aber nur eine Bedingung, in der ein zu suchendes Textliteral angegeben werden kann. Die Klassen zur Spezifikation der Volltextsuche wurden so konstruiert, daß eine Vermengung mit den übrigen Operatoren und Bedingungen nicht möglich ist.

Die Konstruktion einer Suchbedingung geschieht durch Erzeugung eines Graphen von Objekten der gezeigten Klassen. Dieses Konstrukt dient ausschließlich der Spezifikation der Anfrage; in den Klassen ist keinerlei Funktionalität zur Auswertung vorhanden. Diese obliegt alleine den

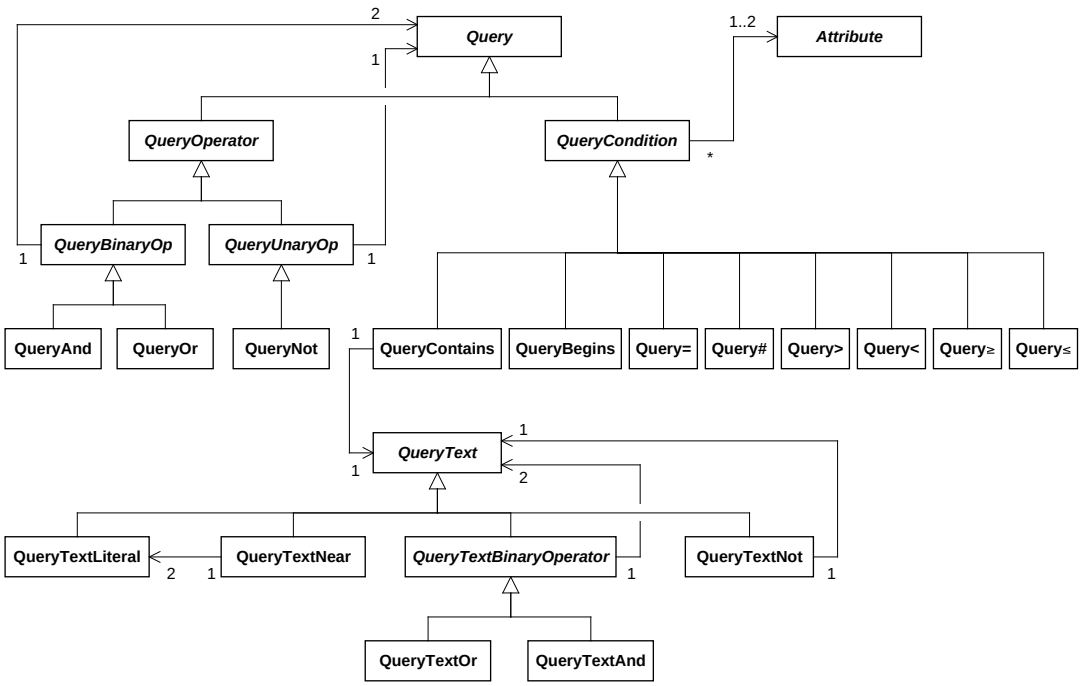

Abbildung 6.4: Klassenhierarchie zum Formulieren von Suchbedingungen

Containern. Zur einfacheren Behandlung der Anfragen in den Container-Implementierungen ist in der Struktur das Besucher-Muster nach [GHJV95] verankert und es sind abstrakte Besucher-Klassen definiert (nicht in der Abbildung gezeigt).

Schließlich folgt nun ein Ausschnitt der Schnittstelle der Inhaltsobjekte (`Content`).

```
getId() :String
getTimestamp() :Date
touch()
write()
getString(attributeName:String) :String
setString(attributeName:String, value :String)
getInteger(attributeName:String) :Integer
setInteger(attributeName:String, value :Integer)
...
```

Neben Operationen zum Abfragen der Metaattribute (`getId()`, `getTimestamp()`) gibt es eine Operation zum Sichern der geänderten Attributwerte. Da das typische Zugriffsverhalten häufig die Änderung einer größeren Anzahl von Attributwerten auf einmal bedingt, und um die Leistung des Systems zu erhalten, ist es bei vielen Systemen – etwa Datenbanken – unpraktikabel, bei jeder einzelnen Änderung eines Attributwertes die Änderung sofort persistent zu machen. Die Schnittstelle garantiert also das Überleben der Änderungen nach dem Aufruf von `write()`, stellt aber nicht sicher, daß Änderungen ohne diesen Aufruf überdauern.

Die Ansprache und Änderung der Attributwerte geschieht über Paare von `get()`- und `set()`-Operationen (Selektoren und Modifikatoren), die für jeden Basisdatentyp existieren, um Typsicherheit in JAVA zu erreichen. Zum Abfragen und Modifizieren eines Attributwertes muß dessen Name angegeben werden. Dies stellt natürlich eine Schwachstelle dar, da hier keinerlei stati-

sche Typsicherheit mehr zu erbringen ist. Diese Situation ist aber grundsätzlich bei dynamischen Typsystemen wie dem hier benutzten Inhaltsmodell unvermeidbar (siehe etwa [Hup98, Weg98]). Eine wichtige Entwurfsfrage ist, wie mit Fehlersituationen umgegangen werden soll, wenn ein ungültiger Attributname angegeben wird oder ein Selektor bzw. Modifikator eines falschen Typs aufgerufen wird. Drei Möglichkeiten sind denkbar, die kurz diskutiert werden:

1. Ignorieren des Fehlers, indem die `get()`-Operation den Nullwert zurückliefert und die `set()`-Operation nichts tut. Dies ist die unbefriedigendste Lösung, da massive Folgefehler induziert werden können, das Systemverhalten unvorhersagbar wird und vor allen Dingen die Fehlerentdeckung und -behebung praktisch undurchführbar wird.

2. Auslösen einer in der Schnittstelle spezifizierten Ausnahme (`Exception`). Dies ist prinzipiell eine gute Lösung und ist konform zur üblichen Methodik der Fehlerbehandlung in JAVA-Programmen. Die Lösung besitzt allerdings den pragmatischen Nachteil, daß die Dichte der Aufrufe der `get()/set()`-Operationen an bestimmten Stellen (etwa den Regeln) außerordentlich hoch ist und so ein hoher Aufwand zur Fehlerbehandlung entsteht. Die Erfahrung in konkreten Projekten hat sogar gezeigt, daß manche Programmierer aus Scheu vor dem Aufwand deshalb einfach dazu übergehen, innerhalb eines größeren Sichtbarkeitsbereiches, etwa einer ganzen Methode, diese Ausnahmen pauschal zu behandeln, indem sie sie ignorieren. Damit wäre praktisch die erste Lösung wieder erreicht.

3. Auslösen einer Laufzeitausnahme, die in JAVA nicht in der Schnittstelle spezifiziert werden muß. Diese auf den ersten Blick etwas unelegantere Lösung besitzt den Vorteil, daß die aufwendige und wie oben gesehen teilweise kontraproduktive Behandlung entfallen kann, aber die Fehlerentdeckung gut möglich bleibt. Es ist natürlich auf relativ hoher Ebene sicherzustellen, daß übersehene Fehler die Stabilität des Gesamtsystems nicht beeinträchtigen. Ein Laufzeitfehler paßt methodisch gut in das Sprachkonzept von JAVA, da dynamische Typfehler etc. in JAVA ebenfalls durch Laufzeitfehler signalisiert werden. Eine strengere und korrektere Sichtweise ist die, daß ja eigentlich keine normale Ausnahmesituation, also ein (dynamischer) Programmfehler, signalisiert wird, sondern ein (statischer) Programmierfehler.

Für die Schnittstelle und die Implementierungen der Inhaltsobjekte wurde nach den Erfahrungen mit der zweiten Lösung nachträglich die dritte gewählt. Auf die Möglichkeit, bei lediglich falschem Selektortyp eine Typumwandlung vorzunehmen (z.B. die Umwandlung eines Datumswertes in eine Zeichenkette) wurde bewußt verzichtet, um die möglichst strenge Beachtung und von vornherein gute Modellierung der Inhaltsbeschreibungen zu erzwingen. Lediglich die Selektoren und Modifikatoren für Zeichenketten und Texte lassen sich für beide Basisdatentypen verwenden, da beide in JAVA gleichermaßen durch `Strings` repräsentiert werden.

6.3.3 Realisierungen der Speicherungsschicht

Das Modell und die Schnittstelle der Speicherungsschicht werden durch die bereits beschriebenen JAVA-Schnittstellen spezifiziert. Konkrete Implementierungen müssen jeweils ein Tripel von Klassen bereitstellen, die Manager, Container und Content implementieren. Da der Zugriff und die Erreichung der Exemplare immer vom Manager ausgeht und über Container hinunter zum Content-Objekt delegiert wird und keine Operationen existieren, die als Argument Container oder Content-Objekte erwarten, ist sichergestellt, daß die Exemplare verschiedener Implementierungen nicht vermischbar sind. Es existieren bereits mehrere Realisierungen der Schnittstellen, die kurz beschrieben werden:[2]

[2] Diese Realisierungen wurden größtenteils vom Autor im Rahmen der in Kapitel 7 beschriebenen Projekte geschaffen.

- Eine Anbindung an relationale Datenbanken. Dabei werden die Container auf Tabellen und die Inhaltsobjekte auf Tupel der Tabellen abgebildet. Deswegen stellt dieser Manager stets homogene Container bereit. Die Anbindung der Datenbanken geschieht über die weit verbreitete und von vielen Herstellern durch Treiber unterstütze Schnittstelle JDBC [Sun01d].

  Um den Besonderheiten verschiedener relationaler Datenbanksysteme und insbesondere den Unterschieden in den verwendeten SQL-Datentypen und SQL-Dialekten Rechnung tragen zu können, wurden zunächst drei abstrakte Basisimplementierungen für Manager, Container und Content gebaut, von denen dann Subklassen für konkrete Datenbanksysteme gebildet werden können, die lediglich wenige Methoden implementieren müssen. Diese handhaben dann die Datentypen und spezielle Syntaxformen von SQL. Zur Zeit existieren Realisierungen zur Anbindung der Datenbanken ORACLE 8I und MYSQL [Ora01, MyS01].

  Die Basisimplementierung löst außerdem wichtige Probleme wie das *Caching* bereits einmal angeforderter Inhaltsobjekte. Dazu werden konfigurierbare LRU-Puffer (*last recently used*) und schwache Objektreferenzen von JAVA verwendet. Schließlich werden die Attributwerte der Basisdatentypen Text und Objekt gesondert behandelt und nur bei Bedarf (Abfrage des Attributwerts) überhaupt geladen. Dies verspricht hohe Einsparungen bezüglich der Zugriffszeit und des verwendeten Hauptspeicherplatzes, da das typische Zugriffsverhalten der darüberliegenden Schichten diese Attribute wesentlich seltener anspricht als andere, etwa im Falle von Listendarstellungen, wo nur wenige, meistens durch kurze Zeichenkettenattribute abgebildete Informationen angezeigt werden.

- Eine Anbindung von Dateisystemen. Hier werden die Verzeichnisse des Dateisystems als Container abgebildet und die einzelnen Dateien als Inhaltsobjekte. Der Manager wird durch die Angabe eines Wurzelverzeichnisses parametrisiert. Die Intention dieser Komponente ist die Integration der Inhalte bestehender Dateisysteme, nicht aber die Abbildung des Inhaltsmodells auf ein Dateisystem (wofür eine weitere getrennte Realisierung existieren könnte). Deshalb besitzen alle Container dieses Managers einen festen, eingebauten und unveränderlichen Typ, der aus lediglich einem Textattribut besteht, das zusätzlich auch als Binärattribut angesprochen werden kann. Dieses Attribut enthält genau den Inhalt der Datei. Der Dateiname ist der Identifikator des Inhaltsobjekts.

- Eine hauptspeicherbasierte Datenbank, die ihren gesamten Inhalt in einer einzelnen Datei persistent sichert. Der zugehörige Manager lädt bei seiner Instantiierung den gesamten Datenbankinhalt in den Hauptspeicher, wo er bis zum Schließen des Managers verbleibt. Beim Schließen wird der modifizierte Inhalt zurück in die ursprüngliche Datei gesichert. Diese Realisierung ist natürlich weder für den Produktivbetrieb noch für größere Datenmengen geeignet; aufgrund der identischen Arbeitsweise verglichen mit der Datenbank-Realisierung eignet sie sich aber hervorragend zum Testen und während des Entwicklungsbetriebs, oder aber für Situationen, in denen keine Datenbank verfügbar ist, beispielsweise für Präsentationen mit einem mobilen Rechner. Zudem ist sie aufgrund ihrer Realisierung wesentlich schneller als eine Datenbank.

  Ähnlich wie bei der Datenbankrealisierung wurde zunächst eine abstrakte Basisimplementierung geschaffen, deren Subklassen lediglich die Lade- und Speicherungsstrategie implementieren müssen. Zur Zeit sind Realisierungen vorhanden, die die Objektserialisierung von JAVA nutzen oder alternativ Textdateien. Letztere bieten den Vorteil, in Fehlersituationen während der Entwicklung die Inhalte extrem einfach manipulieren zu können.

- Schließlich existiert eine Realisierung, die die Integration des kommerziell verfügbaren Content-Management-Systems COREMEDIA gestattet [Cor01]. Die in diesem System vor-

handenen Strukturierungsmittel der Verzeichnisse und Dokumente [Cor00] werden auf Container und Content abgebildet. Die Anbindung geschieht über CORBA [Obj98].

In Abbildung 6.5 sind die wichtigsten Klassen für die Realisierungen der Speicherungsschicht dargestellt. Die Schnittstellen werden durch *Interfaces* beschrieben. Es existiert außer den gerade beschriebenen Realisierungen eine Zwischenschicht abstrakter Klassen (`AbstractManager`, `AbstractContainer`, `AbstractContent`), die für sämtliche Implementierungen wichtige Aufgaben herausfaktorisiert. Dazu zählen beispielsweise die automatische Vergabe von neuen, noch unbenutzten Identifikatoren und insbesondere eine Basisimplementierung des oben beschriebenen Schemaevolutionsalgorithmus.

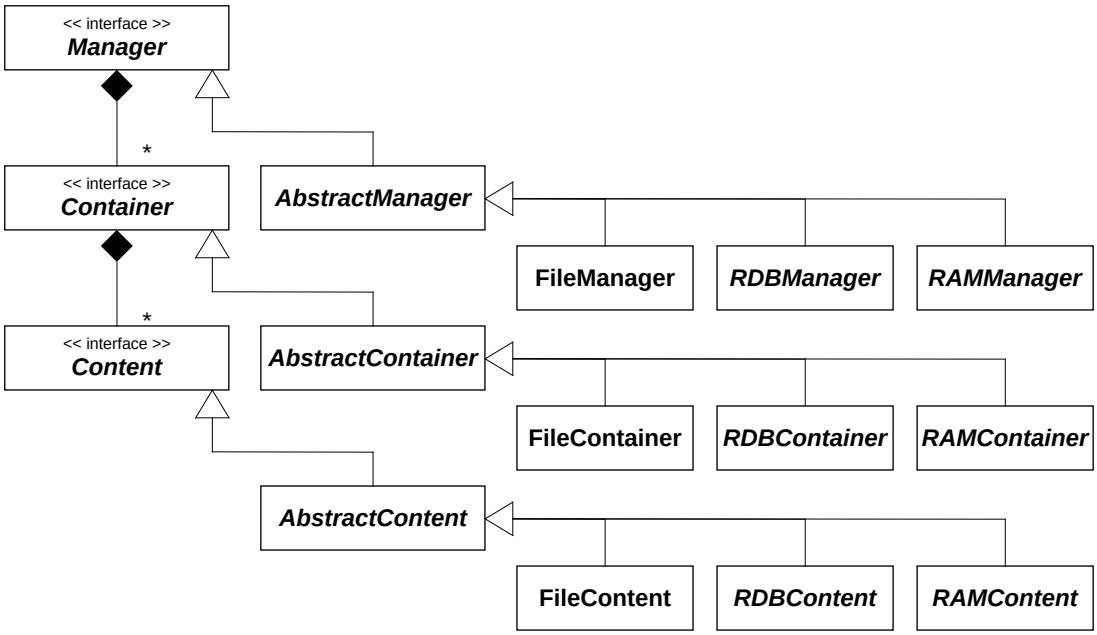

Abbildung 6.5: Realisierungen der Speicherungsschicht

Von größter Wichtigkeit für das Gesamtsystem ist insbesondere die effiziente Umsetzung der Anfrageschnittstellen. Die Realisierungen dieser Schicht müssen dafür geeignet sein, Massendaten effizient handhaben zu können. Alle Schnittstellen sind daher so ausgelegt, daß sie die Ergebnismengen ausschließlich über Iterationsabstraktionen zurückliefern. Dies gestattet eine speichereffiziente Implementierung, die nie alle Elemente einer Ergebnismenge zugleich instantiieren muß.

Abschließend sollen – ohne Anspruch auf Vollständigkeit – einige Realisierungen aufgezählt werden, die zur Zeit nicht existieren, aber deren Verwirklichung in größeren Einsatzszenarien und zur Erfüllung der fachlichen Anforderungen benötigt werden könnten:

- Eine Anbindung an relationale Datenbanken, die inhomogene Container unterstützt. Der jetzige Entwurf der darüberliegenden Dienstschicht ist nicht zwingend auf die Existenz einer solchen Komponente angewiesen; sie könnte aber zur Speicherung heterogener Kollektionen benutzerdefinierter Dokumententypen verwendet werden. Hier bietet sich dann eine erweiterte Spezifikation der Arbeitsweise der Suchbedingungen an. Als Vorbild könnte dabei das in [MS99, Büc99] beschriebene *Fuzzy Retrieval* dienen.

- Eine Anbindung von LDAP-Servern, NIS-Servern etc. zur Integration von Verzeichnisdiensten. Deren Strukturen lassen sich relativ einfach auf Container und Content abbilden. Hiermit wäre etwa der Anforderung Y.6 Rechnung getragen.
- Eine Anbindung von XML-Datenbanken wie TAMINO [Sof01, Tam99]. Es könnten dadurch sowohl inhomogene Container abgebildet werden, wobei die Recherchefunktionen von TAMINO gewinnbringend genutzt werden könnte, als auch bereits existierende XML-Daten integriert und erschlossen werden.

6.4 Die Dienstschicht

Die Dienstschicht (*Services Layer*) liegt direkt oberhalb der Speicherungsschicht und unterhalb der Interaktionsschicht. In dieser Schicht ist die eigentliche Anwendungslogik durch Realisierung der wichtigen konzeptuellen Klassen lokalisiert. Die wichtigsten Entwurfsziele dieser Schicht sind:

- Umsetzung des abstrakten Dokumentenmodells der *Information Assets* durch Entwurf und Implementierung einer entsprechenden Infrastruktur.
- Angebot einer reichhaltigen Schnittstelle in Form eines *Application Programming Interfaces* (API), die in erster Linie für die darüberliegende Interaktionsschicht bestimmt ist, aber auch als Schnittstelle für weitere externe, eventuell einzukoppelnde Dienste Verwendung finden kann.
- Bereitstellung einer transparenten und effizienten Abbildung der *Assets* auf die darunterliegende Speicherungsschicht. Diese ist im wesentlichen für den internen Gebrauch der in dieser Schicht bereitgestellten Dienste bestimmt.
- Erbringung der transparenten Integration aller verschiedener Informationsressourcen zum Zwecke der Suche, Klassifikation etc.
- Bereitstellung einer Architektur, die den Anforderungen an Skalierbarkeit und Sicherheit gerecht wird.

Im folgenden werden der generelle Aufbau der Dienstschicht, die Realisierung der Abbildung der angebotenen *Assets* auf die Speicherungsschicht, die entworfenen Konzepte zur Umsetzung von Verknüpfungen, die Anbindung eines Rahmenwerkes (*Framework*) zur Erschließung von Dokumenten sowie Maßnahmen zur Sicherstellung der Konsistenz bei Nebenläufigkeit in jeweils eigenen Abschnitten eingehend dargestellt. Auf eine Darstellung der Einzelheiten jedes Teildienstes (etwa für Personen, Dokumente, Verzeichnisse, Begriffe etc.) durch Schilderung der Schnittstellen wird verzichtet, da dies konzeptuell unergiebig ist.

6.4.1 Aufbau der Dienstschicht

Der Aufbau der Dienstschicht erfolgt nach folgenden Prinzipien, die den Kern des Entwurfs dieser Schicht bilden:

1. Einerseits soll ein reichhaltiges konzeptuelles Modell nach den fachlichen Erfordernissen des Wissensportals, die in den Anforderungen des Kapitels 4 spezifiziert sind, angeboten werden. Dies geschieht in der üblichen Weise, in der objektorientierte Modelle umgesetzt werden: durch das Angebot einer umfassenden Sammlung von Schnittstellen, die die

konzeptuellen Klassen spezifizieren und ihre Zusammenhänge durch Assoziationen etc. beschreiben. Dieses Angebot wird vorwiegend sowohl von spezialisierten Teilen der Interaktionsschicht und teilweise der Kommunikationsschicht genutzt, als auch von den Implementierungen der Dienstschicht selbst, da hierüber der wohldefinierte, typsichere und effiziente Zugriff auf alle Klassen des Modells möglich ist.

2. Andererseits sollen die generischen und dynamischen Eigenschaften des Dokumentenmodells der *Information Assets* umgesetzt werden: dynamische Erweiterbarkeit der Menge und Art der bestehenden Typen zur Laufzeit des Systems, bidirektionale und sichere Verknüpfungen sowie uniforme Benutzbarkeit aller *Assets*. Statisch typisierte Klassen und Schnittstellen können hier nicht verwendet werden; es ist eine Verwendung von dynamischen Typen in der Art wie in der Speicherungsschicht erforderlich.

Diese Anforderungen stehen zunächst im Widerspruch zueinander. Der Konflikt kann aber durch einen geschickten Entwurf aufgelöst werden, der beide Möglichkeiten harmonisch integriert. Der Weg dahin wird im folgenden systematisch beschrieben.

Zunächst wird ein dem klassischen Vorgehen bei objektorientierter Analyse und Entwurf gehorchendes konzeptuelles Modell auf der Basis der Anforderungen ausgearbeitet. Darin werden insbesondere die Beziehungen zwischen den Klassen berücksichtigt. Das Ergebnis dieser Modellierung ist in Abbildung 6.6 wiedergegeben. Bei der Darstellung in UML wurde in der Abbildung ein neues *Stereotyp* zur Kennzeichnung von *Assets* eingeführt: Ein in der linken oberen Ecke von Klassen positioniertes schwarzes Dreieck. Da fast alle dargestellten Klassen *Assets* sind, das *Asset* aber selbst als (noch abstraktere) Klasse dargestellt ist, vermeidet dies die andernfalls äußerst unübersichtliche Verwendung zahlreichen Kanten zur Darstellung der Vererbungsbeziehungen.

Erkennbar ist, daß fast alle Klassen *Assets* sind; dies entspricht den modellmäßigen Anforderungen. Bis auf einige Ausnahmen sind alle Klassen, die qualifizierte Beziehungen beschreiben, ihrerseits wieder *Assets*. Das bedeutet – um einige damit mögliche Konstruktionen herauszugreifen – beispielsweise, daß Annotationen zu bestimmten Dokumenten in Sammelmappen abgelegt werden können, daß Sammelmappen annotiert werden können, oder Personen andere Dokumente oder Personen (auch sich selbst) bewerten können. Besondere Wichtigkeit besitzen die zahlreichen qualifizierten Beziehungen zwischen Personen und dem allgemeinen *Asset*: Durch sie werden zahlreiche Personalisierungsmöglichkeiten ausgedrückt. Wichtig ist die Beziehung zwischen *Assets* und Begriffen, durch die eine allgemeine Verschlagwortung und Erschließung sämtlicher Informationen ermöglicht wird. Die weitergehenden Funktionen zur automatischen Klassifikation durch Bilden von Klassen (*Clustern*) semantisch verwandter *Assets* oder zum Finden von *Assets* ähnlichen Inhalts werden durch die Konzepte der Klassifikation, Kategorisierung, Kategorie und Ähnlichkeit beschrieben. Diese werden genauer in Abschnitt 6.4.5 beschrieben.

Von besonderer Bedeutung ist die Klasse der Dokumente, da sie dynamisch um weitere Dokumentensubklassen erweitert werden kann (vgl. Anforderung D.5). Die Klassen der Annotation und des Feedbacks sind bereits als Subklassen modelliert. Diese Erweiterungsmöglichkeit erlaubt insbesondere die nahtlose Integration der Informationsartefakte weiterer Informationssysteme, die durch die Speicherungsschicht angesprochen werden können. Der Mechanismus dafür wird unten genauer beschrieben.

Der weitere Aufbau der Dienstschicht unterscheidet zwei wesentliche Teile: Eine Schicht von Schnittstellen, ausgedrückt durch eine Sammlung von *Interfaces* in JAVA, und eine darunterliegende Schicht von Klassen, die diese Schnittstellen implementieren. Die Schnittstellendefinitionen bilden die verbindliche Schnittstelle für die darüberliegenden Schichten und setzen sowohl das konzeptuelle Modell als auch die geforderten dynamischen Zugriffsmöglichkeiten um.

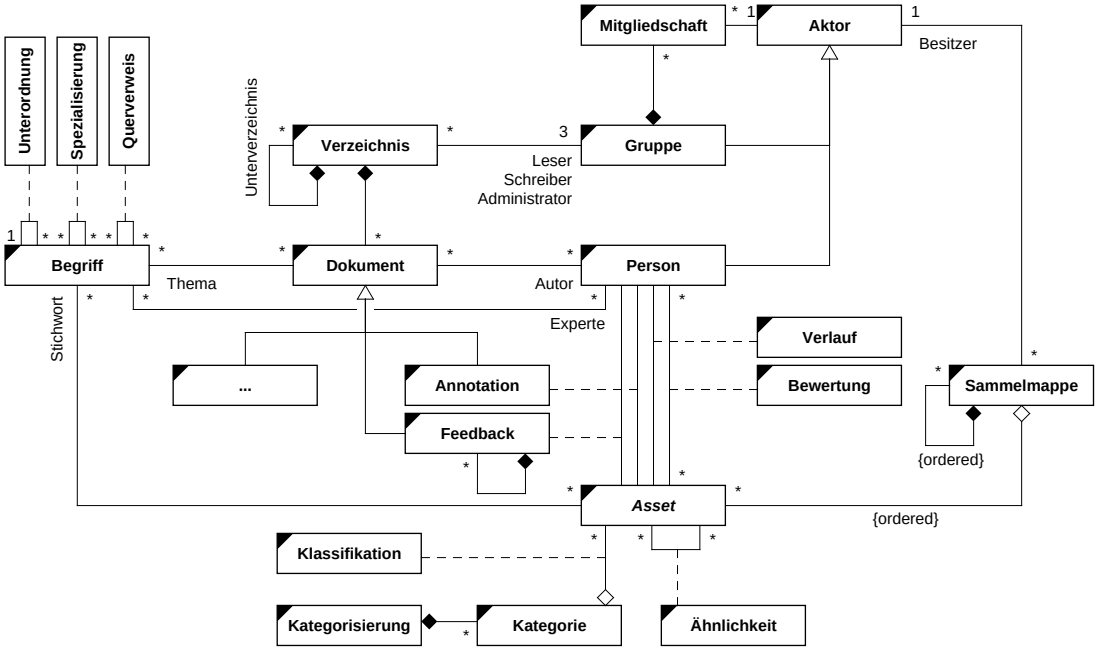

Abbildung 6.6: Konzeptuelles Modell in der Dienstschicht

Es wäre auch möglich gewesen, auf die explizite Schnittstellendefinition per *Interface* zu verzichten und die öffentliche Schnittstelle der implementierenden Klassen als Schnittstelle zu betrachten – ein normalerweise in der objektorientierten Entwicklung übliches Vorgehen. Der abweichende Ansatz wurde gewählt, da er weitere Möglichkeiten bietet und folgende Vorteile hat:

1. Die explizite Schnittstellendefinition stellt eine zweifelsfreie Spezifikation der Funktionalität dar, die vollständig von Implementierungsdetails (Vererbung, private Eigenschaften etc.) abstrahiert. Diese wären zwar formal auch bei den Implementierungsklassen nicht nach außen hin sichtbar, sind aber in der Regel für in den oberen Schichten tätige Entwickler dennoch verfügbar. So können viele unübersichtliche Implementierungsdetails verborgen werden. Dies entspricht dem Entwurfsmuster der Fassade nach [GHJV95].

2. Die explizite Schnittstellendefinition kann besser als Schnittstelle zur Anbindung von externen und eventuell entfernten Diensten genutzt werden. Wenn dazu beispielsweise der in JAVA vorhandene Mechanismus der *Remote Method Invocation* (RMI) verwendet werden soll [Sun01b], ist ohnehin ein *Interface* nötig, das sonst unnötig dupliziert werden müsste. Auch im Falle der Schaffung von weiteren Möglichkeiten zur Interoperabilität durch CORBA oder ENTERPRISE JAVA BEANS (EJB) wäre das Vorhandensein von *Interfaces* nötig [Obj98, Sun01a].

3. Wenn die höheren Schichten lediglich gegen ein explizites *Interface* programmiert werden, ist die statische oder dynamische Austauschbarkeit der tatsächlich verwendeten Implementierungen zur Laufzeit problemloser als bei einer rein durch Klassen definierten Schnittstelle möglich. So lassen sich im Bedarfsfall dynamisch Exemplare verschiedener Implementierungsklassen mischen (die dann natürlich kompatibel zueinander sein müssen), oder verschiedene Klassen für verschiedene Anforderungen oder Szenarien benutzen. Dies bietet für die dynamische Ansprechbarkeit der *Assets* verschiedene Vorteile.

4. Die Skalierbarkeit des Gesamtsystems kann drastisch erhöht werden, indem mehrere Instanzen der Dienstschicht auf verschiedenen Rechnern betrieben werden. Dazu muß von der zentralen Instanz der Interaktionsschicht, die in diesem Szenario nur einmal vorhandenen ist, ein Lastverteiler als Platzhalter (*Proxy*) angesprochen werden. Besitzt dieser exakt die Schnittstelle der Dienstschicht, so kann die verteilte Dienstschicht statt einer einzelnen Dienstschichtinstanz transparent für die Interaktionsschicht angesprochen werden, ohne daß eine einzige Änderung an ihr vorgenommen werden muß. Der Lastverteiler agiert als Dekoration im Sinne des in [GHJV95] vorgestellten Entwurfsmusters. Diese Aufgabe ist am besten durch ein *Interface* (oder mehrere) lösbar, da Platzhalter und tatsächliche Instanz keinerlei gemeinsame Funktionalität besitzen.

Im folgenden wird zunächst der Aufbau der *Interfaces* der Dienstschicht beschrieben. Zur Realisierung der Konzepte von Personen, Dingen, Orten, Begriffen und allen weiteren Arten von *Assets* existieren Sätze von Schnittstellen: jeweils eine zur Spezifikation des *Assets* selber sowie eine weitere, die eine Instanz zur Verwaltung aller *Assets* dieser Art definiert, der sogenannte *Asset Container*. Die Namenskonvention ist dabei so gewählt, daß die erste Schnittstelle den Namen des Konzepts im Singular und die zweite im Plural erhält, also z.B. `Person` und `Persons`. Konzeptuell gesehen ist dies eine Aggregationsbeziehung. Von jedem *Asset Container* darf es jeweils nur genau ein Exemplar geben; dies entspricht dem Entwurfsmuster des *Singletons* nach [GHJV95]. Außerdem existiert eine weitere ausgezeichnete Schnittstelle, `Services` genannt, die ebenfalls ein *Singleton* ist und als Wurzelobjekt der Dienstschicht Zugriff auf alle *Asset Container* bietet. Softwaretechnisch wurde dies so gelöst, daß alle diese *Asset Container*-Schnittstellen von einer gemeinsamen Superschnittstelle namens `AssetContainer` erben, und alle *Asset*-Schnittstellen von einer Superschnittstelle namens `Asset`. Diese Zusammenhänge sind in einem Klassendiagramm in Abbildung 6.7 beispielhaft für genau eine *Asset*-Typ, nämlich Personen, dargestellt. Alle weiteren Typen sind ebenso aufgebaut.

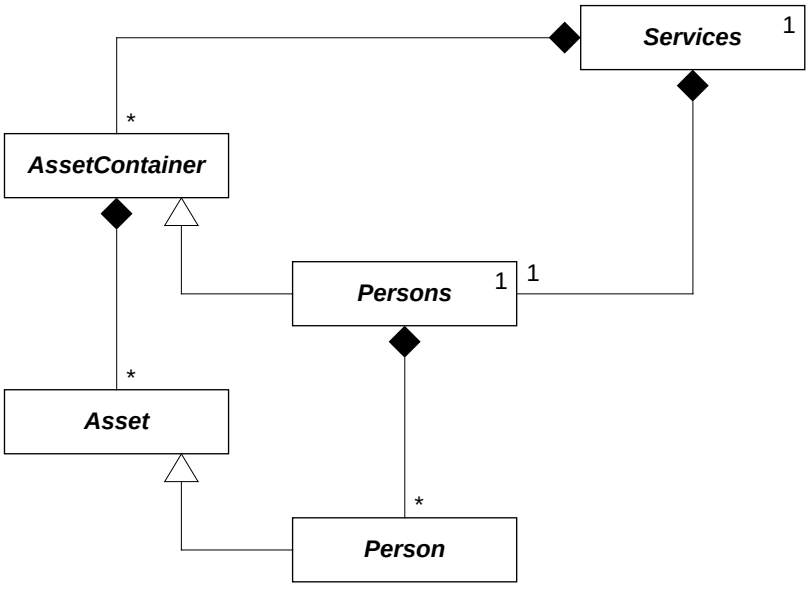

Abbildung 6.7: Schnittstellenstruktur der Dienstschicht am Beispiel der Personen

Die wichtigsten *Asset*-Typen, die von der Dienstschicht angeboten werden, sind Personen, Dokumente, Verzeichnisse und Begriffe. Dazu kommen die ebenfalls als *Asset* modellierten Verknüpfungen zwischen *Assets*. Alle in Abbildung 6.6 wiedergegebenen konzeptuellen Klassen,

die *Assets* sind, werden auf diese Art und Weise durch Schnittstellen repräsentiert. Dies gilt auch für die Beziehungsklassen (etwa Annotationen), so daß einfach auf die Beziehungsobjekte zugegriffen werden kann.

Das *Asset* wird durch das generische, als abstrakte Superschnittstelle benutzte *Interface* des `Asset` beschrieben. Ein solches *Asset* weist die folgenden Merkmale auf:

1. Jedes *Asset* ist von einem bestimmten, festen *Asset*-Typ, der nach der Erzeugung unveränderlich ist. Der Typ wird durch einem Namen identifiziert, etwa „person“.

2. Jedes *Asset* wird innerhalb seines Typs eindeutig durch einen Identifikator, eine Zeichenkette, beschrieben. Dieser ist nach der Erzeugung unveränderbar und ist mit dem Primärschlüssel im relationalen Modell vergleichbar. Der Identifikator wird *AssetId* genannt und ist in der Regel synthetisch, also durch aufsteigende Numerierung oder zufällige Wahl generiert. Zusammen mit dem Typ ist ein *Asset* so systemweit eindeutig identifizierbar; die Kombination wird qualifizierte *AssetId* genannt. Die Zeichenkette „person/hqke4tq9fov“ benennt so das eine bestimmte Person repräsentierende *Asset* zweifelsfrei.

3. Jedes *Asset* besitzt neben der kryptischen *AssetId* einen lesbaren Namen, der nicht eindeutig sein muß und stets änderbar ist. Er dient der primären Identifizierung des *Assets*. Eine Person würde beispielsweise durch ihren Namen gekennzeichnet, ein Dokument durch den Titel oder ein Begriff durch seinen Namen.

4. Neben diesen immer vorhandenen Attributen besitzt ein *Asset* eine Reihe von weiteren Attributen und Beziehungen zu anderen Typen, die durch seinen Typ festgelegt werden und durch das konzeptuelle Modell beschrieben werden. Sie werden durch die Subschnittstellen von `Asset`, also `Person`, `Document` etc. definiert.

5. Ferner besitzt jedes *Asset* einen Zeitstempel, der seine letzte Modifikation angibt.

Die Verwaltung der Menge von *Assets* eines Typs obliegt dem zugehörigen *Asset Container*. Die generische Superschnittstelle `AssetContainer` bietet Funktionalität zur Erzeugung, Identifikation, Existenzprüfung und Zerstörung von *Assets* des verwalteten Typs an. Die Identifikation geschieht dabei ausschließlich über die *AssetId*. Die Funktionalität, die darüber hinausgeht, wie etwa das Angebot spezieller Mengen von *Assets* (bei Personen beispielsweise der gerade angemeldeten), wird in den Schnittstellen der jeweiligen *Asset Container*, also z.B. in `Persons` definiert.

Bislang wurde diese Funktionalität ausschließlich durch statisch typisierte Schnittstellen spezifiziert, wobei gemeinsam benötigte Teile in die Superschnittstellen `Asset` und `AssetContainer` herausfaktorisiert sind. Alle im konzeptuellen Modell vorhandenen Beziehungen werden durch entsprechend typisierte Zugriffsfunktionen abgebildet. Ein Beispiel wäre eine Funktion, die die Autoren eines Dokuments als Iterator von Personen zurückliefert. Dabei ist es sinnvoll, diese Zugriffsfunktion doppelt auszulegen: einmal als eine, die direkt die *Assets* des entsprechenden Typs liefert, und zusätzlich als eine, die lediglich deren *AssetIds* zurückliefert. Dieser pragmatische Aufbau trägt den wechselnden Anforderungen Rechnung, da nicht immer das *Asset* selbst benötigt wird, und so häufig dessen relativ teure Instantiierung vermieden werden kann. Einzelheiten der Realisierung der Abbildung auf die Speicherungsschicht werden später in Abschnitt 6.4.2 diskutiert und Details der Verknüpfungen in Abschnitt 6.4.3.

Die dynamische Erweiterbarkeit von Typen und ihre dynamische Neudefinition sind mit diesen Mechanismen noch nicht abgedeckt. Dazu kommen weitere Mechanismen ins Spiel, die sich auf die zentrale Instanz der `Services`, der `AssetContainer` und `Assets` verteilen. Als wichtigste

Erweiterung kommt eine dynamische Repräsentation der *Asset*-Typen ins Spiel. Die Konzepte sind in Abbildung 6.8 in einem Klassendiagramm gezeigt.

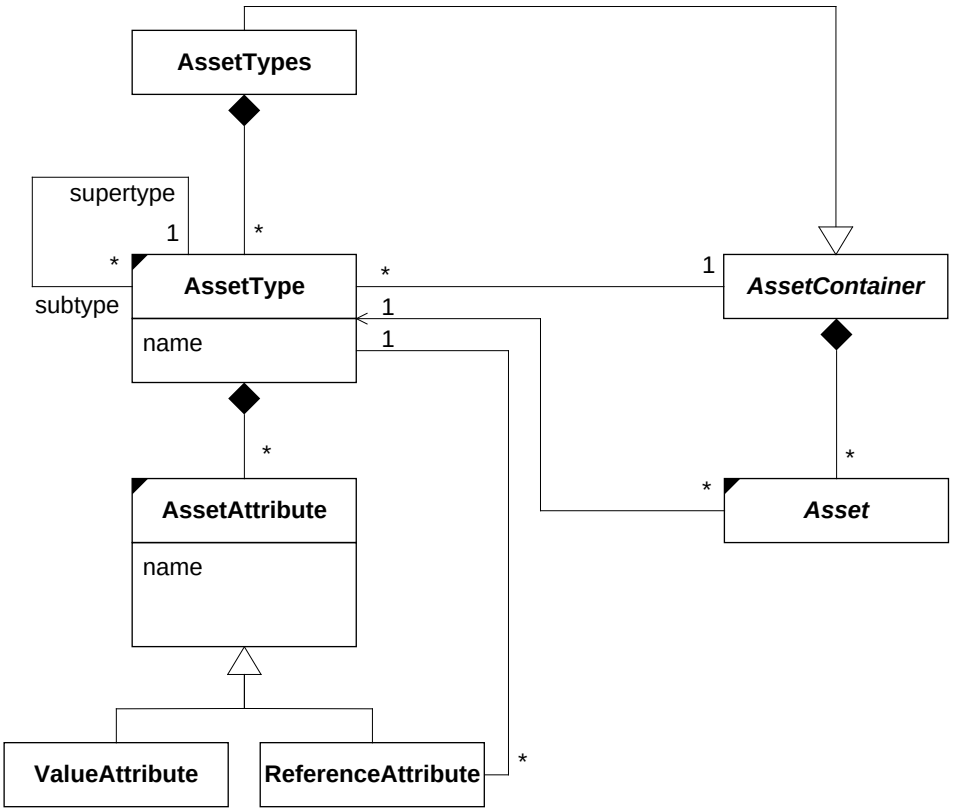

Abbildung 6.8: Aufbau der *Asset*-Typen

Ein `AssetType` wird durch seinen symbolischen Namen und eine Menge von Attributdefinitionen (`AssetAttribute`) spezifiziert. Die Attribute sind grundsätzlich skalar. Entweder handelt es sich bei den Attributen um nicht weiter typisierte Werte, oder aber um unqualifizierte Referenzen auf weitere *Assets*. Ein Typ läßt sich als Erweiterung eines bestehenden Typs definieren und erbt dadurch alle seine Attributdefinitionen. Jeder `AssetContainer` kann mehrere *Asset*-Typen gleichzeitig verwalten, aber die Exemplare eines Typs werden nur von genau einem Container beherbergt. Ferner kennt jedes `Asset` seinen `AssetType`. Die Exemplare dieses Typsystems sind wie alle *Assets* zur Laufzeit konstruierbar und so inspizierbar. Insbesondere kann der *Asset Container* eines gegebenen Typs erfragt werden.

Die Verantwortlichkeiten sind folgendermaßen konstruiert: Auch *Asset Types* und ihre Attributspezifikationen sind *Assets*, und es existieren *Asset Container* zur Verwaltung (in der Abbildung ist nur einer gezeigt). Die zentrale `AssetTypes`-Instanz verwaltet alle Typen und bietet Funktionen zur Erzeugung neuer Typen (Exemplare von `AssetType`) an. Bei der Erzeugung muß zugleich der zugehörige `AssetContainer` angegeben werden. Ferner können hier die bestehenden Typen über ihren Namen identifiziert werden sowie alle existierenden Typen abgefragt werden.

Alle konkreten Implementierungen der Subschnittstellen der *AssetContainer*, also letztlich der konzeptuellen Klassen wie Personen etc., müssen einen künstlichen Typ konstruieren und anbieten, der ihre Schnittstelle und ihre Beziehungen widerspiegelt. Der so aufgebaute Typ für

Personen hat demgemäß zahlreiche Wertattribute für Name, Adresse etc. der Person (vgl. Anforderung P.3) sowie einige Referenzattribute. Die genaue Funktionsweise von mengenwertigen Referenzen, also n:m-Beziehungen, wird eingehend in Abschnitt 6.4.3 erläutert. Analog dazu hat die `Asset`-Schnittstelle Funktionen, die unter Angabe des Attributnamens die entsprechenden Werte bzw. Referenzen (in Form von *AssetIds* oder *Assets*) zurückliefern. Jede dieser Funktionen muß benutzt werden können, auch wenn es sich um ein *Asset* einer speziellen Klasse handelt, die analoge speziell typisierte Funktionen in der Schnittstelle besitzt. Es muß nun jeder konkrete *Asset Container* die dynamische Erweiterung seines ursprünglich fest konstruierten, künstlichen Typs und die Entfernung dieser zusätzlich definierten Attribute unterstützen. Die Werte dieser neuen Attribute sind natürlich nicht mehr über die feste, typisierte Schnittstelle zu erreichen, sondern müssen über die zuletzt beschriebenen Funktionen des *Assets* erfragt und manipuliert werden. Zudem muß es neben der Definition gänzlich neuer Typen möglich sein, diese an bestehende *Asset Container* zum Zwecke der Speicherung zu binden. Typen, die durch Vererbung definiert werden, erben allerdings nicht die Funktionalität der eingebauten Typen (Personen etc.). Sie sollten daher bei Bedarf attributweise erweitert werden. Die Vererbung ist sinnvoller einzusetzen bei dem Aufbau neuer, eigener Typstrukturen, insbesondere bei Subklassen von Dokumenten.

Die Modellierung der Typen als *Assets* entspricht dem im Dokumentenmodell für das Wissensportal geforderten Aufbau (vgl. Abschnitt 5.1.4) und gestattet dadurch die Inspektion, Definition und Änderung von Typen über die Mechanismen der Interaktionsschicht, also z.B. über die webbasierte Oberfläche des Portals selbst.

6.4.2 Abbildung auf die Speicherungsschicht

Unterhalb der Schicht von Schnittstellen, über die auf die konzeptuellen Objekte und die angebotenen Dienste zugegriffen wird, befindet sich die Ebene der Implementierungen dieser Schnittstellen. Hier muß die persistente Verwaltung der Objekte, ihrer Beziehungen und die bereitgestellte Funktionalität realisiert werden. Da die Sicherung der Persistenz eine der heikelsten Aufgaben für den Entwurf ist, wird sie in diesem Abschnitt ausführlich betrachtet. Die Realisierung der Verknüpfungen wird im darauffolgenden Abschnitt 6.4.3 diskutiert.

Zur Sicherung der Persistenz wird die unter der Dienstschicht liegende Speicherungsschicht in Anspruch genommen. Der Grundgedanke ist, jedes *Asset* auf genau ein Inhaltsobjekt der Speicherungsschicht abzubilden und alle *Assets* eines Typs in jeweils genau einem Container der Speicherungsschicht aufzubewahren. Der Container erhält dazu einen Inhaltstyp, der direkt dem Typ des *Assets* entspricht. Da die Realisierung für alle Arten von *Assets* auf schematische Weise geleistet werden kann, wurden abstrakte Klassen entworfen, die den größten Teil der Abbildungsarbeit herausfaktorisieren. Dazu existieren die Klassen `AbstractAssetContainer` und `AbstractAsset`, die jeweils Basisimplementierungen für alle Arten von *Asset Containern* und *Assets* bereitstellen. Ihr Entwurf und ihre Arbeitsweise werden im folgenden vorgestellt.

Im wesentlichen sind hier die Aufgaben zu bewältigen, die häufig als Objekt-Relationale Abbildung bezeichnet werden [Sku98, Akc00]. Da hier nicht die Zerlegung komplexer Objektgraphen geleistet werden muß, sondern lediglich die relativ einfache Abbildung von flach strukturierten *Asset*-Typen auf das Inhaltsmodell der Container der Speicherungsschicht, ist diese Aufgabe gut lösbar. Als zentrale Anforderung muß stets die effiziente und korrekte Abbildung der *Assets* auf die Speicherungsschicht gelten. Dies bedeutet im einzelnen:

- Ein *Asset* darf nur auf explizite Anfrage hin instantiiert werden, also nur dann, wenn es wirklich benötigt wird. Da unter Umständen (abhängig vom Einsatzszenario) viele Tau-

send *Assets* bestimmter Typen existieren können (etwa Dokumente), sind nur diejenigen im Speicher zu materialisieren, die gerade von der Interaktionsschicht benötigt werden, keinesfalls aber einfach alle. Das transitive Instantiieren aller von einem benötigten *Asset* aus referenzierten weiteren *Assets* muß vermieden werden, da dies unter Umständen große Mengen von *Assets* sein können und im Extremfall durch die enge Vernetzung nahezu den gesamten Bestand umfaßt.

- Um andererseits die Leistung des Systems zu erhöhen, müssen einmal materialisierte *Assets* für eine gewisse Zeit materialisiert erhalten bleiben, da das Zugriffsverhalten der Interaktionsschicht typischerweise ein und dasselbe *Asset* mehrfach kurz hintereinander anspricht, beispielsweise zunächst zum Ansehen und kurz danach zum Zurückschreiben von Änderungen. Zudem gibt es *Assets*, die ständig angesprochen werden, wie zum Beispiel die der gerade im Portal angemeldeten Personen. Hier ist also eine geeignete und flexible *Cache*-Strategie nötig.
- Die Konsistenz der *Assets* im nebenläufigen Zugriff zahlreicher gleichzeitig agierender Benutzer des Portals muß gewährleistet sein. Da die Parallelität der Zugriffe in der Interaktionsschicht durch leichtgewichtige Prozesse in JAVA (*Threads*) erreicht wird, ist ein dementsprechender Entwurf nötig. Dieses Thema wird im Abschnitt 6.4.4 separat behandelt, da es relativ unabhängig von der Abbildung auf die Speicherungsschicht ist.

Jede Implementierung der Schnittstelle eines `AssetContainers` bedient sich der abstrakten Superklasse `AbstractAssetContainer`, und die der `Assets` der des `AbstractAsset`. Der abstrakte Container wird dazu von der konkreten Implementierung über seinen Konstruktor mit den Angaben des Managers, eines Inhaltstyps und eines Containernamens parametrisiert, die dazu genutzt werden, einen entsprechenden Container bereitzustellen. Dabei wird insbesondere die Schemaevolution der Speicherungsebene ausgenutzt. Der Inhaltstyp muß von der konkreten Implementierung des *Asset Containers* nach seinen Anforderungen konstruiert werden. Ferner unterstützt die abstrakte Klasse die Identifikation, Erzeugung und Zerstörung von ihren *Assets* und sorgt dabei zugleich dafür, daß die benutzten Inhaltsobjekte ebenfalls angelegt bzw. wieder entfernt werden. Die Instantiierung der konkreten *Assets* wird hier zusammen mit der Realisierung der *Cache*-Strategie geleistet. Um *Assets* der richtigen Klasse erzeugen zu können, wird der abstrakte Container mit dem Klassenobjekt der konkreten Implementierungsklasse des *Assets* versorgt. Durch Ausnutzen der reflektiven Eigenschaften von JAVA kann er dann Exemplare davon erzeugen. Dies entspricht dem Entwurfsmuster der Fabrik und ähnelt dem des Prototypen nach [GHJV95]. Außerdem stellt der abstrakte Container die Funktionalität zum Erzeugen der dynamischen Repräsentationen der *Asset*-Typen aus den von den Subklassen bereitgestellten Inhaltstypen für die Speicherungsschicht bereit und meldet diese selbsttätig bei der zuständigen Instanz der `AssetTypes` an.

Die konkreten Implementierungen der *Asset*-Klassen müssen zwei abstrakte Methoden zur Initialisierung gänzlich neuer Exemplare und zur Rematerialisierung bereits vorhandener *Assets* überschreiben. Alle Attributwerte werden direkt in dem von dem *Asset* aggregierten Inhaltsobjekt gespeichert. Die Zugriffs- und Manipulationsmethoden, die von der Schnittstelle gefordert und von der konkreten Klasse implementiert werden, können dazu das Inhaltsobjekt direkt ansprechen, so daß – sofern keine weitere Funktionalität oder Repräsentationsumwandlung zu Speicherungszwecken benötigt wird – die Implementierung mit jeweils einem Methodenaufruf zu leisten ist. Das Laden der Inhaltsobjekte wird vollständig von einem Implementierungsrahmen der abstrakten Klassen geleistet. Schließlich bleibt festzustellen, daß die jedem Inhaltsobjekt eigene ID zugleich ohne Änderung als *AssetId* benutzt wird.

Die Zusammenhänge dieser Nutzung der Speicherungsschicht sind in Abbildung 6.9 in einem Klassendiagramm wiedergegeben. Alle konkreten Klassen, die die Schnittstellen implementie-

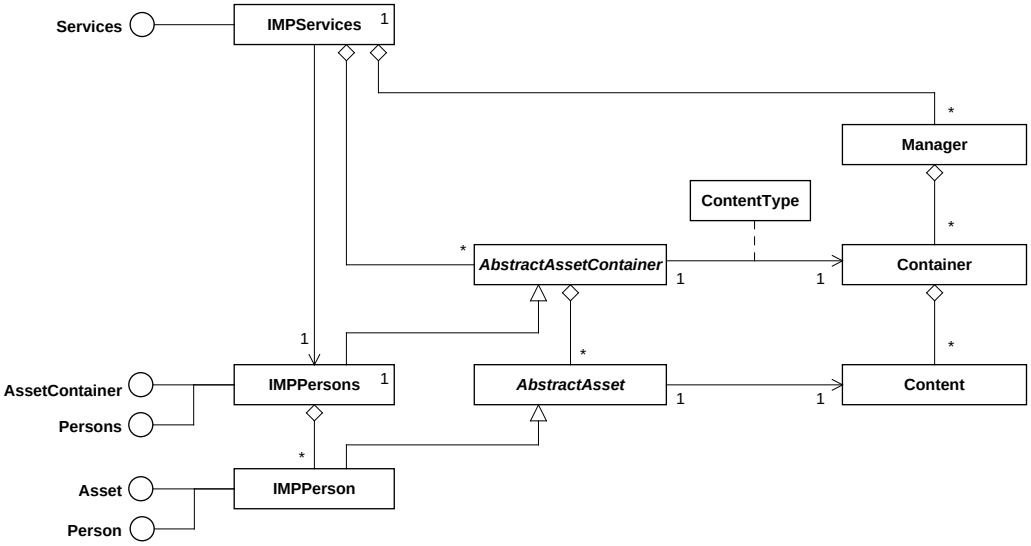

Abbildung 6.9: Nutzung der Speicherungsschicht

ren, besitzen einen Namen mit dem Präfix „IMP“. Hier sind als Beispiel wiederum die Klassen gewählt worden, die Personen verwalten. In dem Diagramm ist ferner zu erkennen, daß alle Manager der Speicherungsebene zentral von der `Services`-Instanz verwaltet werden.

6.4.3 Realisierung von Verknüpfungen

Von essentieller Bedeutung für die Anforderungen der Konsistenz, Aktualität und Performanz der zahlreichen Verknüpfungen ist die Art der Realisierung. Sowohl das dynamische Typmodell als auch die statischen Schnittstellen sehen Referenzen zwischen *Assets* vor. Intern sind alle Referenzen wertbasiert über die *AssetId* der jeweiligen *Assets* aufgebaut, also ähnlich wie im relationalen Datenmodell. Dies wird durch die objektorientiert entworfenen Schnittstellen der *Assets* zum Teil verdeckt. Die verschiedenen Möglichkeiten (Kardinalitäten) der Beziehungen werden im folgenden einzeln analysiert und ihre Handhabung sowohl seitens der Schnittstellen als auch bezüglich der Realisierung beschrieben. Die Darstellung erfolgt dabei so ausführlich, da von der Art der Realisierung die Effizienz der Lösung entscheidend abhängt.

- **1:1**-Beziehungen zwischen zwei *Asset*-Typen. Ein Beispiel wäre ein Bilddokument eines fiktiven dafür eingeführten *Asset*-Typs `Picture`, das einer Person zugeordnet sein kann, um ein Foto der betreffenden Person abzulegen. Die Modellierung der beteiligten Typen muß intern auf der Seite des Bildes ein Referenzattribut vorsehen, das die ID der Person speichert. Durch entsprechende Anfragen läßt sich eine solche Beziehung in beiden Richtungen traversieren, was durch symmetrische Schnittstellen von `Picture` und `Person` möglich ist:

```
interface Picture extends Asset {
  ...
  Person getPerson();
  String getPersonId();
  ...
}
```

```
interface Person extends Asset {
  ...
  Picture getPicture();
  String getPictureId();
  ...
}
```

Die Realisierung erfolgt innerhalb der implementierenden Klassen auf unterschiedliche Art: Auf der Seite der implementierenden Klasse `IMPPicture` muß Gebrauch von einer allgemeinen Identifikationsmethode des *Asset Containers* der Personen gemacht werden, um das *Asset* derjenigen Person zu materialisieren, deren ID das Bild-*Asset* enthält. Diese Methode ist normalerweise ohnehin Bestandteil der Schnittstelle. Auf der Personenseite muß analog dazu eine allerdings extra dafür zu definierende Methode des *Asset Containers* der Bilder verwendet werden, um das Bild, das zu einer Person gehört, zu materialisieren. Solchen Methoden, die im folgenden häufiger vorgestellt werden, wird immer eine ID übergeben und nicht das *Asset*, das die ID enthält, da so das Materialisieren von *Assets* minimiert werden kann. Die folgenden Schnittstellen sind also nötig:

```
interface Pictures extends AssetContainer {
  ...
  Picture getPictureForPerson(String personId);
  ...
}

interface Persons extends AssetContainer {
  ...
  Person getPerson(String id);
  ...
}
```

Die Implementierung der entsprechenden `get...Id()`-Methoden ist auf der Bilderseite trivial; auf der Personenseite muß entweder das Bild materialisiert werden, um daraus die ID zu extrahieren, oder auf der Seite der Bilder wieder eine Methode eingeführt werden, die diese ID liefert:

```
interface Pictures extends AssetContainer {
  ...
  String getPictureIdForPerson(String personId);
  ...
}
```

Der erste Weg ist ineffizient, wenn wirklich nur die ID benötigt wird; es sollte dann der zweite (zusätzlich) implementiert werden. Wird dieser benutzt, aber nach dem Ermitteln der ID doch noch das Bild geladen, wäre der erste wieder effektiver gewesen. Daraus ist zu erkennen, daß die Art der Benutzung genau im voraus bekannt sein muß und die dafür beste Implementierung stark auf die Gestaltung der Schnittstellen einwirkt. Die beste Entwurfsstrategie ist daher, relativ reichhaltige Schnittstellen anzubieten, die beide typische Arten des Zugriffs (direkt auf das Objekt vs. lediglich auf dessen ID) gestatten. Welche Art benutzt wird, muß der Implementierer jeweils aufgrund der Nutzungscharakteristik seiner Module abschätzen.

Die Lokalisierung des Attributs einer bestimmten Seite legt die Abhängigkeitsrichtung und die Lebensdauer fest: Mit dem Löschen der Person muß auch das Bild entfernt werden,

nicht aber anders herum. Diese Erhaltung der referentiellen Integrität wird für alle Beziehungsarten gemeinsam nach ihrer Diskussion besprochen.

- **1:n**-Beziehungen zwischen *Assets* kommen wesentlich häufiger vor. Als Beispiel sei die Beziehung zwischen Verzeichnissen und Dokumenten genannt, wo viele Dokumente genau einem Verzeichnis zugeordnet sind. Auch hier läßt sich die Beziehung durch Einführung eines Attributs auf der Seite der Dokumente modellieren, das die ID des jeweiligen Verzeichnisses enthält. Gegenüber der 1:n-Beziehung erhält die eine Seite, die der Verzeichnisse, nun eine mengenorientierte Schnittstelle:[3]

```
interface Directory extends Asset {
  ...
  Iterator<Document> getDocuments();
  Iterator<String> getDocumentIds();
  ...
}
```

  Es erweist sich wieder als nötig, die Schnittstelle des entsprechenden anderen *Asset Containers* so zu gestalten, daß sowohl an Identifikatoren als auch an Objekten orientiert gearbeitet werden kann, daß beide Richtungen der Traversierung gleichermaßen unterstützt werden und daß die Implementierung der Dokumentenklasse einfach möglich ist:

```
interface Documents extends AssetContainer {
  ...
  Iterator<Document> getDocumentsOfDirectory(String directoryId);
  Iterator<String> getDocumentIdsOfDirectory(String directoryId);
  ...
}
```

- **n:m**-Beziehungen lassen sich genau wie im relationalen Modell nicht direkt ausdrücken, sondern müssen in zwei 1:n-Beziehungen zerlegt werden. Viele solche Beziehungen des konzeptuellen Modells sind bereits durch qualifizierte Assoziationen ausgedrückt worden, etwa Mitgliedschaften, Annotationen oder Ähnlichkeiten (vgl. Abbildung 6.6). Die meisten dieser Beziehungsklassen sind als *Assets* modelliert worden, so daß hier dann normalerweise zwei separate 1:n-Beziehungen verwendet werden können, wie sie eben beschrieben wurden.

  Dennoch müssen auch die nicht weiter qualifizierten n:m-Beziehungen realisiert werden. Eine Realisierungsmöglichkeit ist die Benutzung eines weiteren Containers der Speicherungsschicht zur Abbildung der Beziehung durch die Implementierungsklassen der beteiligten *Assets*. Hier wäre dann nur zu entscheiden, auf welcher Seite dies geschieht. Die Schnittstellen beider Seiten werden am Beispiel der Beziehung zwischen Personen (als Autoren) und Dokumenten erläutert:

```
interface Document extends Asset {
  ...
  Iterator<Person> getAuthors();
  Iterator<String> getAuthorIds();
  ...
}
```

[3] Die Notation mit parametrisierten Typen, hier Iteratoren über Dokumente bzw. Zeichenketten, ist natürlich in JAVA derzeit nicht möglich, verdeutlicht aber die Schnittstelle besser.

```
interface Person extends Asset {
  ...
  Iterator<Document> getDocuments();
  Iterator<String> getDocumentIds();
  ...
}
```

Wird der Beziehungscontainer auf der Seite der Dokumente verwaltet, so werden wieder Methoden im *Asset Container* der Dokumente nötig, die die Dokumente (bzw. deren IDs) einer anzugebenden Person liefern. Zudem kann es nützlich sein, Methoden anzubieten, die die Autoren (Personen) eines per ID anzugebenden Dokuments liefern.

Die Schnittstellen der *Assets* und *Asset Container* müssen also zur Verwaltung der Beziehungen relativ umfangreich ausgelegt werden, um allen typischerweise vorkommenden Anforderungssituationen gerecht zu werden. Der entscheidende Leitgedanke muß dabei immer sein, möglichst wenige *Assets* und damit Objekte aus der Speicherungsschicht zu materialisieren. Es ist nämlich zu bedenken, daß aus jeder Instantiierung eines *Assets* bei den typischerweise verwendeten Managern der Speicherungsschicht eine vergleichsweise teure Datenbankabfrage resultiert. Dies stets in Betracht zu ziehen ist deshalb so wichtig, weil nur der sparsame Umgang mit Rechenzeit und Speicherplatz den Betrieb eines Portalsystems mit akzeptablen Antwortzeiten und ausreichenden Skalierungsreserven erlaubt.

Neben der Performanz der Dienstschicht ist die Aktualität und Korrektheit der verwalteten Beziehungen von Belang. Die dem Sprachgebrauch des relationalen Modells folgend referentielle Integrität genannten Bedingungen müssen von der Dienstschicht für alle Arten von Beziehungen garantiert werden. Zur Vereinfachung werden zunächst 1:1-Beziehungen als spezielle 1:n-Beziehungen aufgefaßt, die alle unqualifiziert sind, und n:m-Beziehungen stets als qualifiziert. So ist nur zwischen unqualifizierten und qualifizierten Beziehungen zu unterscheiden. Ihre Konsistenz wird durch die folgenden Mechanismen gewährleistet:

1. Zur Verwaltung *qualifizierter* Beziehungen wurde ein neuer, spezieller *Asset*-Typ für Beziehungen namens `Relationship` eingeführt. Er ist als nicht notwendigerweise abstrakter Basistyp verwendbar, von dem entweder direkt oder aber von Subtypen seines Typs Exemplare gebildet werden. Ein Exemplar besteht im wesentlichen aus zwei wertbasierten Referenzen auf zwei weitere *Assets*, die sich aus der Angabe des jeweiligen Typs und der jeweiligen ID zusammensetzen. Es beschreibt also die Beziehung zwischen genau zwei *Assets*. Ferner besitzt ein Beziehungsexemplar ein Gewicht als numerisches Attribut, mit dessen Hilfe zahlreiche Klassifikationsaufgaben gut unterstützbar sind [Die01], sowie eine Rolle als rein deskriptives Zeichenkettenattribut, das eine nähere Beschreibung der Art der Beziehung zuläßt. Der Aufbau und die Art der Benutzung ist in Abbildung 6.10 am Beispiel der Annotationen durch ein Klassendiagramm illustriert.

   Alle qualifizierten Beziehungen des konzeptuellen Modells wie Annotationen, Bewertungen, Feedback, Verlauf, persistente Suchergebnisse, Einträge in Sammelmappen oder Verschlagwortungen, die zwischen heterogenen oder homogenen Mengen von *Assets* bestehen, können damit dargestellt werden. Die bereits fest eingebauten Beziehungen werden – wie bereits angedeutet – durch eigene *Asset*-Typen beschrieben, die durch Subschnittstellenbildung auf der Schnittstellenebene und durch Subklassenbildung auf der Implementierungsebene aufgebaut werden. Dies ist in der Abbildung am Beispiel der Annotation gezeigt, die beliebige *Assets* mit Personen (als Autoren der Annotationen) verknüpft. Die qualifizierte Beziehung zwischen Person und *Asset* wird durch eine Verfeinerung der allgemeinen *Asset*-zu-*Asset*-Beziehung dargestellt. Auf der Implementierungsebene wird aus-

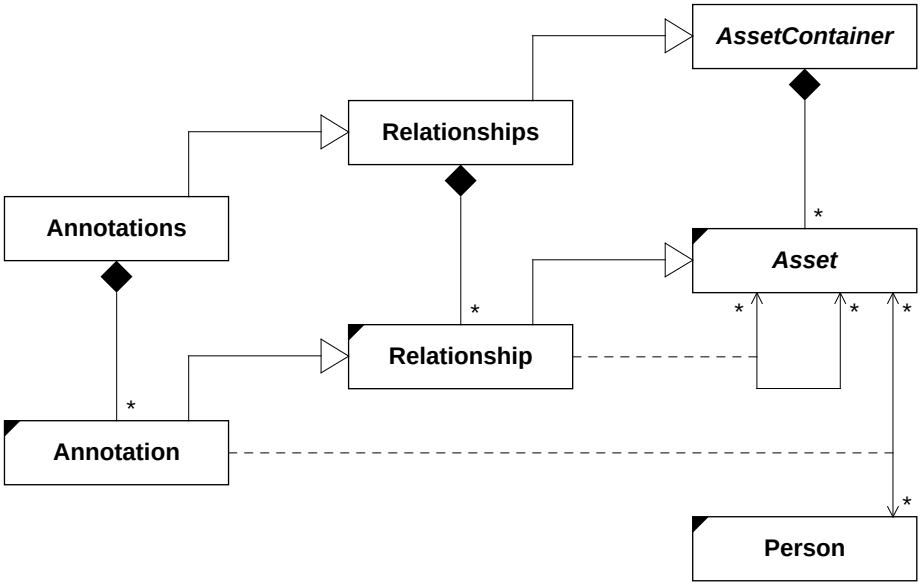

Abbildung 6.10: Aufbau und Verwendung der Beziehungstypen

genutzt, daß ein *Asset Container* verschiedene Arten von *Assets* zugleich verwalten kann (vgl. Abbildung 6.8).

Die wichtigste Entwurfsentscheidung ist dabei die, daß alle Beziehungsobjekte letztlich von demselben evtl. gekapselten *Asset Container* verwaltet werden und so auf der darunterliegenden Ebene in demselben Container eines Managers residieren. Diese Konstruktion ermöglicht folgendes:

- Die Bewahrung der Konsistenz aller Beziehungen im Falle des Löschens eines beliebigen *Assets*, das an irgendeiner Beziehung teilnimmt. Dazu ist es lediglich erforderlich, die Menge der Beziehungsexemplare nach der Typ-ID-Kombination des gelöschten *Assets* zu durchsuchen und die gefundenen Einträge zu löschen. Da alle Beziehungsobjekte in genau einem Container liegen, ist diese Operation überaus effizient zu realisieren. Das automatische Entfernen wird durch eine Schnittstelle aller *Asset Container* ermöglicht, mit der Ereignisbehandlungsfunktionen für bestimmte Ereignisse wie Erzeugen, Modifizieren oder Löschen von *Assets* angemeldet werden können. Über diesen Mechanismus behält die `Relationships`-Basisimplementierung alle relevanten Ereignisse im Blick und kann darauf reagieren. Durch Analyse aller *Asset*-Typ-Informationen kann diese Komponente die Ereignisbehandlung automatisch bei allen existierenden Typen anmelden.
- Dadurch, daß die Beziehungen sowohl auf der Ebene der Speicherungsschicht als auch auf der Ebene der Implementierung der Dienstschicht von den referenzierten Typen getrennt verwaltet werden, ist erst der Gedanke der Integration und der Erschließung der Inhalte vielfältiger heterogener Informationsquellen und -systeme umsetzbar. Es können so beispielsweise völlig neue *Asset*-Typen (etwa als spezialisierte Dokumentenarten) aus neuen Systemen in dem Portal für die Klassifikation, die semantische Erschließung durch Verschlagwortung, die Annotation und Verknüpfung mit Autoren sowie vieles mehr genutzt werden, ohne daß diese Dienste dafür geändert werden müssen.
- Aus dieser Unabhängigkeit resultiert die Möglichkeit, für sehr große Einsatzszenarien

auf hohe Geschwindigkeit spezialisierte Datenbanken zur Beziehungsverwaltung auf der Ebene der Speicherungsschicht anzusprechen. Ebenso lassen sich für bestimmte Arten von *Assets* spezialisierte Persistenzsysteme verwenden.

Werden außer dem Gewicht und der Rolle weitere Attribute zur Qualifikation der Beziehung benötigt, so können Subtypen von `Relationships` diese dem Basistyp über eine spezielle Schnittstelle bekanntmachen. Der Inhaltstyp, der zur Sicherung in der Speicherungsschicht verwendet wird, wird dann entsprechend angepaßt, so daß er die Obermenge aller Unterklassen bildet. Dies hat natürlich den Nachteil, daß viele Beziehungsobjekte ungenutzte Attribute mit sich führen, bietet aber den größeren Vorteil, die Konsistenz effizient garantieren zu können. Zudem sind diese Attribute für andere Subtypen nicht sichtbar. Sind Namen und Basisdatentyp der zusätzlichen Attribute verschiedener Subtypen außerdem gleich, so können diese in ein gemeinsam genutztes Attribut kompaktiert werden.

2. Die Verwaltung *unqualifizierter* Beziehungen beschränkt sich auf homogene Mengen von *Assets*, etwa die Beziehungen zwischen Dokumenten und Autoren. Diese können grundsätzlich auf `Relationships` abgebildet werden, was für n:m-Beziehungen sinnvoll ist, da hier ohnehin entweder ein expliziter, aber künstlicher Beziehungstyp eingeführt oder aber von der *Asset Container*-Implementierung ein zusätzlicher Container der Speicherungsschicht erzeugt und verwendet werden müsste. Der Fall von homogenen 1:n-Beziehungen läßt sich wie oben bereits diskutiert genau wie im relationalen Modell besonders einfach mit wertbasierter Assoziation durch Einführung neuer Attribute lösen. Die gravierenden Nachteile einer solchen Lösung und der Nichtbenutzung der Beziehungstypen sind die zunächst fehlende Garantie der referentiellen Integrität sowie die damit verhinderte Flexibilität und Integrationsfähigkeit. Letzteres läßt sich für bestimmte Beziehungen rechtfertigen, etwa für die zwischen den Begriffen selbst (Unterordnung, Spezialisierung, Querverweis), da es hier ohnehin sinnvoll ist, die für das Gesamtbegriffsnetz benötigten Container der Speicherungsschicht in genau einem Manager zu halten. Die Implementierung der *Asset Container* und *Assets* der Begriffe muß dann die Konsistenz für diese Beziehungen selbst sicherstellen.

   Die praktischen Erfahrungen mit Prototypen, die noch ohne Beziehungstypen gebaut wurden, haben gezeigt, daß die Nichtverwendung der Beziehungstypen die üblichen Probleme hervorbringt: Die Programmierer übersehen aufgrund der Komplexität des konzeptuellen Modells mit seinen zahlreichen Beziehungen die an diversen Stellen nachzupflegenden Änderungen zur Konsistenzerhaltung, sobald neue Beziehungen eingeführt oder geändert werden. Zwei Lösungen dieser Problematik sind denkbar:

   - Umstellung aller Beziehungen, auch der homogenen 1:n-Beziehungen auf die bestehenden oder neu darauf aufgebauten Beziehungstypen.
   - Einführung einer Komponente, die in der Art der Ereignisüberwachung von Beziehungstypen zusätzliche Typen auf Konsistenz überwacht. Diese Komponente müßte die *Asset*-Typen analysieren und entsprechende Ereignisbehandlungsroutinen registrieren. Das Problem ist dabei, daß so keine optionalen Beziehungen ausgedrückt werden können und nicht klar ist, welche Operation zur Konsistenzerhaltung durchgeführt werden muß: Löschen abhängiger *Assets* oder lediglich die Nullsetzung der assoziierenden Attribute. Ein Ausweg wäre die explizite Anmeldung der Beziehungen unter Angabe der zu verwendenden Strategie bei dem Konsistenzdienst.

   Aufgrund der zuletzt angesprochenen Problematik ist die Umsetzung des ersten Weges empfehlenswerter.

Durch die Abbildung sämtlicher qualifizierter Beziehungen auf *Asset*-Typen oder Subtypen von `Relationship` lassen sich auf einfache Art sowohl bidirektional traversierbare Beziehungen als auch deren ständige Konsistenz erreichen. Einfachere Beziehungen können durch die Art der zuvor beschriebenen Schnittstellengestaltung und Implementierung ebenfalls bidirektional verfolgbar sein und durch explizite Ausnutzung eines Konsistenzdienstes integritätserhaltend verwaltet werden.

6.4.4 Erreichung von Nebenläufigkeitssicherheit

Zeit ist das System, das dafür sorgen soll, daß nicht alles gleichzeitig geschieht.
– CEES NOOTEBOOM, Die folgende Geschichte

Die die technische und die inhaltliche Konsistenz betreffende Sicherheit der Implementierung muß durch die Dienstschicht gewährleistet werden. Hier ist das Problem die Sicherstellung der Konsistenz bei lesenden und schreibenden Zugriffen von eventuell zahlreichen gleichzeitig im Portal agierenden Benutzern. Da parallele Zugriffe in der Interaktionsschicht durch Nutzung von leichtgewichtigen Prozessen in JAVA (*Threads*) parallel bearbeitet werden, diese aber entkoppelt laufen sollen, müssen auf dieser Ebene Synchronisierungsmaßnahmen vorgesehen werden. Die Anforderungen sind hierbei:

- Vermeidung der klassischen technischen Nebenläufigkeitsprobleme wie das Auslösen von Ausnahmesituationen durch den gleichzeitigen lesenden und schreibenden Zugriff auf ein und dasselbe Datum. Dies wird in JAVA typischerweise durch Änderungen von Kollektionen hervorgerufen, auf denen gerade eine Iteration durch die Elemente stattfindet. Da alle mengenorientierten Schnittstellen auf Iteratoren basieren, ist gerade dies eine ernstzunehmende Fehlerquelle.
- Vermeidung inhaltlicher Inkonsistenzen, die immer noch trotz bereits ausgeschlossener technischer Probleme durch die gleichzeitige Änderung oder das simultane Lesen und Schreiben etwa desselben *Assets* durch parallele Prozesse auftreten können.

Die Lösung beider Problemklassen läßt sich zusammenfassen durch ein geeignetes Konzept zum Sperren der gerade von der Interaktionsschicht benutzten *Assets* und *Asset Container*. Dies ist an der klassischen Lösung des Leser/Schreiber-Problems orientiert [JV87]. Die Sperrung läßt sich nun auf zwei Granularitätsebenen erreichen:

1. Sperren eines einzelnen *Assets* entweder im lesenden oder im schreibenden Modus, was jeweils für die Dauer der Sperrung das Schreiben ganz und im Falle der Schreibsperre auch das Lesen durch andere Prozesse verhindert. Alle Operationen auf dem *Asset* wie Auslesen oder Ändern von Attributen und Beziehungen müssen entsprechend der Zugriffsart durch Sperren abgesichert werden.
2. Sperren eines ganzen *Asset Containers* im lesenden oder schreibenden Modus. Diese Sperren betreffen Zugriffe, die den Container als ganzes betreffen wie das Erzeugen neuer *Assets*, ihr Löschen und das Iterieren über Mengen von *Assets*, nicht aber das Lesen oder Modifizieren einzelner *Assets*. Dies bleibt weiterhin trotz bestehender Sperren des Containers möglich, da zum einen die jeweiligen Operationen sich nicht gegenseitig beeinflussen, und zum anderen so eine höhere Parallelität und die Verringerung der Gefahr von Verklemmungen (*deadlocks*) erreicht wird.

Neben der räumlichen Dimension der Granularität der Sperren existiert zusätzlich die zeitliche Dimension, also die Frage nach der Dauer der Sperren. Für die Antworten liefert das Prozeßmodell der *Business Conversations* und das daraus für das Portalsystem abgeleitete Modell wichtige Beiträge (vgl. Abschnitt 5.2.1).

Die Prozesse, die die Interaktionsschicht abwickelt, sind als Folge von einzelnen Bearbeitungsschritten strukturiert, wobei jeweils ein Dokument bzw. Formular ausgetauscht wird. Im Falle eines web-basierten Portals tritt hier eine interne Komponente als Dienstleister auf, die ein Dokument absendet, und es nach geraumer Zeit mit den Änderungen des Kunden wieder in Empfang nimmt. Für das Senden und Empfangen sind dabei verschiedene Regeln zuständig. In ihren Implementierungen wird Zugriff auf die benötigten *Assets* der Dienstschicht genommen. Unkritisch ist der Fall, in dem lediglich ein *Asset* angezeigt wird: Hier sperrt die Regel das betreffende *Asset* während der Informationsentnahme zur Generierung des Ausgabedokuments (etwa einer HTML-Seite) und löst die Sperren, sobald das Dokument fertig erstellt ist; danach ist eine Sperrung nicht mehr nötig. Kritisch ist hingegen der Fall der Übermittlung eines Dokuments (etwa durch ein HTML-Formular) zur Änderung durch den Kunden: Hier müßte die erzeugende Regel das betreffende *Asset* mit einer Schreibsperre versehen, das Dokument ausliefern und danach die Sperre zunächst nicht freigeben. Dies würde der nächsten Regel obliegen, die das geänderte Dokument in Empfang nimmt und die Änderungen zurück in das *Asset* überträgt. Dieses theoretisch einleuchtende Modell hält aber leider der Praxis nicht stand, denn aufgrund des in web-basierten Anwendungen verwendeten zustandslosen Protokolls HTTP ist es für den Kunden möglich, die Sitzung zu beenden oder einen anderen Navigationsweg einzuschlagen als geplant, ohne daß der Dienstleister dies erführe. Dadurch wäre die Freigabe gesperrter Objekte nicht sichergestellt, was langfristig das gesamte System zum Stillstand bringen kann.

Zur Synchronisation wurde daher folgender Entwurf vorgenommen: Jedes *Asset* und jeder *Asset Container* besitzt ein Synchronisationsobjekt (ein Semaphor), das gemäß der Diskussion der Granularität von der Interaktionsschicht unbedingt eingesetzt werden muß. Dies verhindert alle technischen und inhaltlichen Synchronisationsprobleme. Weiterhin werden mit diesen Sperrobjekten keine langandauernden, d.h. die Aktionszeit einer Regel überdauernden Sperren gesetzt, was die Ansammlung toter Sperren verhindert. Um dennoch die Regelausführungszeit übersteigende Sperren benutzen zu können, werden explizite, logische Sperren für anwendungsspezifische Aufgabenstellungen eingeführt. Diese sind nicht von vornherein physisch an bestimmte *Assets* gebunden, sondern stellen logische Einheiten der Sperrung dar, die per Konvention zur Sperrung eines *Assets* oder einer beliebigen anderen logischen, funktionalen Einheit verwendet werden können. Denkbar wäre, daß darüber bestimmte Bereiche der Funktionalität derart eingeschränkt werden, daß nur ein oder wenige Benutzer sie gleichzeitig in Anspruch nehmen können, etwa für Administrationsaufgaben. Um Verklemmungen und tote Sperren zu vermeiden, werden diese logischen Sperren mit einem zeitlichen Intervall angelegt, nachdem sie bei fehlender Entsperrung automatisch freigegeben werden. Ferner läßt sich eine Rückruffunktion (*Callback*) installieren, die bei Zeitablauf aufgerufen wird um festzustellen, ob die Sperrsituation noch eindeutig besteht. Wenn ja, kann die Sperrzeit verlängert werden, ansonsten wird sie bei Unsicherheit freigegeben.

Außerdem ist der Entwurf der logischen Sperren so angelegt, daß sie persistent sind, d.h. Systemausfälle überdauern können, und so langandauernde Prozesse der Interaktionsschicht nicht gefährden. Um dies einfach realisieren zu können, sind die Sperren als *Assets* modelliert. Dies bringt zudem erhöhte Flexibilität mit sich: Die Sperren lassen sich notfalls über die Interaktionsschicht per web-basierter Oberfläche administrieren. Zudem ist die Nutzung solcher Sperren in einem verteilten Szenario, wo zur Lastverteilung mehrerer Dienstschichtinstanzen auf verschiedenen Rechnern bestehen, vorteilhaft. Prinzipiell ähneln diese logischen Sperren den verteilten Sperren des SAP R/3-Systems [Zie97, MZ98].

Eine über diese Lösung hinausgehende Sicherung der Integrität durch die Einführung von Transaktionskonzepten wäre für ein zukünftiges System möglicherweise sinnvoll. Eine solche Lösung bedarf allerdings hohen zusätzlichen Aufwandes, insbesondere um die dann nötige Isolation der *Assets* voneinander zu erreichen. Die Einführung expliziter Transaktionen als Konzepte auf der Dienstschicht wäre dazu nötig, was eine relativ große Erweiterung der bestehenden, bisher eher zustandslos ausgerichteten Architektur dieser Schicht zu Folge hätte.

6.4.5 Anbindung eines Klassifikationsframeworks

Eine der wichtigsten Aufgaben heutiger Wissensmanagementsysteme, also insbesondere eines Wissensportals, ist die automatische oder semiautomatische Klassifikation und Erschließung der verwalteten Informationsbestände. Diese Anforderungen wurden bereits in Abschnitt 4.3.7 aufgeführt und folgende wichtige Aufgaben wurden genannt: Klassifikation, Kategorienbildung, Ähnlichkeitssuche und Empfehlungen. Im folgenden werden diese Aufgaben unter dem Begriff der Erschließung zusammengefaßt.

Heutzutage bereits weit fortgeschritten sind die Ergebnisse, die Vektorrepräsentationen für die Darstellung der Informationen nutzen und mathematische Operationen in den resultierenden hochdimensionalen Vektorräumen einsetzen, um Ähnlichkeiten zu entdecken, Klassen zu bilden und neue Vektoren bestehenden Klassen zuzuordnen. Um diese Aufgaben in der Dienstschicht lösen zu können, müßte ein umfangreicher Teildienst entwickelt oder integriert werden. Hier besteht wiederum die Frage, ob ein solcher umfangreicher Klassifikationsdienst selbst entworfen oder ein bestehendes System als Komponente integriert werden soll.

Günstigerweise stellte sich diese Frage beim Entwurf nicht direkt, da zur Erschließung auf ein Rahmenwerk (*Framework*), das von [Büc01] objektorientiert entworfen und in JAVA realisiert wurde, zurückgegriffen werden konnte. Dabei konnte trotz seines Anspruchs auf universelle Einsetzbarkeit der Entwurf des *Frameworks* gut an die Konzepte der Dienstschicht angenähert werden, was im übrigen auch andersherum gilt. In beiden Systemen ist ein abstraktes, informationstragendes Objekt Gegenstand aller Verfahren. Im Portal wird dieses wegen der hier gültigen Ansprüche gemäß des Dokumentenmodells der *Information Assets* modelliert, entworfen und realisiert; im *Framework* steht die effiziente Vektorrepräsentation der Objekte im Vordergrund. Das Bindeglied ist die Umwandlung von *Assets* in semantisch bedeutungshaltige Vektorrepräsentationen, die Auswahl der zur Erschließungsaufgabe passenden Analysebausteine im *Framework* sowie die Rückübermittlung der gefundenen Ergebnisse. Für die Umwandlung müssen sogenannte *Feature*-Selektoren entwickelt bzw. aus der Menge der bereits bestehenden ausgewählt werden, die aus den *Assets* bedeutungshaltige Teilvektoren extrahieren. Insgesamt stellt sich diese Aufgabe als umfangreiche Konfigurationsarbeit dar. Für Details dazu sei auf [Büc01] verwiesen. Weiterhin wurden die Ergebnisse aus [Die01] integriert, die ein System beschreiben, das zur Personalisierung dem Benutzer Vorschläge von für ihn aufgrund seines Interessenprofils und seiner Nutzungscharakteristik besonders relevanten Verweisen auf *Assets* anbietet. Dieses System basiert teilweise auf dem zuerst genannten *Framework*.

Von Bedeutung für den Entwurf und die Realisierung der Dienstschicht ist besonders die Rückübermittlung der nach erfolgter Erschließung anfallenden Resultate. Je nach Erschließungsaufgabe fallen unterschiedliche Ergebnisse an:

- Im Falle der Kategorisierung, also dem automatischen Bilden von Kategorien (*Clustern*) von zu klassifizierenden *Assets* müssen genau diese Kategorien angelegt und die Beziehungen zwischen *Assets* und Kategorien festgehalten werden. Zur Speicherung der Kategorisierung bedient sich die Anbindung zweier eigener *Asset*-Typen: Kategorien und

Klassifikationen. Letztere sind als Subtyp des allgemeinen Beziehungstyps `Relationship` entworfen und repräsentieren die Zuordnung von *Assets* zu Kategorien (vgl. Abbildung 6.6).

- Im Falle der reinen Klassifikation, also der Einordnung in bestehende Kategorien ist die Speicherung wie oben gelöst, es muß lediglich eine weitere Beziehung etabliert werden.
- Im Falle der Ähnlichkeitssuche gibt es zwei Möglichkeiten: Entweder werden die als zu einem gegebenen *Asset* ähnlich erkannten *Assets* transient dem Benutzer angeboten, oder aber die Beziehungen werden permanent festgehalten. Dazu kann wiederum der Beziehungstyp `Relationship` verwendet werden, der die Ähnlichkeit der *Assets* untereinander gut in seinem Gewichtsattribut aufnehmen kann.
- Die Erzeugung von Empfehlungen durch die *Recommendation Engine*, die in [Die01] entwickelt wurde, greift u.a. auf die Bewertungen und Klassifikationen zurück. Die Ergebnisse lassen sich ebenfalls sowohl transient verwenden als auch persistent als (automatisch) generierte Bewertung speichern.

Die Kategorien, die zur Klassifikation verwendet werden, werden ihrerseits wieder von Kategorisierungen genannten *Assets* aggregiert, da es durchaus verschiedene Arten von Klassifikationen mit unterschiedlichen Kategorien geben können soll. Die Klassifikationen, also die evtl. durch ein Gewicht qualifizierten Zuordnungen eines *Assets* zu einer Kategorie, werden durch einen Subtyp des `Relationship`-Typs realisiert.

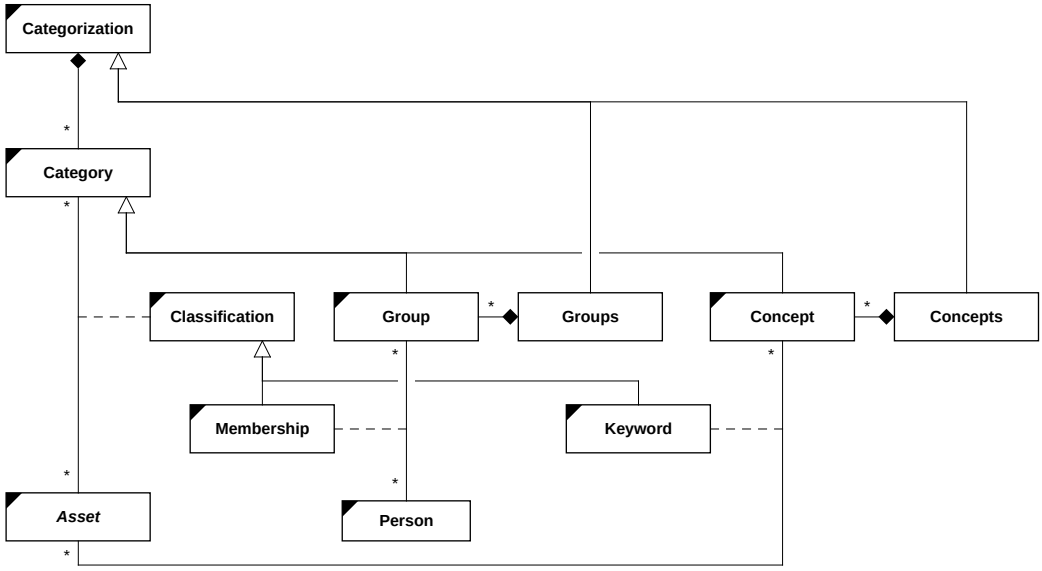

Abbildung 6.11: Konzeptuelles Schema der Kategorisierung

Eine darauf basierende Erweiterung des bisherigen konzeptuellen Modells versteht auch bestimmte weitere Beziehungen, die strukturell zu den Klassifikationsbeziehungen äquivalent sind, als Kategorisierungen. Dazu zählen beispielsweise die Mitgliedschaften von Personen in Gruppen (hier werden die Gruppen als Kategorien, in die Personen eingeordnet werden, betrachtet), oder die Verknüpfung von beliebigen *Assets* mit Begriffen zur Verschlagwortung (hier werden die Begriffe als Kategorien betrachtet, in die *Assets* per Schlagwort eingeordnet werden). Die so entstehende Konstruktion ist in Abbildung 6.11 dargestellt.

Hier tritt nun der interessante Fall auf, daß einzelne *Asset Container* als Kategorisierung betrachtet werden. Da alle Kategorisierungen (Schnittstelle: `Categorization`) von einem (in der Abbildung nicht aufgeführten) *Asset Container* (`Categorizations`) verwaltet werden, also selbst *Assets* sind, muß die Implementierung diese speziellen *Assets* gesondert behandeln, da sie z.B. natürlich nicht löschbar sind. Hier kommt der Vorteil der Beschreibung der Dienstschicht durch *Interfaces* zum Tragen, denn so kann die implementierende Klasse von `Categorizations` einige *Assets* in seine Kollektion aufnehmen, die nicht von der Standardimplementierung der `Categorization` stammen, sondern einer speziellen, die lediglich die genannten *Asset Container* in der Art eines Adapters (nach [GHJV95]) umschließen.

6.5 Die Interaktionsschicht

Die Interaktionsschicht liegt direkt oberhalb der Dienstschicht und stellt das Bindeglied zwischen ihr und der Kommunikationsschicht dar. Ihre Aufgabe ist die Vermittlung zwischen den Diensten, die hier über ihre Schnittstellen angesprochen werden, und den Kunden bzw. Dienstleistern, die mit dem System über die Kommunikationsschicht interagieren. Die Entwurfsziele dieser Schicht sind:

- Umsetzung des aus dem Modell der *Business Conversations* übernommenen regelbasierten Interaktionsansatzes. Die Bearbeitung jeder Anfrage von einem Kunden bzw. das Stellen jeder Anfrage muß von einer spezifischen Regel vorgenommen werden.
- Abstraktion von den konkret verwendeten Aktoren und Medien, also den verwendeten Protokollen. Die gleiche Regel muß prinzipiell für alle Medien verwendet werden können, also beispielsweise sowohl für die Interaktion über HTTP als auch per e-Mail.
- Realisierung der Medien- und Aktorenunabhängigkeit und Trennung von Struktur, Inhalt, Layout und Funktionalität. Dazu muß ein vorlagenbasierter Ansatz eingeführt werden, der die medienunabhängige Generierung der jeweiligen Ausgabeformate erlaubt.
- Realisierung der für diverse Interaktionsprotokolle benötigten Zustandsverwaltung, also der Sitzungen (*Sessions*). Diese wird insbesondere von an sich zustandslosen Protokollen wie HTTP benötigt. Da Sitzungen jeweils nutzerspezifisch sind, müssen die Personen-Dienste der Dienstschicht zur Authentisierung und Identifikation herangezogen werden. Persistente Sitzungsinformationen sind dabei von der Dienstschicht zu verwalten.

Im folgenden werden erst der Aufbau und die Arbeitsweise der Interaktionsschicht beschrieben, worauf eine Darstellung der Konzepte und der Arbeitsweise von Vorlagen folgt.

6.5.1 Aufbau und Arbeitsweise der Interaktionsschicht

Die Hauptaufgabe der Interaktionsschicht, die Vermittlung zwischen Dienstschicht und entfernten Aktoren, geschieht durch das zentrale Konzept der Regel, die auch *Handler* genannt wird. Eine Regel ist für einen speziellen Interaktionsschritt innerhalb einer größeren, eventuell langandauernden Konversation zuständig. Dies folgt dem Modell der Regel in den *Business Conversations* (vgl. Abschnitt 5.2.1). Die wesentlichen Eigenschaften einer Regel sind:

1. Eine Regel abstrahiert von dem konkreten Partner, mit dem sie interagiert (Aktorenunabhängigkeit), indem keinerlei Annahmen über Protokolle und Modalitäten der Interaktionen

vorausgesetzt werden. Diese Entkopplung wird von der Kommunikationsschicht oberhalb der Interaktionsschicht in Verbindung mit den Sitzungen dieser Schicht geleistet.

2. Eine Regel abstrahiert vom Medium, über das die Interaktion mit dem Partner geschieht (Medienunabhängigkeit). Die gleiche Regel kann also ohne Änderung beispielsweise sowohl für eine web-basierte Kommunikation als auch für eine über e-Mail verwendet werden. Diese Leistung wird durch das Konzept der Vorlagen und von der Kommunikationsschicht realisiert.

Der Entwurf der Interaktionsschicht sieht als wichtigste konzeptuelle Klassen daher Sitzungen (*Sessions*), Regeln (*Handler*) und Vorlagen (*Templates*) vor. Die Schnittstelle der Interaktionsschicht zur Kommunikationsschicht wird durch die Sitzungen gebildet. Die aktoren- und medienabhängigen Anteile für verschiedene Protokolle (HTTP, e-Mail, FTP, ...) werden von Server-Modulen der Kommunikationsschicht implementiert, und diese greifen über eine medien- und aktorenunabhängige Schnittstelle auf die Sitzungen zu. Diese delegieren gemäß spezieller Konfigurationseinstellungen die jeweiligen Anfragen an die dafür definierten Regeln, die dann die Parameter in einem neutralen Format über die Sitzungsschnittstelle entgegennehmen und mittels des Mechanismus der Vorlagen geeignete Antworten zusammenstellen. Diese werden dann von den Server-Modulen entgegengenommen und gemäß der verwendeten Protokolle weitergeleitet.

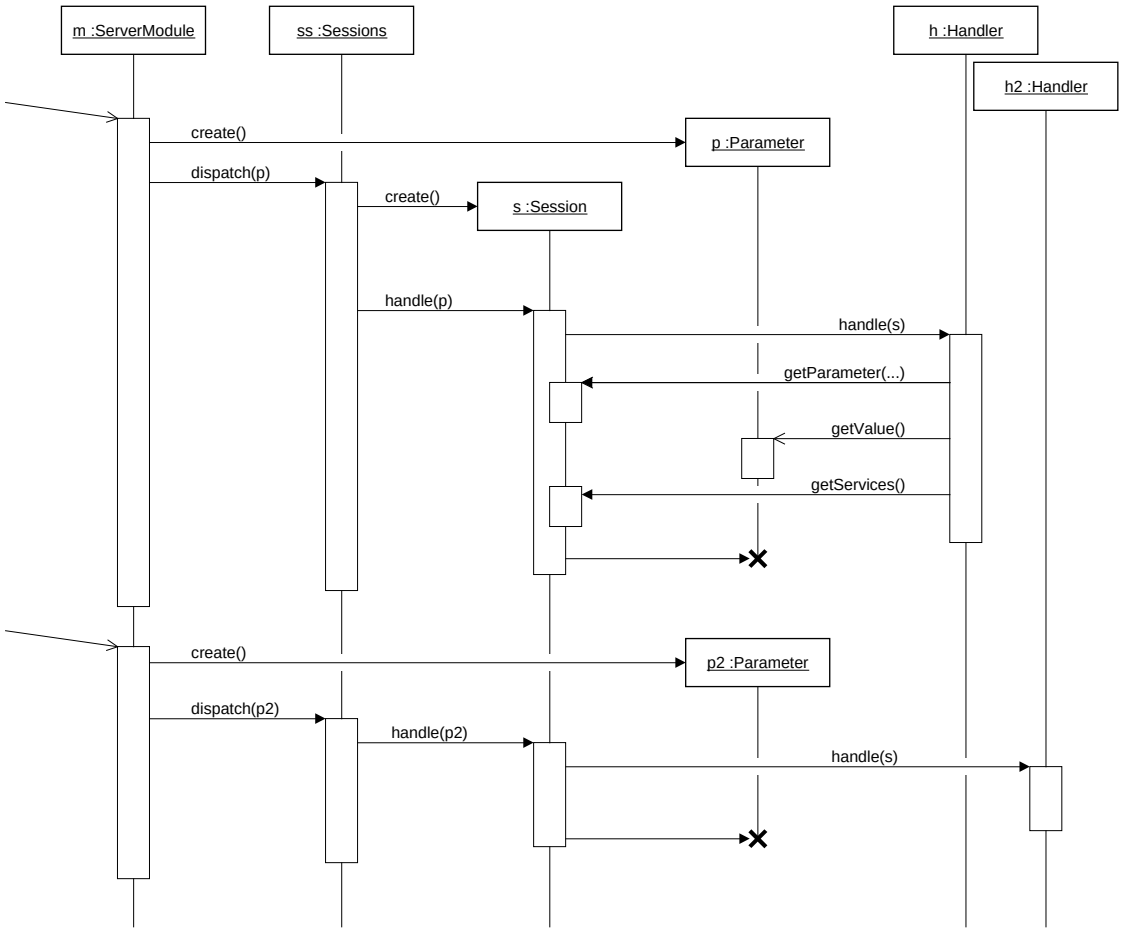

Abbildung 6.12: Interaktionen bei der Regelausführung

Der genaue Ablauf einer Regelausführung ist in Abbildung 6.12 durch ein Sequenzdiagramm in UML illustriert. Es werden dabei folgende Aktionen durchgeführt:

1. Die Ausführung wird hier von einem externen Aktor angestoßen, der eine Anfrage an ein beliebiges Server-Modul richtet, beispielsweise an ein Web-Server-Modul.

2. Das Modul reagiert entsprechend seinem Protokoll (etwa HTTP) und extrahiert aus der Anfrage die Parameter. Diese werden in einem neuen Parameterobjekt gespeichert, das neutralen Zugriff auf alle Werte bietet, der von den Protokollen etc. unabhängig ist. Die Parameter werden von dem Parameterobjekt dabei als geordnete Liste von Schlüssel/Wert-Paaren verwaltet, die alle als Zeichenketten ansprechbar sind. Weitergehende Annahmen über Typ oder Semantik der Parameter werden nicht getroffen.

3. Sobald die Parameter extrahiert sind, delegiert das Modul die weitere Bearbeitung der Anfrage an den Sitzungsdienst (`Sessions`). Dies ist ein genau einmal existierendes Exemplar (*Singleton*), dem die Verwaltung aller Sitzungen obliegt. Dazu werden die Parameter sowie eine protokollunabhängige Sitzungskennung (*Session ID*) übergeben. Die Sitzungskennung wird ähnlich wie die Parameter vom Modul aus dem Protokoll bzw. der Anfrage extrahiert. Im Falle von HTTP könnte diese über URL-Parameter oder *Cookies* transportiert werden.

4. Aufgrund der übergebenen Sitzungskennung muß die Sitzungsverwaltung erkennen, ob sich die Anfrage auf eine bereits existierende Sitzung bezieht oder ob eine neue Sitzung begonnen werden soll. Im hier illustrierten Fall wird zunächst eine neue Sitzung `s` erzeugt. Die Sitzung ist dabei stets an eine Person der Dienstschicht gebunden, deren Anmeldeinformationen (Kennwort, Berechtigungen etc.) verwendet werden.

5. Nach der Erzeugung übernimmt das neugeschaffene Sitzungsexemplar die weitere Bearbeitung. Durch eine konfigurierbare Auflösungsstrategie, die eine Zuordnung der Namen der einzelnen Anfragen (die mit den Parametern zusammen übermittelt werden) zu Regeln vornimmt, wählt die Sitzung das Regelexemplar aus, die für diese Anfrage vorgesehen ist. Die Regeln werden durch Klassen implementiert und müssen eine bestimmte Schnittstelle erfüllen, die im wesentlichen die Existenz einer Bearbeitungsmethode `handle()` fordert. Die Regelexemplare werden dynamisch beim Start gemäß der Konfiguration instantiiert und initialisiert. Für die Bearbeitung aller Anfragen wird jeweils genau ein Exemplar jeder Klasse erzeugt, das über die gesamte Betriebsdauer des Systems hinweg verwendet wird. Bei der Implementierung der Regeln ist also zu berücksichtigen, daß zur Behandlung stets dasselbe Objekt verwendet wird und daß dies auch nebenläufig durch mehrere Prozesse gleichzeitig geschehen kann.

6. Das zuständige Regelexemplar wird aufgerufen, wobei es das Sitzungsobjekt übergeben bekommt. Es bildet so den Kontext dieses Schrittes der Anfragebearbeitung. Von dem Sitzungsobjekt lassen sich die Parameter, die Anfrage und weitere Informationen abfragen. Insbesondere bietet es Zugang zu der Dienstschicht über das zentrale `Services`-Objekt, über das alle *Asset Container* erreichbar sind (vgl. Abschnitt 6.4.1).

7. In der Abbildung sind die Benutzung der Parameter und der Dienste skizziert. Die komplexen Benutzungsmodalitäten für die Vorlagen sind hier nicht weiter illustriert; diese werden im folgenden Abschnitt 6.5.2 ausführlich diskutiert.

8. Nach Beendigung der Regelausführung entfernt das Sitzungsobjekt die Parameter und gibt die Kontrolle an die Sitzungsverwaltung zurück. Diese liefert das Ergebnis dem Server-Modul und die Anfrage ist damit für die Interaktionsschicht beendet.

9. Eine weitere Anfrage für die gleiche Sitzung ist in der unteren Hälfte der Abbildung 6.12 dargestellt. Hier entfällt die Erzeugung eines neuen Sitzungsobjektes; das eben erzeugte wird hier wieder verwendet. Die Abarbeitung folgt demselben Schema wie eben mit dem Unterschied, daß ein anderes Regelexemplar verwendet wird. Details der Regelausführung sind ferner weggelassen worden.

Neben der Abwicklung der Sitzungen durch Erzeugung und Nutzung der Sitzungsobjekte stellt die Sitzungsverwaltung ferner sicher, daß inaktive Sitzungen nach einem konfigurierbaren Zeitraum entfernt werden.

Das Laden der Regelexemplare und das Auflösen der Anfragenamen in Regeln geschieht durch eine Subkomponente, die in der Abbildung nicht weiter dargestellt wurde. Diese *Dispatcher* genannte Komponente wird durch eine Schnittstelle (eine abstrakte Klasse in JAVA) beschrieben, zu der es verschiedene konkrete Implementierungen gibt. Die Basisimplementierung liest die Konfiguration der Regeln aus einer Konfigurationsdatei, erzeugt durch Aufruf von in Subklassen implementierten Methoden die Regelexemplare, verwaltet diese und implementiert die Namensauflösung. Dadurch ist es möglich, zu verschiedenen Zwecken verschiedene *Dispatcher*-Subklassen zu bauen, die beispielsweise die Regelklassen dynamisch nachladen, wenn sie geändert und neu übersetzt wurden. Dies ist gerade bei der Entwicklung und beim Testen der Regeln nützlich, da so nicht das gesamte System neu gestartet werden muß. Im Produktivbetrieb kann dann eine Implementierung verwendet werden, die diese relativ zeitaufwendigen Prüfungen unterläßt.

Wie bereits oben angesprochen, besitzen die Anfragen, die von Aktoren in der Kundenrolle an das System gerichtet werden (hier also zunächst an das betreffende Server-Modul) einen Namen, der dazu verwendet wird, die Bearbeitungsregel auszuwählen. Dies entspricht soweit genau den Anfragen aus dem Modell der *Business Conversations*. Außerdem sind diese Anfragenamen eng an die Existenz von einzelnen *Assets* gebunden. Für die Anfragenamen wurde deshalb folgende Namenskonventionen entwickelt:

- Da viele Anfragen direkt oder indirekt *Assets* betreffen, die durch die Anfrage beispielsweise angezeigt oder verändert werden sollen, sind bis auf gewisse Ausnahmen die Namen analog zu qualifizierten *Asset IDs* aufgebaut. So hat die Anfrage „person/hqke4tq9fov“ die Anzeige des *Assets* einer ganz bestimmten Person zum Ziel.
- Um durchzuführende Aktionen auf *Assets* wie die Änderung bestimmter Attribute, das Löschen oder den Wunsch nach Erzeugung eines neuen *Assets* auszudrücken, wird eine ähnliche Syntax verwendet, die statt des Typs den symbolischen Namen des zuständigen *Asset Containers* und statt der ID die Aktion trägt. Die Anfrage, ein *Asset* zu modifizieren könnte demgemäß „persons/edit“ lauten. In diesem Fall müssen natürlich die zu ändernden Attributwerte und insbesondere die *Asset ID* des zu ändernden *Assets* als Parameter mit übertragen werden.

Zum Binden der Regeln an Anfragenamen wird eine Konfigurationsdatei benutzt, die eine Zuordnung von Anfragenamen zu Klassennamen vornimmt. Dabei können Platzhalter (Joker) verwendet werden, um beispielsweise alle mit „person/“ beginnenden Anfragen zum Anzeigen bestimmter Personen von einer Regel bearbeiten zu lassen. Die bisher vorgestellten Konzepte sind in Abbildung 6.13 in einem konzeptuelle Klassendiagramm wiedergegeben.

Die Implementierung einer Regel kann über das Sitzungsobjekt an den Anfragenamen gelangen, um beispielsweise die *Asset ID* zu extrahieren und um weitere Parameter in Erfahrung zu bringen. Die medien- und geräteunabhängige Erzeugung der Ausgabe wird über den Mechanismus der Vorlagen bewerkstelligt, der im folgenden Abschnitt eingehend dargestellt wird.

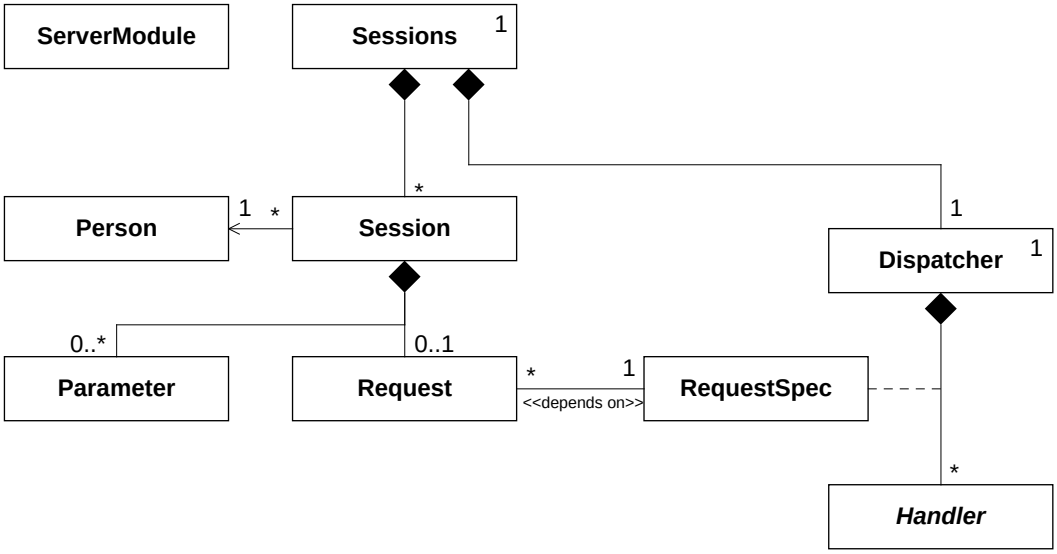

Abbildung 6.13: Konzepte der Interaktionsschicht

6.5.2 Funktionsweise der Vorlagen

Die Kernanforderungen der Trennung von Struktur, Inhalt, Layout und Funktionalität auf der einen Seite und die Forderung nach Medien- und Aktorenunabhängigkeit auf der anderen Seite werden durch das im folgenden vorgestellte Konzept der Vorlagen abgedeckt.

Das bereits oben dargestellte Grundkonzept basiert auf dem Vorhandensein von medien- bzw. protokollspezifischen Server-Modulen, die über neutrale Schnittstellen mit den Regeln kommunizieren. Diese müssen im nächsten Schritt über eine wiederum medienneutrale Schnittstelle medienspezifische Vorlagen ansprechen, um aus ihnen und den Informationen, die sie von der Dienstschicht erhalten, die entsprechenden Ausgaben erstellen zu können, die von dem gerade verwendeten Protokoll benötigt werden. Durch die Bereitstellung der für eine logische Anfrage benötigten Vorlagen für alle Medien, Protokolle und Geräte lassen sich ohne Änderung der Regeln alle Interaktionsformen realisieren. Welche Vorlage verwendet wird, hängt dabei direkt von dem die Regelausführung initiierenden Server-Modul ab.

Natürlich ist es nicht sinnvoll, alle möglichen Anfragen, also die gesamte Funktionalität des Systems, für alle Medien und Geräte bereitzustellen; hier ist eine den Erfordernissen der Medien, Geräte und Aktoren angemessene Einschränkung des Funktionsumfangs sinnvoll. Beispielsweise ist die aufwendige Bearbeitung von Personendaten über ein Mobiltelefon per WAP derzeit praktisch nicht möglich. Eine ausführliche Diskussion der Frage, was in einem solchen Szenario sinnvoll umsetzbar ist, findet sich in [Bar01].

Konzepte, die Visualisierungsvorlagen mit Funktionalität verbinden, gibt es gerade für webbasierte Systeme einige. Als wichtige Vertreter sind die *Java Server Pages* (JSP), *Active Server Pages* (ASP) und der *PHP Hypertext Preprocessor* (PHP) zu nennen [Sun01c, ASP01, PHP01]. Deren Konzepte, die alle die Generierung von HTML-Seiten zum Ziel haben, haben den gemeinsamen Ansatz, in HTML-Seiten durch spezielle Auszeichnungen markierten Programmcode einzubetten, der bei Auslieferung der Seite durch einen geeigneten Server interpretiert wird. Der Code wird dabei dann durch die Ergebnisse, die er liefert, ersetzt. Der gravierende Nachteil dieser Lösungen ist, daß so keine Trennung des Layouts von der Funktionalität erreicht wird, was

eine wichtige Anforderung an das Portalsystem war, um eine aufgabengerechte Arbeitsteilung und die einfachere Anpaßbarkeit des Systems zu ermöglichen. Weiterhin sind diese Ansätze entweder auf HTML als den Programmcode umgebende Vorlage oder auf die zu verwendende Programmiersprache festgelegt.

Ein anderer häufig genutzter Ansatz basiert auf der Nutzung von XML mit XSL oder Transformationen durch XSLT [WWW98a, Dea99, Cla99]. Die diesen Ansätzen zugrundeliegende Annahme, daß alle Informationsobjekte in Form von XML-Dateien (bzw. in XML-Datenbanken) vorliegen, wird von dem hier entworfenen System aber nicht erfüllt. Besonders nachteilig wegen des hohen Aufwandes wäre also der Versuch, die *Assets* zunächst in eine XML-basierte Repräsentation umzuwandeln, um diese dann gemäß des gewünschten Ausgabeformats nochmals zu transformieren.

Diese Nachteile werden durch ein auf format- und sprachneutralen Platzhaltern beruhendes Konzept für Vorlagen vermieden, das für dieses Portal neu entworfen wurde. Es basiert auf folgenden Annahmen:

- Die meisten Arten von Vorlagen sind textbasiert (etwa die für HTML, WML, e-Mail) bzw. besitzen bei geeigneter Behandlung eine ähnliche Struktur (etwa Textverarbeitungsdokumente). Die Platzhalter lassen sich also auf einfache Weise in die Texte einbetten.
- Die Gestaltung der Vorlagen kann gerade im Falle von HTML-Vorlagen von ausgebildeten Web-Designern geleistet werden, die keine Kenntnisse in Programmierung besitzen. Bei Bereitstellung von ausreichender Funktionalität seitens der Regeln durch dem Bedarf entsprechende Unterstützung von Platzhaltern ist die Erstellung oder Anpassung weiter Teile der Benutzungsoberfläche alleine durch die Gestaltung der Vorlagen möglich.
- Darüber hinausgehende Anpassungen lassen sich relativ einfach von Programmierern auf der Ebene der Regeln durchführen, da sie gegen die stabilen Schnittstellen der Dienstschicht arbeiten und so von vielen komplexen Details (wie Sicherung der Konsistenz, Nebenläufigkeitssicherheit und Persistenz) abstrahieren können.

Durch den Verzicht auf den Einsatz einer Programmiersprache ist die Ausdrucksmächtigkeit innerhalb der Vorlagen zunächst relativ gering. Tatsächlich reicht ein zu einfaches Konzept, das lediglich skalare Platzhalter vorsieht, die von den Regeln durch jeweils genau ein Textfragment ersetzt werden, nicht aus, um Gestaltung und Funktionalität strikt zu trennen. Der primäre Grund ist in den häufig benötigten Wiederholungsstrukturen wie Listen und bedingten Einschüben zu sehen: Listen (beispielsweise von Personen) lassen sich nicht durch skalare Platzhalter beschreiben, wenn die gestaltenden Elemente (also z.B. die Markierungen in HTML für Listen, Zeilen und Zellen) lediglich in der Vorlage erscheinen dürfen. Anders herum ließe sich die Liste natürlich als ganzes als Platzhalter ansehen; dann müsste sie aber wiederum in der Regel erzeugt werden und die Trennung von Gestaltung und Funktionalität wäre verletzt. Gleiches gilt für bedingte Teile der Vorlage, also z.B. Bereiche, die selektiv je nach Berechtigungsstufe des aktuellen Benutzers verfügbar sein sollen.

Die Lösung dieser Problematik liegt in der Einführung ebenso gestaltungsneutraler Platzhalter zum Ausdrücken von Listen und Bedingungen. Es werden so also insgesamt drei Arten von Platzhaltern benötigt. Ihre Syntax und Semantik werden auf der Ebene der Vorlagen zunächst vorgestellt; wie die Anbindung an die Regeln geschieht, wird danach eingehend beschrieben.

- **Skalare Platzhalter:** Diese Platzhalter können überall innerhalb der Vorlage erscheinen. Sie werden durch spezielle Schreibweise gekennzeichnet, um sie effektiv identifizieren zu

können: Jeder Platzhalter beginnt und endet mit dem Dollarzeichen '$' und darf lediglich Buchstaben und Ziffern enthalten. Ein Beispiel wäre `$name$`. Ein skalarer Platzhalter wird bei der Auslieferung der Vorlage von der Regel durch ein beliebiges Textfragment ersetzt. Die Ersetzung ist dabei im Programmcode der Regel definiert. Hierbei gilt natürlich, daß der Ersetzungstext medienneutral sein muß, um problemlos in alle Arten von Vorlagen eingebettet werden zu können. Insbesondere dürfen keine formatspezifischen Bestandteile verwendet werden, also z.B. HTML-Fragmente. Die Namenskonventionen sind alleine zwischen den Programmierern der Regeln und den Gestaltern der Vorlagen zu klären.

- **Bedingte Platzhalter:** Die Platzhalter zur Markierung bedingter Passagen können überall innerhalb der Vorlage verwendet werden. Sie bestehen im Unterschied zu skalaren Platzhaltern jeweils aus einem Paar einander zugeordneter Platzhalter, die den bedingten Bereich einschließen. Um sie von skalaren Platzhaltern eindeutig unterscheiden zu können, werden sie durch zusätzliche eckige Klammern gekennzeichnet. Ein Paar bedingter Platzhalter könnte also z.B. `$[isAdmin$ ... $isAdmin]$` lauten. Der eingeschlossene Teil wird nur dann aus der Vorlage in das Ergebnis übernommen, wenn die Bedingung, die mit dem Platzhalter ausgedrückt wird, wahr ist. Die Entscheidung darüber wird innerhalb der betreffenden Regel gefällt. Eine an die jeweiligen logischen Bedingungen angelehnte Namensgebung ist empfehlenswert.

  Neben der eben dargestellten Syntax existiert eine erweiterte, die auch die jeweils andere Alternative der Bedingung berücksichtigt. Ihre Syntax lautet:

```
$[condition$
  ...
$]condition[$
  ...
$condition]$
```

  So lassen sich bequem zwei Alternativen mit einer Bedingung ausdrücken. Hilfreich ist diese Funktion gerade zur Personalisierung der Vorlagen, da mit den bedingten Teilen sehr selektiv Funktionalität ausgewählt werden kann.

- **Listen-Platzhalter:** Platzhalter für Listen lassen sich überall in einer Vorlage einsetzen. Sie kennzeichnen Bereiche, die wiederholt in Form einer Liste ausgegeben werden sollen. Ähnlich wie die bedingten Platzhalter umschließen sie den jeweiligen Bereich durch ein Paar von zusammengehörigen Platzhaltern mit bestimmter Syntax. Da der wiederholte Bereich natürlich nicht immer identisch repliziert werden soll, muß er weitere Platzhalter enthalten, die in jeder Wiederholung mit neuen Werten gefüllt werden. Die enthaltenen Platzhalter sind dabei dergestalt an jede Iteration gebunden, daß sie jeweils andere, neue Werte für das nächste Listenelement liefern.

  Syntaktisch wird die Bindung durch Qualifikation der Namen ausgedrückt: Jeder eröffnende Listenplatzhalter führt einen neuen Namensbereich ein, indem er einen Präfix angibt, z.B. durch `$[persons p$`. Dies unterscheidet die Syntax zugleich von der der bedingten Platzhalter. Alle in der Liste enthaltene Platzhalter, die an der Iteration teilnehmen, müssen dann mit dem Präfix qualifiziert werden, also z.B. durch `$p.name$` statt `$name$`. Da alle Konstrukte orthogonal zueinander verwendet werden können, also beispielsweise Listen in Listen geschachtelt werden können, ist die Qualifikation unbedingt erforderlich, um die Bindung an die korrekte Liste ausdrücken zu können.

  Um weitreichendere Gestaltungsmöglichkeiten zu haben, sehen die Listenplatzhalter vor, daß genau zwei Beispieleinträge der Liste angegeben werden müssen, von denen der erste für alle Elemente der Liste bis auf das letzte und der zweite für den letzten Eintrag

benutzt wird. Existiert nur ein Eintrag, so gilt er als der letzte. Die Identifikation der Grenze zwischen erstem und zweitem Eintrag erfolgt dabei durch die Analyse der enthaltenen Platzhalter: Ist die Anzahl $2n$ gerade, und entsprechen die in der ersten Hälfte genau denen in der zweiten, so beginnt die zweite Hälfte genau mit dem n-ten enthaltenen Platzhalter. Ein Beispiel, das eine Autorenliste generieren soll, verdeutlicht die Vorteile:

```
Autoren sind:
$[persons p$
  $p.name$, $p.name$
$persons]$
```

Im Falle nur eines Autors wird der Teil ab dem zweiten qualifizierten Platzhalter verwendet, sonst der erste bis zum vorletzten Eintrag und dann der zweite für den letzten. Jeweils zwei Autorennamen sind so durch ein Komma getrennt, das aber entfällt, wenn es nur einen Autor gibt. Ähnlich wie bei den bedingten Platzhaltern existiert eine syntaktische Form, die einen Alternativ-Zweig für den Fall der leeren Liste vorsieht. Wieder verdeutlicht ein Beispiel die Funktionsweise am besten:

```
$[authors a$
  Die $numberOfAuthors$ Autoren sind: $a.name$, $a.name$.
$]authors[$
  Autor unbekannt.
$authors]$
```

Hier würde bei einer leeren Autorenliste stattdessen der Text, daß der Autor unbekannt ist, produziert werden. An diesem Beispiel wird eine weitere sinnvolle Eigenschaft deutlich: Der erste Beispieleintrag der Liste beginnt erst mit dem ersten qualifizierten Platzhalter der Liste, so daß der einleitende Text inklusive eines unqualifizierten Platzhalters vor den eigentlichen Listenelementen genau einmal produziert wird und im Falle der leeren Liste ganz unterdrückt wird. Dies erspart komplexere Konstruktionen aus bedingten Platzhaltern und Listenplatzhaltern. Ein willkommener Nebeneffekt ist, daß durch dieses Verhalten die Erzeugung von Tabellen in HTML außerordentlich leicht gemacht wird, was das abschließende Beispiel zeigt:

```
$[authors a$
  <table>
    <th><td>Autoren:</td></th>
    <tr><td>$a.name$</td></tr>
    <tr><td>$a.name$</td></tr>
  </table>
$]authors[$
  Autor unbekannt.
$authors]$
```

Die Kontrolle der Iteration und die Definition der Ersetzungen für die qualifizierten Platzhalter der Liste geschieht durch Programmierung in der der Vorlage zugewiesenen Regel.

Als nächstes wird der Mechanismus zur Nutzung der Platzhalter in den Vorlagen seitens der Regeln beschrieben. Die Spezifikation des Verhaltens der Platzhalter wurde bereits oben vorgenommen; die Herausforderung des Entwurfs ist die neutrale Anbindung an die Programmlogik der Regeln.

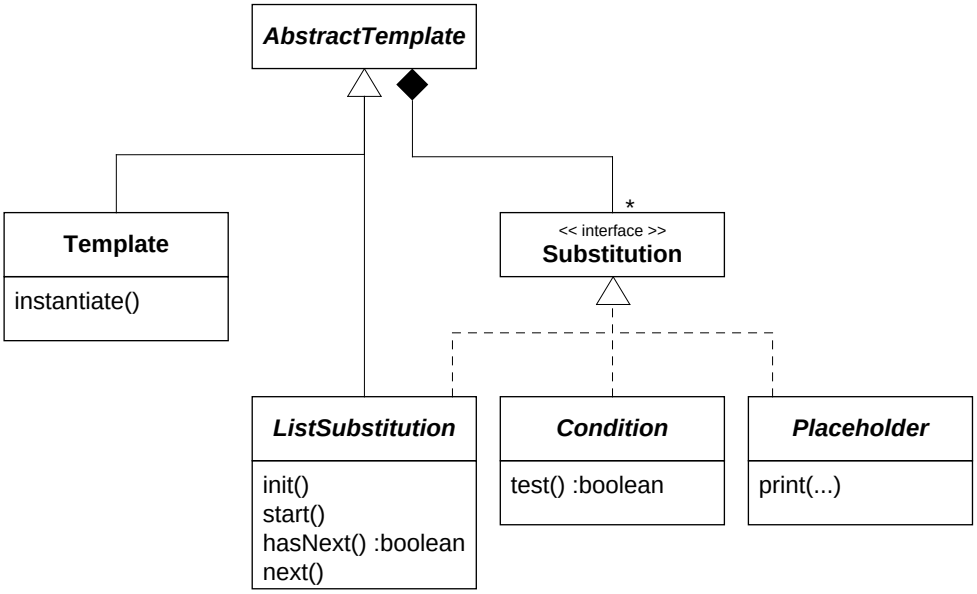

Abbildung 6.14: Klassendiagramm der Vorlagen und Platzhalter

Der Entwurf sieht als zentrales Konzept zur Realisierung der Ersetzung von Platzhaltern die Klassen für Vorlagen (`Template`) und Substitutionen vor. Die genauen Zusammenhänge sind in Abbildung 6.14 in einem konzeptuellen Klassendiagramm dargestellt.

Eine Vorlage wird durch ein Exemplar der konkreten Klasse `Template` repräsentiert. Jedes Mal, wenn eine Regel eine Vorlage nutzen will, muß ein solches Exemplar erzeugt werden. Es wird dann mit Exemplaren vom Typ `Substitution` parametrisiert, die die Ersetzungen der einzelnen Platzhalter definieren. Da sowohl die Vorlage an sich als auch die Listen-Platzhalter (`ListSubstitution`) Substitutionen aggregieren, ist diese Funktionalität in der gemeinsamen, abstrakten Superklasse `AbstractTemplate` herausfaktorisiert. Hier ist zu bemerken, daß auch Bedingungen konzeptuell weitere Ersetzungen aggregieren; da dadurch aber kein neuer Namensraum für qualifizierte Platzhalter aufgespannt wird, kann bei der Bedingung die Möglichkeit zur Aggregation entfallen. Die enthaltenen Platzhalter werden stattdessen einfach flach innerhalb der umgebenden Vorlage oder Liste aggregiert.

Die Implementierungen der Substitutionen sind als abstrakte Klassen ausgeführt. Dies ist so konstruiert, da zur Definition der aktuell in den Regeln an die Vorlage gebundenen Substitutionen ad hoc definierte anonyme innere Klassen benutzt werden, die jeweils Subklassen von `Placeholder`, `ListSubstitution` und `Condition` sind. Die grundsätzliche Verwendung von Vorlagen wird durch ein Beispiel am deutlichsten. Es stellt einen Auszug aus der Methode `handle()` einer Regel dar, die zur Bearbeitung einer Anfrage dient:

```
...
public void handle(Session session) {
  ...
  Template template = new Template("persons/123", session);
  ...
  template.instantiate();
  ...
}
...
```

Zwischen der Erzeugung des Vorlagenobjekts und seiner Auswertung durch `instantiate()` müssen entsprechend der Vorlage Ersetzungen gebunden werden. Die Verwendung aller drei Arten gestaltet sich wie folgt:

1. Im Falle von einfachen, skalaren Platzhaltern muß ein Exemplar einer Subklasse von `Placeholder` an die Vorlage unter Angabe des Namens des Platzhalters gebunden werden. Dies geschieht typischerweise durch die gleichzeitige Definition einer inneren, anonymen Klasse, die Instantiierung eines Objekts dieser Klasse und die Bindung an das Vorlagenobjekt mit der Methode `put()`. Dabei muß der Name des Platzhalters angegeben werden, wobei die Begrenzungszeichen '`$`' entfallen müssen. Wie in dem Klassendiagramm in Abbildung 6.14 ersichtlich ist, wird die Methode `print()` verlangt, die genau den durch den Platzhalter zu ersetzenden Text liefert. Die Ausgabe erfolgt dabei in einen `PrintStream`. Eine beispielhafte Verwendung, wie sie in dem obigen Beispiel eingefügt werden könnte, sähe folgendermaßen aus:

   ```
   ...
   final Person person = session.getServices().getPersons().getPerson(...
   ...
   template.put("name", new Placeholder() {
     public void print(Session session, PrintStream p) {
       p.print(person.getName());
     }
   });
   ...
   ```

   Im Prinzip handelt es sich bei dem gebundenen Platzhalter um eine Funktion höherer Ordnung; in JAVA läßt sich dieses Konzept allerdings nicht direkt ausdrücken, sondern muß über anonyme innere Klassen simuliert werden. Aufgrund der Sprachdefinition von JAVA ist es erforderlich, alle innerhalb der Definition der inneren Klasse benutzten Variablen des umgebenden Sichtbarkeitsbereichs als **`final`**, d.h. unveränderbar nach der ersten Wertzuweisung zu deklarieren (siehe dazu [GJS96]). Dies trifft für die Variable `person` zu, die ein mit der Vorlage anzuzeigendes *Asset* vom Typ Person referenziert.

2. Bedingte Platzhalter werden ähnlich wie skalare Platzhalter gebunden, nämlich ebenso über Exemplare von ad hoc definierten inneren anonymen Klassen. Diese müssen Subklassen der abstrakten Klasse `Condition` sein und die Methode `test()` implementieren, die die eigentliche Entscheidung über das Zutreffen der Bedingung fällt. Eine beispielhafte Verwendung wäre folgende:

   ```
   ...
   template.put("isAdmin", new Condition() {
     public boolean test(Session session) {
       return person.getFirstName().equals("Holm");
     }
   });
   ...
   ```

   Wie hier gut zu sehen ist, wird lediglich ein Wahrheitswert geliefert; die in der Vorlage innerhalb der Bedingung vorhandenen weiteren Platzhalter werden ebenso wie die Bedingung selbst direkt an das `Template`-Objekt gebunden.

3. Listen stellen die komplexeste Art der Ersetzung dar, da hier eine Iterationskontrolle nötig ist und ferner die Namensräume für qualifizierte Ersetzungen aufgebaut werden müssen.

Genau wie bei den beiden vorher beschriebenen Arten wird auch hier für die Liste als ganzes ein Objekt einer ad hoc definierten inneren anonymen Klasse unter Angabe des Namens der Liste an das Vorlagenobjekt gebunden. Dazu müssen diese Klassen Subklassen von `ListSubstitution` sein. Es wird dabei die Implementierung von vier Methoden gefordert, die bei der Initialisierung und der Steuerung der Iteration verschiedene Aufgaben haben:

(a) Die `init()`-Methode wird direkt nach der Bindung des neu erzeugten Objekts genau einmal aufgerufen. Hier können weitere Substitutionen definiert und gebunden werden, die für qualifizierte Platzhalter innerhalb der Liste gelten sollen. Sie dürfen dazu nicht an das umgebende Vorlagenobjekt gebunden werden, sondern müssen an das `ListSubstitution`-Objekt selbst gebunden werden. Dazu erbt es wie das `Template` selbst von der abstrakten Klasse `AbstractTemplate`. Die Bindung weiterer Ersetzungen geschieht hier stets ohne die Qualifikation, da diese auf dieser Ebene bereits durch die Bindungshierarchie ausgedrückt wird; das unten angegebene Beispiel verdeutlicht dies.

(b) Die `start()`-Methode wird zu Beginn der Iteration über die Listenelemente einmal aufgerufen. Hier wird der inneren Klasse Gelegenheit gegeben, einen eventuell selbst zur Iteration benötigten Iterator oder eine Liste aufzubauen. Diese Methode ist von der Initialisierung getrennt, da es so ermöglicht wird, die gleiche Listenersetzung in einer Vorlage mehrfach zu verwenden. Für jedes Auftreten der Liste in der Vorlage würde die Iteration separat wiederholt, also erneut mit `start()` initiiert. Die Initialisierung mit `init()` erfolgt allerdings nur genau einmal.

(c) Die `hasNext()`-Methode hat genau wie in dem Iterator der JAVA-Klassenbibliothek die Aufgabe, seiteneffektfrei zu prüfen, ob noch ein weiteres Element in der Liste vorliegt. Nur dann, wenn diese Methode **`true`** liefert, darf die folgende Methode `next()` verwendet werden; andernfalls ist mit Fehlern und unerwarteten Ausnahmen zu rechnen. Zu Beginn der Iteration liefert die `hasNext()`-Methode die Aussage, ob die Iteration überhaupt Elemente liefert.

(d) Die `next()`-Methode bringt die Iteration um genau einen Schritt weiter. Sie wird vor Verwendung des allerersten Elements aufgerufen, wenn die Iteration nicht leer ist. Zu beachten ist, daß diese Methode keinen Rückgabewert besitzt; sie dient lediglich der von der Vorlagenmechanik gesteuerten Weiterschaltung der Iteration. Die Verwaltung der Elemente, die die Liste ausmachen, muß gänzlich innerhalb der inneren Klasse erfolgen.

Ein Beispiel, das an das weiter oben gegebene Beispiel der Vorlage zur Anzeige von Autoren – etwa eines Dokuments – anknüpft (Seite 179), verdeutlicht den typischen Aufbau einer Listenersetzung:

```
...
final Document doc = ...;
...
template.put("authors", new ListSubstitution() {
  Iterator i;
  Person a;
  public void init() {
    this.put("name", new Placeholder() {
      public void print(Session session, PrintStream p) {
        p.print(a.getName());
      }
    });
```

```
    }
    public void start() {
      i = doc.getAuthors();
    }
    public boolean hasNext() {
      return i.hasNext();
    }
    public void next() {
      a = (Person) i.next();
    }
  });
  ...
```

In diesem Beispiel ist gut zu erkennen, daß die Bindung der Ersetzung an den qualifizierten Platzhalter `$a.name$` aus der Vorlage mit Angabe des unqualifizierten Namens (`"name"`) geschieht, und zwar in der Methode `init()` direkt an die Listenersetzung selbst. Des weiteren ist die Art der Kapselung eines in der `start()`-Methode ermittelten Iterators deutlich zu sehen: Zur Speicherung des Iterators und insbesondere des jeweils aktuellen Elements der Liste, einer Person also, dient ein normales Attribut der inneren Klasse, auf das deswegen alle Methoden der inneren Klasse und insbesondere die hier – doppelt verschachtelte – als innere Klasse der inneren Klasse definierte einfache Ersetzung Zugriff haben. Durch diese Konstruktion lassen sich wieder recht einfach Listen innerhalb von Listen definieren.

Die Entwurfsentscheidung, die Ersetzungen (Platzhalter etc.) als Funktionen höherer Ordnung zu binden, bietet – verglichen mit der Alternative, die Ersetzungswerte direkt unter Angabe des Platzhalternamens dem Vorlagenobjekt bekanntzumachen – diverse Vorteile, die kurz diskutiert werden:

- Die Ausführung der Methoden der gebundenen Objekte (also der Funktionen höherer Ordnung) ist nicht signifikant aufwendiger als die direkte Bindung der resultierenden Werte, da die Berechnung des Wertes in jedem Fall stattfinden muß.
- Ein wirklicher Laufzeitvorteil entsteht, wenn es relativ generische, reichhaltige Ersetzungen anbietende Regeln gibt, die von den Gestaltern der Vorlagen so eingesetzt werden, daß nur einige den aktuellen Bedürfnissen entsprechende Ersetzungen verwendet werden: Hier werden dann tatsächlich nur diejenigen Ersetzungen berechnet, die auch in der Vorlage substituiert werden müssen. Das gleiche Argument gilt für bedingte Passagen, da bei deren Weglassung die innerhalb des weggelassenen Bereichs vorhandenen Platzhalter nicht berechnet zu werden brauchen.
- Die mehrfache Verwendung desselben Platzhalters an verschiedenen Stellen einer Vorlage resultiert in der mehrfachen Berechnung der Funktion. Dies kann u.U. ein Vorteil sein, um jedesmal einen anderen Wert zu produzieren. Ist dies nicht gewünscht, so kann die Ersetzung den einmal berechnete Wert ohne Aufwand zwischenspeichern und bei erneuter Anfrage rasch wieder ohne Neuberechnung ausgeben.
- Ein gewichtiges Argument ist, daß sich mit Funktionen höherer Ordnung die Iterationsabstraktion wesentlich einfacher und effizienter darstellen läßt als mit einer reinen Wertübergabe. In diesem Fall müssten zunächst alle Listenwerte berechnet werden und als komplexe Datenstruktur an das Vorlagenobjekt übergeben werden. Gerade der Fall von verschachtelten Listen, der mit Funktionen höherer Ordnung außerordentlich elegant lösbar ist, ließe sich andernfalls nur mit komplexen Datenstrukturen bewältigen.

Zudem – und das ist das wirklich entscheidende Argument – ist die gewählte Lösung bei Listen wesentlich effizienter, da das Aufbauen von unter Umständen größeren Mengen von Zwischenergebnissen vollständig entfällt. Im Idealfall kapselt die Listenersetzung einen Iterator der Dienstschicht, der seinerseits einen Iterator der Speicherungsschicht kapselt, der wiederum direkt einen *Cursor* zum Durchlaufen der Ergebnismenge einer Datenbankabfrage umschließt, was insgesamt eine optimale Speichernutzung bedeutet. Wird zusätzlich das aus der Vorlage und den Ersetzungen generierte Erzeugnis direkt (etwa über eine Netzwerkverbindung für HTTP) an den Kunden ausgeliefert, entstehen außer Zustandsvariablen und einigen Pufferbereichen mit konstanter Größe keinerlei weitere temporäre Daten. Diese Überlegungen sind deshalb wichtig, um die Anforderungen, die aus Szenarien mit hoher Last (etwa 100 gleichzeitigen Anfragen) entstehen, bewältigen und so die hohe Skalierbarkeit des Systems ermöglichen zu können.

Abschließend wird die Realisierung der Ausgabe-Erzeugung aus Vorlagen und Ersetzungen vorgestellt. Als wichtigste Anforderung wurde bereits genannt, daß die Verwendung verschiedenartiger Vorlagen durch die Regeln transparent zu erfolgen habe. In Abhängigkeit von dem jeweiligen Protokoll und Medium, das von der Kommunikationsschicht für die Anfragen und Antworten darauf verwendet wird, kommen unterschiedliche Arten von Vorlagen zum Einsatz: Für HTTP-basierte Kommunikation kommen in HTML formulierte Vorlagen zum Einsatz, für WAP-basierte Kommunikation mit Mobiltelephonen WML, für Kommunikation per e-Mail einfache Textdateien bzw. HTML-Dateien und für speziellere Zwecke können Textverarbeitungsdokumente verwendet werden.

Der wesentliche Entwurfsgedanke ist nun, in den Regeln uniform als neutrale Schnittstelle das `Template` zu verwenden, die wesentlichen Arten von zu manipulierenden Vorlagen zu identifizieren und Strategien zur Behandlung jeder Art durch konkrete Klassen zu implementieren. Diese Strategie-Objekte werden von dem jeweiligen `Template` in Abhängigkeit von den jeweiligen Anforderungen der Kommunikationsschicht gekapselt und verwendet. Der Entwurf dafür ist in Abbildung 6.15 in einem Klassendiagramm illustriert und entspricht dem Entwurfsmuster der Strategie und des Erbauers nach [GHJV95].

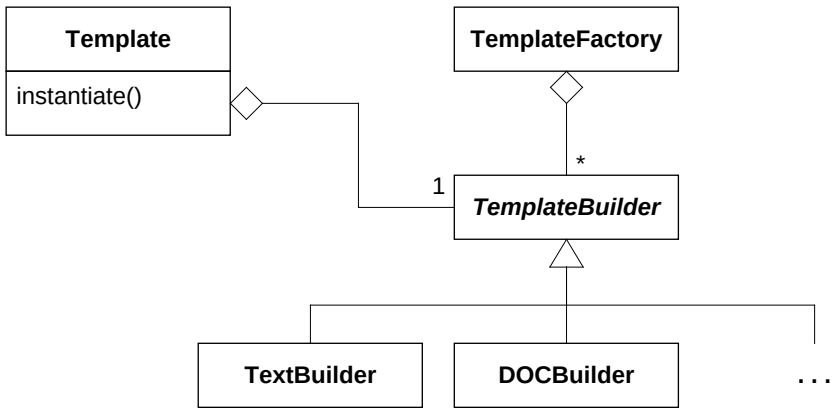

Abbildung 6.15: Klassen für die Instantiierung von Vorlagen

Ein `Template`-Objekt aggregiert ein Objekt einer Subklasse der abstrakten Klasse `TemplateBuilder`. Da die meisten Arten von Vorlagen textbasiert sind (HTML-, WML- oder reine Textdateien), ist die Ersetzungsstrategie der Platzhalter für alle diese Arten die gleiche. Sie wird in der konkreten Klasse `TextBuilder` realisiert. Eine andere Strategie wäre die Nutzung von MS WORD-Dateien (Klasse `DOCBuilder`). Weitere Strategien sind denkbar.

Das Vorlagenobjekt aggregiert nun sowohl die von der Regel erzeugten Ersetzungen als auch ein Strategieobjekt, das es von einer Fabrik, der `TemplateFactory`, zu Beginn der Instantiierung via `instantiate()` anfordert. Die Fabrik erzeugt dann in Abhängigkeit von sitzungsspezifischen Informationen die zu dem jeweiligen Server-Modul passende Strategie, indem die Erzeugung an das Server-Modul delegiert wird. Die Fabrik ist dann frei, die Strategieobjekte beliebig wiederzuverwenden. Das Strategieobjekt entscheidet ferner, welche konkrete Vorlage verwendet wird: da auf der Ebene der Regel der Name der Vorlage beim Erzeugen nur abstrakt in einer an die *Asset ID* angelehnten Syntax angegeben wird (siehe obiges Beispiel: `"persons/123"`), liegt die Verantwortlichkeit zum Lokalisieren der konkreten Datei bei der Strategie. Hier kann in Abhängigkeit vom konkreten Endgerät jeweils eine andere Vorlage ausgewählt werden, um etwa sowohl Web-Browser via HTML als auch Mobiltelefone via WML zu bedienen, oder um Web-Browser-spezifische Unterschiede zu berücksichtigen. Da die konkreten Strategien in die Kommunikationsschicht fallen, werden diese Einzelheiten im nächsten Abschnitt besprochen.

6.6 Die Kommunikationsschicht

Die Kommunikationsschicht als die direkt oberhalb der Interaktionsschicht angesiedelte oberste Schicht der Serverseite des Portalsystems hat die Aufgabe, mit den Clients bzw. Aktoren als Interaktionspartnern die protokoll- und medienabhängige Kommunikation zu führen und dazu die Anfragen für die Interaktionsschicht neutral bereitzustellen sowie deren Antworten zu übermitteln. Die Entwurfsziele für diese Schicht sind:

- Die Schaffung einer Infrastruktur, die die Erweiterung der Kommunikationsschicht um weitere Komponenten für neue Protokolle und Medien gut unterstützt, ohne daß die darunterliegende Interaktionsschicht verändert werden muß.
- Ein Entwurf, der die effiziente Realisierung eines als Mindestanforderung geltenden Web-Servers gestattet, also einer Komponente, die über HTTP kommuniziert und die dort gängigen Austauschformate (HTML, Bilder etc.) zu benutzen in der Lage ist. Diese Komponente soll zugleich die Anforderungen abdecken, die aus der Verwendung von WAP bzw. WML für mobile Endgeräte resultieren, da diese auf einer Teilstrecke ebenfalls HTTP verwenden.
- Bereitstellung von Strategien für die Verwendung von verschiedenen Arten von Vorlagen durch die Interaktionsschicht. Hier ist zumindest die Unterstützung der Realisierung einer konkreten Lösung für textbasierte Vorlagen nötig, um wenigstens HTML-Vorlagen nutzen zu können.

Gerade durch den letzten Punkt sind die Entwürfe der Interaktionsschicht und der Kommunikationsschicht verhältnismäßig eng miteinander verknüpft, so daß eine eher breite Schnittstelle zwischen beiden Schichten erforderlich wird. Im folgenden wird zunächst die Struktur der Schicht beschrieben, um dann detailliert auf die Realisierung der Strategien zur Erreichung von relativer Geräteunabhängigkeit einzugehen. Eine Vorstellung der Web-Server-Komponente und ihrer Konfigurationsmöglichkeiten schließt sich an.

6.6.1 Aufgaben und Struktur der Kommunikationsschicht

Die Kommunikationsschicht besitzt als zentrale Instanz ein *Singleton* der Klasse `Broker`, die sämtliche weiteren Komponenten aggregiert und verfügbar macht. Der Name der Klasse ist eher historisch bedingt und spiegelt die Tatsache wider, daß hierüber zwischen sämtlichen weiteren

Instanzen des Systems vermittelt wird. Außerdem ist dieses Objekt das erste, das beim Starten des Systems erzeugt wird; ihm obliegt daher der Aufbau und die Initialisierung aller Schichten. Dazu gehören im wesentlichen:

- Einlesen der Konfigurationsdateien und ihre Bereitstellung für alle Schichten des Gesamtsystems;
- Erzeugung und Verwaltung von Protokolldateien, die von den verschiedenen Schichten und Komponenten verwendet werden, um Fehler und Vorkommnisse festzuhalten;
- Erzeugung aller Komponenten der Kommunikationsschicht (Server-Module);
- Erzeugung und Initialisierung aller benötigten Manager der Speicherungsschicht;
- Instantiieren und Initialisieren der Dienstschicht; dazu muß im wesentlichen die dort zentrale Instanz der `Services` erzeugt werden, die dann alle weiteren *Asset Container* aufbaut.

Die Konfiguration aller dieser Teile geschieht über eine zentrale Konfigurationsdatei, die die Einstellungen für alle weiteren Schichten und Komponenten enthält. Der Vorteil verglichen mit mehreren verschiedenen Konfigurationsdateien für die verschiedenen Schichten und Komponenten liegt in der deutlich einfacheren Pflege, da die immer vorhandenen funktionalen Abhängigkeiten so an einem Platz definiert werden. Zudem tendieren umfangreiche Konglomerate von zusammengehörigen Dateien dazu, undokumentierte Abhängigkeiten zu entwickeln und so die Installation auf neuen Systemen unter geänderten Rahmenbedingungen erheblich zu erschweren; die Anpassung wird dabei häufig stark verzögert.

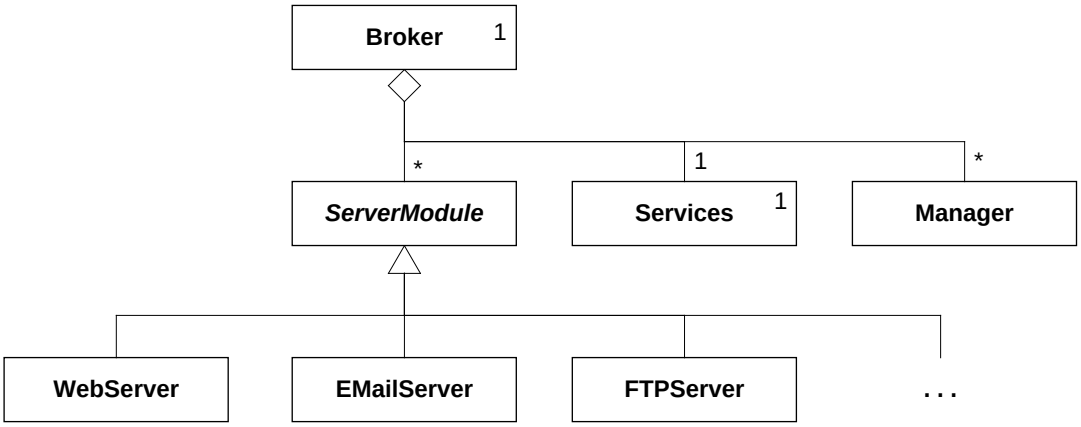

Abbildung 6.16: Komponenten der Kommunikationsschicht

Alle wesentlichen Komponenten der Kommunikationsschicht sind in einem Klassendiagramm in Abbildung 6.16 dargestellt. Als wichtigste weitere Komponenten der Kommunikationsschicht sind die einzelnen Server-Module anzusehen, die jeweils bestimmte Protokolle zur Interaktion mit Partnern realisieren (HTTP, WebDAV, SMTP, POP, FTP, SMB etc.). Eine dieser Komponenten ist ein Web-Server. Es können von jeder Art von Server-Modul mehrere Instanzen gleichzeitig betrieben werden. Details des Web-Servers werden in Abschnitt 6.6.3 erläutert. Die Kopplung mit der Interaktionsschicht zur Erreichung der Geräteunabhängigkeit wird im folgenden Abschnitt 6.6.2 beschrieben.

6.6.2 Strategien für die Geräteunabhängigkeit

Um auf der Ebene der Interaktionsschicht Geräte- und Medienunabhängigkeit zu erreichen, wurden dort die Konzepte der Regeln, der Vorlagen und Ersetzungen eingeführt. Dort wird völlig davon abstrahiert, welche Art von Vorlage und welche Vorlage im einzelnen verwendet wird; die Vorlagen werden dazu über symbolische Namen referenziert. Die Auflösung der Namen und Abbildung auf konkrete Dateien geschieht in Abhängigkeit vom konkreten Server-Modul und dem damit bedienten Endgerät. Auf dieser Ebene sind also genau zwei Aufgaben zu lösen:

1. Realisierung einer Auswahlstrategie für konkrete Vorlagen
2. Angabe der zu benutzenden Implementierung zur Auswertung der Vorlagen

Der Grundgedanke für die Realisierung der Geräteunabhängigkeit ist zunächst ganz einfach: Im Prinzip müsste jede Vorlage, die benötigt wird, für jedes Gerät entsprechend angepaßt existieren, soweit dies zumindest für die Bereitstellung der gewünschten Funktionalität pro Gerät erforderlich ist. Im Falle von HTML als Vorlagenart, die verwendet wird, um Web-Browser zu bedienen, kommt man normalerweise trotz aller Unterschiede zwischen den verschiedenen angebotenen Web-Browsern tatsächlich mit einem Satz von Vorlagen aus, wenn diese entsprechend sorgfältig gestaltet werden.

Ganz anders stellt sich die Situation dar, wenn mobile Endgeräte wie Mobiltelefone (sogenannte WAP-*Handys*), persönliche digitale Assistenten (PDA) etc. bedient werden sollen. Zwar steht eine umfangreiche Spezifikation der Protokolle zum mobilen Zugriff auf Informationsressourcen in Form des *Wireless Access Protocols* (WAP) zur Verfügung, das eine umfangreiche Definition der für die Inhalte zu verwendenden Auszeichnungssprache WML (*Wireless Markup Language*, ein XML-Derivat) enthält [WML00], jedoch ergab eine systematisch im Rahmen von [Bar01] vorgenommene Untersuchung der Implementierung dieser Standards durch im Handel erhältliche mobile Endgeräte ein anderes Bild: Da insbesondere der WML-Standard viele gravierende Spezifikationslücken enthält und zusätzlich die Realisierung diverser Eigenschaften explizit den Geräteherstellern überläßt, ist es praktisch unmöglich, mit einem Satz von WML-Vorlagen alle Geräte zu bedienen.[4] Der Aufwand, alle Vorlagen für alle Geräte in jeweils für das Gerät optimierter Darstellung zu produzieren, ist prohibitiv hoch, da mittlerweile Dutzende von Geräten auf dem Markt sind. Die Lösung dieser Problematik ist in einer zweigeteilten Strategie zu sehen:

- Erstens kann versucht werden, eine möglichst große Schnittmenge an von allen Geräten mit vernünftigen Ergebnissen darstellbaren Eigenschaften zu identifizieren und dafür eine Sammlung von Gestaltungsregeln für WML zu verfassen. Diese wären als Leitlinien für die Gestalter der Benutzungsoberfläche zu verwenden und haben keinen direkten Einfluß auf den softwaretechnischen Entwurf.
- Zweitens muß versucht werden, eine flexible Strategie zur Nutzung der Vorlagen zu finden. Dem liegen die Beobachtungen zugrunde, daß einerseits größere Gruppen von Geräten ein ähnliches Verhalten aufweisen, und daß sich andererseits die nach dem ersten Schritt verbleibenden Unterschiede häufig auf nur einige wenige Vorlagen beschränken. Die Ausgestaltung dieser Strategie muß durch den Entwurf geschehen.

Der erste Teil der Lösung wurde – obwohl es keinen direkten Einfluß auf den Entwurf des restlichen Systems und insbesondere den zweiten Teil hat – durch die Definition der oben geforderten

[4] Nicht zuletzt wegen dieses als äußerst schwach zu bezeichnenden Standards ist vermutlich der erwartete Erfolg der anfangs mit größter Euphorie begrüßten WAP-Dienste bisher ausgeblieben.

minimalem Menge von sinnvoll auf allen Geräten darstellbaren Eigenschaften durch ein XML-Dialekt unterstützt. Diese Definition ist stark an WML angelehnt und konkrete Vorlagen können durch entsprechende XSL-Transformationen leicht in die jeweiligen gerätespezifischen Formate überführt werden [Bar01, Cla99]. Ein ähnlicher Ansatz wird in [Kam01] zum *Cross media publishing* beschrieben.

Die naheliegende – durch Anwendung der objektorientierten Prinzipien gefundene – Strategie für den zweiten Teil der Lösung ist in der Schaffung einer flexiblen, hierarchischen Klassenstruktur für die Endgeräte zu sehen, in die jedes zu unterstützende Gerät eingeteilt wird. Durch Generalisierung und Spezialisierung innerhalb der Klassenhierarchie lassen sich alle Unterschiede im Verhalten der Geräte und somit in den Vorlagen effizient mit einem Minimum an konkret benötigten Vorlagen beschreiben. Die Klassenstruktur für die Geräte wird nun nicht direkt durch Klassen in JAVA modelliert, sondern durch zur Laufzeit aus Konfigurationsdaten aufgebauten Objektgraphen. Diese Objekte sind Exemplare bestimmter generischer Klassen, die in diesem Sinne Metaklassen der Geräteklassen sind.[5] Diese Klasse zur Beschreibung der Geräteklassen ist die in Abbildung 6.17 gezeigte Klasse `UserAgentClass`; die einzelnen Gerätetypen werden durch Exemplare der Klasse `UserAgent` modelliert. Die Exemplare der ersten Klasse lassen sich rekursiv aggregieren, um die Hierarchie der Geräteklassen auszudrücken. Dabei darf es mehrere Wurzeln, d.h. Objekte ohne Elternknoten geben. Die Gesamtheit aller Geräteklassen wird durch eine Fabrik (`UserAgentFactory`) verwaltet und beim Start des Systems von ihr aufgebaut.

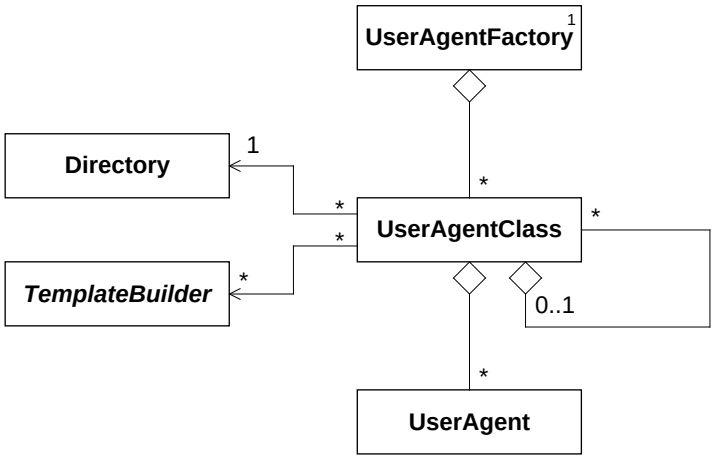

Abbildung 6.17: Metaklassen zur Beschreibung der Geräteklassen

Zur Anfragezeit muß jedes Server-Modul jeder Anfrage ein Gerät durch ein Exemplar der Klasse `UserAgent` zuordnen, damit die im folgenden beschriebenen Strategien zur Auffindung der Vorlagen etc. korrekt durchgeführt werden können. Die Geräteexemplare werden dabei ebenfalls von der Fabrik verwaltet und herausgegeben. Hier bietet sich innerhalb der Fabrik das Entwurfsmuster des Fliegengewichts (*Flyweight*) nach [GHJV95] zur Realisierung der Geräteobjekte an.

Jede Geräteklasse hat ein assoziiertes Verzeichnis, innerhalb dessen und seiner Unterverzeichnisse die jeweils benötigten Vorlagen für Geräte dieser Klasse abgelegt und gesucht werden. Als Verzeichnis wird ein Verzeichnis der Dienstschicht verwendet; damit sind auch die Vorlagen reflektiv über das Portalsystem selbst pflegbar und lassen sich dafür dem Berechtigungskonzept des Portals unterwerfen. Die Auflösungsstrategie der Vorlagen ist nun, bei Anforderung einer Vorlage durch ein `Template`-Objekt zunächst in dem der Klasse zugeordneten Verzeichnis zu suchen; wenn die Suche dort erfolglos ist, wird im Verzeichnis der Superklasse weitergesucht. Dieser

[5] Dies ist dann vergleichbar mit den Metaklassen in TYCOON-2 [GW98]

Algorithmus muß innerhalb der Klasse `UserAgent` implementiert werden. Als zweite wichtige Eigenschaft einer Geräteklasse ist die Verbindung zu einer konkreten `TemplateBuilder`-Subklasse zu sehen. Diese muß – eventuell in Abhängigkeit von der angeforderten Vorlage – von der Geräteklasse bei jeder Instantiierung einer Vorlage angegeben werden. Zusammen mit der gemäß des oben beschriebenen Suchalgorithmus gefundenen Vorlagendatei kann dann die Ersetzung der Platzhalter der Vorlage durchgeführt werden.

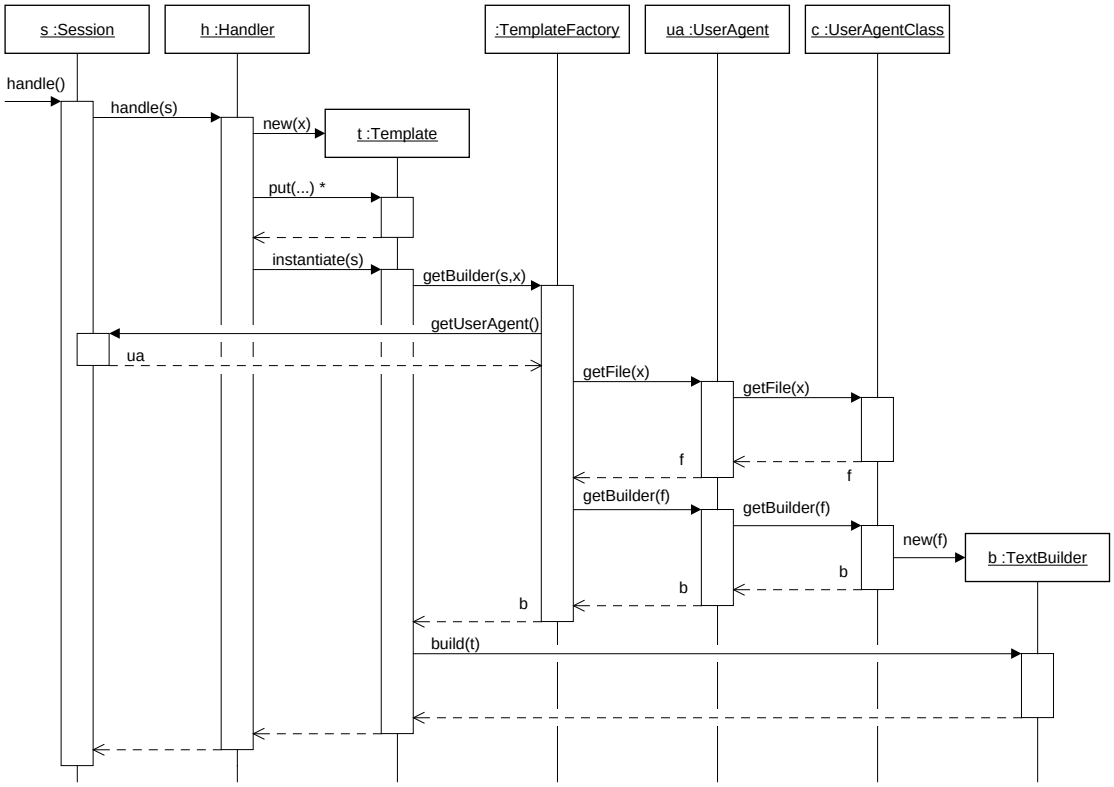

Abbildung 6.18: Interaktionen bei Instantiierung einer Vorlage

Das genaue Zusammenspiel aller Komponenten, insbesondere der Geräteklassen und Vorlagen, ist in Abbildung 6.18 durch ein Sequenzdiagramm in UML beschrieben. Hier ist zunächst die Nutzung der Vorlage `t` durch die Regel `h` – ähnlich wie in Abbildung 6.12 – zu erkennen. Wichtig sind dann die Interaktionen, die auf die Instantiierung der Vorlage `t` durch `instantiate()` folgen: Zunächst besorgt sich die Vorlage bei der `TemplateFactory` einen `TemplateBuilder` (vgl. auch Abbildung 6.15), um diesen dann zur Generierung der Ausgabe via `build()` zu verwenden. Die Fabrik bemüht ihrerseits zunächst die Sitzung `s`, um das aktuelle Endgerät in Erfahrung zu bringen. Dieses muß von dem Server-Modul bei jeder Sitzung entsprechend eingetragen werden, wozu die in Abbildung 6.17 erwähnte `UserAgentFactory` verwendet werden kann. An das Endgeräteobjekt `ua` wird sodann die Ermittlung der konkreten Vorlage delegiert, und daraufhin unter Angabe der Vorlage nochmals die Ermittlung der Ersetzungsstrategie. Hier steht der Fabrik die Möglichkeit offen, bereits einmal erzeugte Strategieobjekte in einem Zwischenspeicher zu behalten, um die zukünftigen Zugriffe zu beschleunigen (im Sequenzdiagramm nicht gezeigt). Dies ist gerade bei der Ersetzungsstrategie für Textdateien sinnvoll, um die erneute, relativ aufwendige Analyse der Positionen von Platzhaltern zu vermeiden. Dazu kapselt jedes Exemplar von `TextBuilder` den Ableitungsbaum, der zur syntaktischen Analyse der Platzhalterstruktur generiert wird. Aus diesem Ableitungsbaum kann dann ohne Zeitverlust

beliebig häufig die Vorlage instantiiert werden. Tatsächlich merkt sich die aktuelle Implementierung sogar in einem dadurch sehr kompakten Ableitungsgraphen nur die Positionen der Platzhalter und statischen Textfragmente innerhalb der Datei, um diese dann rasch wahlfrei einlesen und wieder ausgeben zu können; dies vermeidet, daß größere Vorlagen im Zwischenspeicher zu Hauptspeicherengpässen führen.

Die Ergebnisse des Ersetzungsprozesses werden in allen Fällen über Ausgabeströme (`OutputStreams`) zurück zum Server-Modul geleitet. Dank der relativ reichhaltigen Ausstattung der JAVA-Klassenbibliothek für die Ein- und Ausgabe sowie der Konstruktion der Ausgabeströme nach dem Entwurfsmuster der Dekoration [GHJV95] ist deren Verwendung für alle Protokolle in allen Server-Modulen leicht zu bewerkstelligen.

Eine besonderer Vorteil des Konzepts der Listenersetzungen in Verbindung mit endgeräteabhängigen Strategien für den Bereich der Kommunikation mit mobilen Endgeräten soll abschließend kurz dargestellt werden. Aufgrund der technischen Beschränkungen von WAP-fähigen Mobiltelefonen bezüglich der maximalen Größe überhaupt darstellbarer WML-Seiten wird die Generierung noch akzeptabler Ausgaben für diese Klassen von Geräten deutlich erschwert. Besonders problematisch ist dabei, daß die jeweilige Obergrenze von Gerät zu Gerät unterschiedlich ist und sich die Größe der Ausgabe erst durch die auf der Kommunikationsstrecke transparent verwendete Komprimierung ergibt [Bar01, WML00]. Praktisch bedeutet dies insbesondere, daß generierte Listen eine bestimmte geräteabhängige Anzahl an Einträgen nicht überschreiten dürfen. Dabei bestimmt zusätzlich die Größe der jeweiligen Einträge die Anzahl der noch möglichen Einträge. Da die Größe jedes Eintrags nicht von vornherein bekannt ist, muß die Größe einer Liste dynamisch bestimmt werden. Zur Lösung dieser Probleme wurde eine spezielle für Mobilgeräte angepaßte Listenersetzungen (`ListSubstitution`) entworfen und realisiert, die als Dekoration für normale Listenersetzungen benutzt werden kann. Sie macht sich die Tatsache zunutze, daß eine Liste auch mehrfach generiert werden kann und erzeugt zunächst eine leere Liste, um die Größe der die Liste umgebenden Ausgabe zu ermitteln. Diese Ausgabe wird durch die Benutzung eines funktionslosen Ausgabestroms verworfen. Im nächsten Schritt wird dann die Liste schrittweise aufgebaut, wobei durch eine spezielle Dekoration des eigentlichen Ausgabestroms die Größe gemessen wird. Wird durch einen Eintrag die zu erwartende Maximalgröße überschritten, so wird dieser Schritt zurückgenommen, indem die vollständig gepufferte Ausgabedekoration auf den vorletzten Schritt zurückgesetzt wird. Damit wird erreicht, daß die maximale Größe einer WML-Seite nie überschritten wird. Durch Kombination dieser Listenersetzung mit Ersetzungen für Funktionen zum Blättern durch mehrere Ergebnisseiten sind so auch verschiedenste mobile Endgeräte effektiv zu bedienen. Diese speziellen Listenersetzungen arbeiten eng mit den bereits beschriebenen Strategien (`TemplateBuilder`) zusammen und werden von ihr in Abhängigkeit von der aktuellen Endgeräteklasse gewählt. Die einzige Einschränkung dieser Listenersetzung ist, daß pro Vorlage nur eine solche nutzbar ist und daß sie natürlich nicht geschachtelt werden können. Der Gewinn dieser Lösung liegt wiederum darin, daß die Regeln der Interaktionsschicht ohne Änderung auch in diesem Fall benutzt werden können, auch wenn je nach Gerät eine signifikant andere Ausgabe erzeugt wird.

6.6.3 Aufbau und Konfiguration der HTTP-Server

Bereits in Abschnitt 6.1.1 wurde die Entscheidung begründet, die Kernfunktionalität des Portalsystems vollständig und integriert neu zu entwerfen und zu implementieren. Dazu zählen auch die Server-Module und damit der als wichtigstes Modul anzusehende HTTP-Server (Web-Server). Das Protokoll HTTP [FGM+99] selbst ist zumindest in der Version 1.0 nicht sehr aufwendig zu realisieren, aber gibt es einige Punkte, die den Entwurf des HTTP-Servers interessant machen. Wichtige Anforderungen, die an das HTTP-Server-Modul gestellt werden müssen, sind:

- Es müssen gleichzeitig mehrere Instanzen des Moduls betrieben werden können, die sich als nach außen hin unabhängige HTTP-Server erkennen lassen. Jeder muß unter einer eigenen Adresse (*Port*-Nummer) ansprechbar sein.
- Die einzelnen Instanzen der HTTP-Server-Module müssen sich mit Sicherheitseinstellungen konfigurieren lassen, um nur bestimmten Bereichen (Internet-Adressräumen) Zugriff zu gewähren oder um nur bestimmte Bereiche des Portals zugänglich zu machen. Dies ist wichtig für Szenarien, in denen es strikt getrennte Benutzergruppen für unterschiedliche Bereich des Portals gibt, die von innerhalb oder außerhalb der Organisation via Extranet oder Internet auf das Portal zugreifen wollen.
- Der HTTP-Server soll hohe Lasten bewältigen können und geringe Antwortzeiten bieten. Die parallele Bearbeitung gleichzeitig ankommender Anfragen ist also auf jeden Fall nötig. Dies gilt im übrigen für alle Server-Module.
- Die Server-Module müssen eine hohe Stabilität bieten und mit Fehlersituationen der darunterliegenden Schichten umgehen können. Keinesfalls akzeptabel ist der Abbruch des Betriebes durch Fehler in einzelnen Anfragen.

Um diesen Anforderungen gerecht zu werden, wurde der HTTP-Server wie in Abbildung 6.19 durch ein Klassendiagramm dargestellt entworfen.

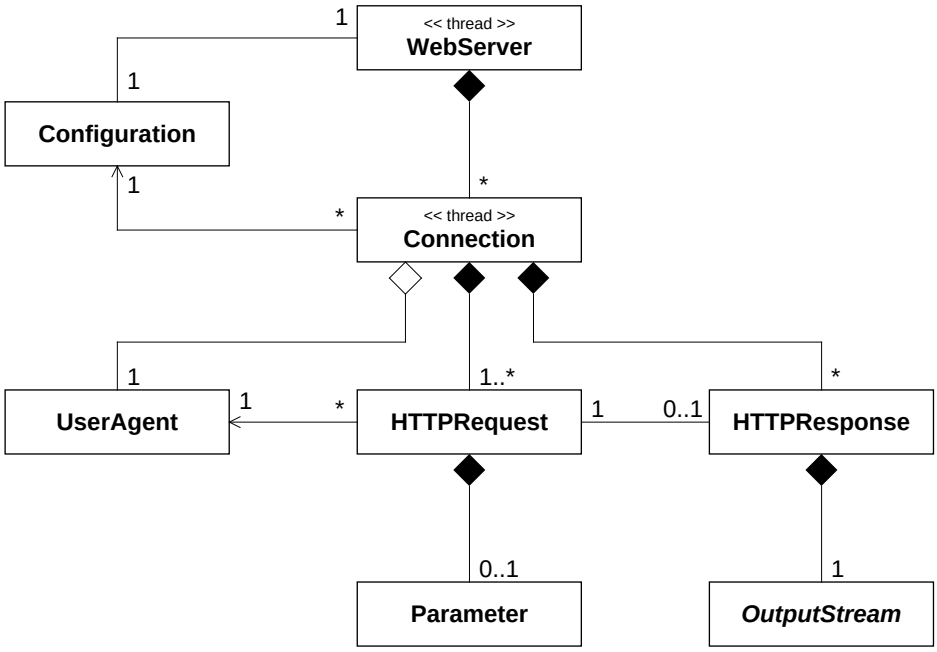

Abbildung 6.19: Entwurf des HTTP-Server-Moduls

Ein HTTP-Servermodul wird durch die Klasse `WebServer` repräsentiert, von der beliebig viele Exemplare existieren können. Sie werden durch Exemplare der Klasse `Configuration` mit Konfigurationseinstellungen versehen. Jeder Server wird als unabhängiger Prozess betrieben (`Thread`). Seine Aufgabe nach der Initialisierung ist es, auf dem konfigurierten *Port* auf HTTP-Anfragen zu warten. Sobald eine Verbindung zustande gekommen ist, instantiiert der Server ein Objekt der Klasse `Connection` und delegiert die Anfrage an dieses. Um mehrere gleichzeitig an eine Instanz des Server-Moduls gerichtete Anfragen parallel behandeln zu können, wird die

Bearbeitung der Anfrage durch die `Connection` als eigenständiger Prozess ausgeführt, so daß der Server sofort wieder reaktionsfähig ist. Von der Verwendung früher üblicher *Thread-Pools*, also der Erzeugung und Verwaltung ausführungsbereiter Prozesse auf Vorrat, wird hier abgesehen, da die modernen virtuellen Maschinen von JAVA diesbezüglich keine deutlichen Engpässe mehr besitzen [ES00].

Das Verbindungsobjekt erzeugt ein Objekt der Klasse `HTTPRequest`, das die Anfrage zu verarbeiten in der Lage ist. Die Anfrageparameter werden dabei in einem neuen Objekt der Klasse `Parameter` als Schlüssel/Wert-Paare gesammelt. Ferner wird das Endgerät des Kommunikationspartners bestimmt. Wenn die Sicherheitseinstellungen der Konfiguration dies zulassen (was normalerweise der Fall ist), wird ein Antwortobjekt der Klasse `HTTPResponse` erzeugt, das die Netzwerkverbindung zum entfernten Partner in einem `OutputStream` kapselt. Erst dann wird von dem Verbindungsobjekt die Anfrage an die Interaktionsschicht geleitet, indem Anfragename, Sitzungskennung, Parameter und Ausgabestrom an die Sitzungsverwaltung übergeben werden. Der von dort an stattfindende Ablauf wurde bereits eingehend beschrieben. Wird vom entfernten Kommunikationspartner das Protokoll HTTP in der leistungsfähigeren Version 1.1 verwendet [FGM+99], so kann die Netzwerkverbindung geöffnet bleiben, um darüber gleich die nächste Anfrage zu entgegenzunehmen. In diesem Fall erzeugt das `Connection`-Objekt neue Exemplare von `HTTPRequest` und `HTTPResponse` und führt den eben beschriebenen Ablauf erneut durch. Die erneute Ermittlung des Endgerätes kann dabei natürlich entfallen.

Der anfangs angesprochenen Forderung nach hoher Performanz wird dadurch nachgekommen, daß jede Anfrage in einem separaten Prozess ausgeführt wird, wodurch eine hohe Parallelität und damit geringe Antwortzeiten erreicht werden. Zusätzlich erforderlich ist natürlich eine effiziente Implementierung besonders der Analyse der HTTP-Anfrage in `HTTPRequest`. Weitere Leistungsvorteile ergeben sich durch die enge Kopplung des Servers an die gesamte restliche Anwendungslogik, also die Interaktionsschicht und die damit verbundenen weiteren Schichten. Die typische Antwortzeit einzelner Anfragen für durchschnittlich komplexe Seiten, also inklusive Regelausführung, Platzhalterersetzung und Aufrufen von Diensten der Dienstschnittstelle liegt fast immer deutlich unter einer halben Sekunde bei Benutzung zeitgemäßer Hardwaresysteme. Experimente mit einer Anbindung der Interaktionsschicht über bestehende Web-Server und *Servlet Engines* statt durch den HTTP-Server Kommunikationsschicht haben durchweg deutlich schlechtere Ergebnisse geliefert.

Die Sicherheitseinstellungen, die durch die Konfiguration getrennt für jedes Server-Modul vorzunehmen sind, lassen neben der Angabe, unter welcher *Port*-Nummer der Server laufen soll, die Einschränkung des Zugangs auf bestimmte Adressbereiche des TCP/IP-Protokolls sowie die selektive Freigabe ausgewählter Bereiche von Anfragen zu. So ließe sich etwa ein Portalsystem betreiben, dessen eine Serverinstanz von außerhalb der Organisation nur von bestimmten Rechnern angesprochen werden kann, wobei inhaltlich nur bestimmte Bereiche zulässig sind, während von innerhalb der Organisation eine andere Serverinstanz (die durch *Firewalls* nach außen vollständig abgeschirmt sein kann) ohne Einschränkungen genutzt werden kann. Für Szenarien des *Content Commerce* ist eine solche Unterscheidung von Zugängen für Kunden, Verkäufer und Redakteure sinnvoll einsetzbar und nötig, um gerade den höheren Sicherheitsanforderungen großer Organisationen nachkommen zu können.

Die technische Ausfallsicherheit durch Abfangen von Fehlersituationen wird einerseits durch die Aufteilung aller Anfragen auf jeweils neue Prozesse erreicht; im schlimmsten Fall wird der fehlerverursachende Prozess vorzeitig beendet, was die anderen Prozesse normalerweise nicht beeinträchtigt. Dennoch ist das gezielte Abfangen von Ausnahmen, die nicht anderweitig behandelt wurden, zumindest auf der Ebene der Verbindung `Connection` sinnvoll; einerseits um sie für die Fehlerbehebung zu protokollieren, und um dem entfernten Kommunikationspartner andererseits wenigstens eine aussagekräftige Fehlermeldung präsentieren zu können.

6.7 Zusammenfassung

Der in den letzten Abschnitten beschriebene Entwurf ermöglicht den Bau eines Portalsystems, das die in Kapitel 4 gestellten Anforderungen durchgängig erfüllen kann. In Abschnitt 4.2 wurden mehrere entscheidende Basisanforderungen formuliert. Diese werden hier wieder aufgegriffen und in Verbindung mit einem Rückblick auf wichtige Entwurfsentscheidungen zur Stützung der obigen Aussage verwendet.

1. Die Trennung von Struktur, Inhalt, Gestaltung und Funktionalität der Informationsartefakte wurde durch folgende Aufteilung erreicht:
   - **Struktur** = Konzept der *Asset*-Typen auf der Dienstschicht bzw. der Inhaltsbeschreibungen auf der Speicherungsschicht;
   - **Inhalt** = Konzept der *Assets* auf der Dienstschicht bzw. Konzept der Manager, Container und Inhaltsobjekte auf der Speicherungsschicht;
   - **Gestaltung** = Konzept der Vorlagen auf der Interaktionsschicht;
   - **Funktionalität** = Konzept der Trennung von Vorlage und Regel auf der Interaktionsschicht.
2. Die Metaphern von Personen, Dingen, Orten und Begriffen (*People, Places, Things, Concepts*) wurde durch ein reichhaltiges konzeptuelles Modell (vgl. Abbildung 6.6) und entsprechende Schnittstellen und *Asset*-Typen in der Dienstschicht umgesetzt.
3. Durch das Konzept der *Assets* und ihre Umsetzung in der Dienstschicht ist die uniforme Behandlung aller Informationsartefakte ermöglicht worden.
4. Die tiefe Erschließung durch Klassifikation und semantische Vernetzung aller Informationsartefakte ist wiederum durch das Konzept der *Assets* uniform möglich. Insbesondere die Modellierung aller wichtiger Beziehungen als *Assets* bildet u.a. die Grundlage für die Erreichung von Integration und Konsistenz.
5. Die Integration heterogener Systemlandschaften ist durch die die Unterschiede absorbierenden Konzepte der Speicherungsschicht sowie die explizite Modellierung aller Verknüpfungen als *Assets* erst auf der Ebene der Dienstschicht ermöglicht.
6. Die von den Benutzern gewohnte Arbeitsweise wird durch die Möglichkeit der Unterstützung zahlreicher Protokolle in der Kommunikationsschicht erleichtert.

Alle diese wichtigen Anforderungen ließen sich nur dadurch erfüllen, daß der gesamte Kern des Portalsystems vollständig und durchgängig neu entworfen wurde. Die Kopplung separat entworfener und nicht füreinander bestimmter, bereits vorhandener Komponenten hätte wesentliche Brüche in der Modellierung und Zusammenarbeit der Einzelkomponenten bedeutet sowie hohen Anpassungsaufwand verursacht.

Kapitel 7

Benutzungsoberfläche und Einsatzbeispiele

Es ist nicht genug, zu wissen, man muß auch anwenden;
es ist nicht genug, zu wollen, man muß auch tun.
– JOHANN WOLFGANG VON GOETHE, Wilhelm Meisters Wanderjahre

In diesem abschließenden Kapitel sollen einige Projekte vorgestellt werden, in denen die konkrete Realisierung des zuvor entworfenen Wissensportals – also das praktische Ergebnis dieser Arbeit – tatsächlich eingesetzt wird. Ziel dieses Kapitels ist es, nachzuweisen, daß das entworfene System den Anforderungen aus der Praxis standhält und zu zeigen, wie verschiedenartige Problemstellungen mit demselben Kernsystem gelöst werden können. Zudem wird ein Eindruck der bestehenden Benutzungsoberfläche vermittelt und dargestellt, wie diese im einzelnen realisiert ist.

Im ersten Abschnitt 7.1 wird zunächst die generische, allgemein gehaltene Benutzungsoberfläche des Web-Zugangs des Portals beschrieben, die in ähnlicher Form in allen im folgenden geschilderten Projekten verwendet wird. Daran schließt sich in Abschnitt 7.2 ein Erfahrungsbericht über den Einsatz des Wissensportals für die begriffsorientierte Dokumentenverwaltung im internetgestützten Projektmanagement der Finanzbehörde Hamburgs an. Abschließend wird in Abschnitt 7.3 ein in den Bereich des Kompetenzmanagements fallendes System zur Erstellung von Anforderungsprofilen und der automatischen Durchführung von Tests zur Bewerberauswahl vorgestellt.

7.1 Benutzungsoberfläche

Der in Kapitel 6 vorgestellte Entwurf war auf die Serverseite des Portals beschränkt. Auf der Client-Seite wird für den web-basierten Zugriff natürlich primär ein Web-Browser verwendet. Die Erstellung der von der serverseitigen Interaktionsschicht und den Web-Browsern genutzten Vorlagen und der entsprechenden Regeln fällt nicht unter den softwaretechnischen Entwurf im engeren Sinne, sondern bleibt neben der Programmierarbeit sowohl eine gestalterische Aufgabe als auch eine der Softwareergonomie. Noch wichtiger ist, daß sich erst durch diese Ausgestaltung die Benutzbarkeit und viele Funktionen eines Wissensportals erschließen. Ohne eine Beschreibung, wie die in Kapitel 4 geforderten Funktionen auf der Oberfläche realisiert werden können,

wäre die Darstellung unvollständig. Deswegen soll im folgenden ein Ansatz zum Aufbau der Benutzungsoberfläche geschildert werden, der auf dem Gebrauch von HTML und JAVASCRIPT beruht. Dabei werden sowohl die grundsätzliche (optische) Gestaltung als auch die (innere) Funktionsweise der client-seitigen Funktionalität beschrieben.

7.1.1 Gestaltung

Die Gestaltung der Benutzungsoberfläche geschah nach folgenden Grundsätzen:

- Die wesentlichen Bereiche der in Kapitel 4 genannten Metaphern der Personen, Dokumente, Verzeichnisse und Begriffe müssen stets erreichbare Einstiege erhalten. Zudem muß immer erkennbar sein, in welchem Bereich der Benutzer gerade arbeitet. Die Verwendung von *Frames* zur Aufteilung der HTML-Seiten bietet sich dafür an; ein feststehender Navigationsbereich (etwa am oberen oder linken Rand) leistet das eben Geforderte.
- Ebenso müssen Recherchefunktionen stets erreichbar sein; die Einbettung in die Navigationsleiste oder die Nutzung einer eigenen Leiste sind Möglichkeiten dafür.
- Die Funktionen zur Annotation, zur Bewertung und zum Aufnehmen bestimmter *Assets* in Sammelmappen (vgl. Anforderung M.4) sollen sehr leicht durchführbar sein, ohne den Arbeitsfluß zu behindern. Hier ist eine Lösung, die diese Informationen auf der Seite des Clients sammelt, ohne zunächst relativ langwierige Anfragen an den Server zu senden, deutlich vorteilhafter. Allerdings erfordert diese Lösung den Einsatz eines umfassenden JAVASCRIPT-Systems für alle Vorlagen. Andererseits kann dieses auch die ansonsten aufwendige Pflege der innerhalb der Vorlagen zu benutzenden Verknüpfungen (Hyperlinks) deutlich vereinfachen.
- Zur Erleichterung der Navigation in umfangreichen Begriffsnetzen muß ein graphischer Begriffsnavigator eingesetzt werden können. Hier bietet sich etwa ein JAVA-*Applet* an. Ein prinzipiell gut geeignetes Produkt, um die hier verwendeten Begriffsnetze zu visualisieren, ist das Programm THEBRAIN, das auch als *Applet* zur Verfügung steht [The01].

Einen Eindruck von der Umsetzung vermittelt Abbildung 7.1, in der ein Bildschirmabzug der Eingangsseite des am Arbeitsbereich Softwaresysteme der Technischen Universität Hamburg-Harburg verwendeten Wissensportals wiedergegeben ist. Dieses Portal basiert ebenso wie die folgenden Beispiele auf den Ergebnissen dieser Arbeit. Im oberen Bereich ist der angesprochene Begriffsnavigator zu sehen; hier ist gut erkennbar, wie ein Begriff („Java-Klassenbibliothek“) im Kontext seiner Ober- und Unterbegriffe angeordnet ist. Die Navigation und Suche geschieht durch die am linken Rand angeordnete Leiste; gut erkennbar ist das Eingabefeld für Suchbegriffe und die darunter befindliche Auswahlmöglichkeit für die zu durchsuchenden Bereiche (z.B. Dokumente). Im unteren Bereich befindet sich die eigentliche Navigationsstruktur, die hier hierarchisch aufgebaut ist und einen thematisch gegliederten Zugang zu allen Bereichen bietet. Als Begrüßungsseite für angemeldete Benutzer erscheint die dargestellte Seite mit Informationen zum persönlichen Profil, d.h. Zugängen zu allen den Benutzer betreffenden Bereichen wie erstellten Dokumenten, persönlichen Informationen etc. Die Art der Präsentation der eigentlichen Inhalte des Portals wird in den folgenden Beispielen noch näher dargestellt.

7.1.2 Arbeitsweise der client-seitigen Funktionalität

Die client-seitige Funktionalität wird durch JAVASCRIPT realisiert [JS96]. Die Einstiegsseite, die auch die *Frames* definiert, fordert dazu anfangs den gesamten Code vom Server an, der dazu

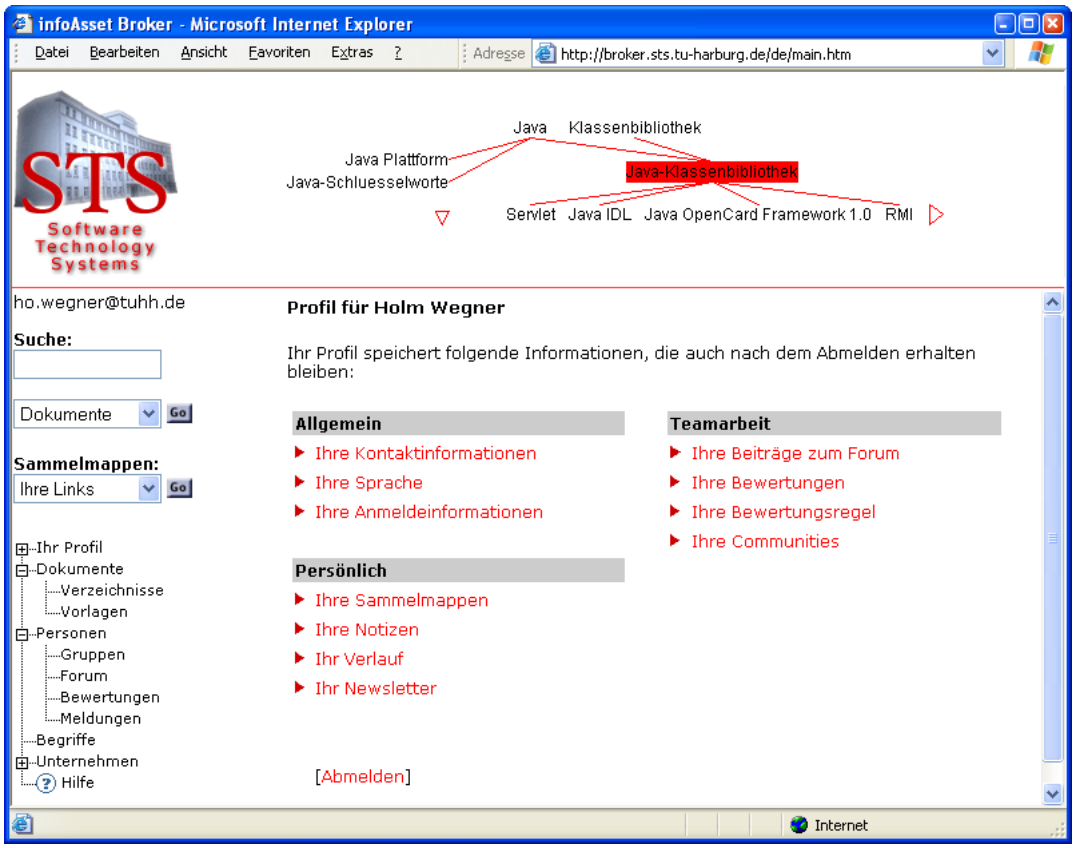

Abbildung 7.1: Benutzungsoberfläche des Portals

in einer Datei zusammengefaßt ist. Im folgenden wird die so realisierte und im Web-Browser ablaufende Funktionalität auch kurz als Client bezeichnet. Die Funktionalität des Clients umfaßt insbesondere die folgenden Punkte:

- Als wichtigste Aufgabe wird von dem Client die aktuelle Sitzungskennung (*Session ID*) verwaltet. Diese wird mit der Antwort auf die initiierende Anfrage vom Server gesendet, vom Client ausgelesen und in einem speziellen *Session*-Objekt vermerkt, das mithilfe von JAVASCRIPT als zusätzliches Attribut des HTML-Dokuments definiert ist. Dieses dynamisch erzeugte und definierte Objekt besitzt weitere Attribute und Funktionen, die überall innerhalb der Vorlagen verwendet werden können.

- Statt normaler Hyperlinks innerhalb einer Vorlagen zu einer anderen Vorlage wird eine JAVASCRIPT-Funktion verwendet, der der logische Anfragename (vgl. Abschnitt 6.5 und 6.6) übergeben wird. Die Funktion ergänzt die Sitzungskennung und weitere Parameter und sendet die so zusammengestellte HTTP-Anfrage ihrerseits ab. Ein typisches Beispiel sieht im HTML-Code folgendermaßen aus:

  ```
  ...
  <a href="javascript:s.show('persons/default.htm')">Personen</a>
  ...
  ```

  Das hier benutzte JAVASCRIPT-Objekt `s` ist genau das oben angesprochene Sitzungsobjekt, das u.a. die Sitzungskennung enthält. Die Verknüpfung verweist auf eine Übersichtssei-

te für alle Personen. Durch Nutzung von Platzhaltern innerhalb von Verknüpfungen der Vorlage lassen sich dynamische Inhalte generieren. Zur Illustration greifen wir das Beispiel von Seite 179 erneut auf, um es in der tatsächlich häufig eingesetzen Form zu zeigen:

```
...
$[authors a$
  <table>
    <th><td>Autoren</td></th>
    <tr>
      <td>
        <a href="javascript:s.show('person/$a.id$.htm')">$a.name$</a>
      </td>
    </tr>
    <tr>
      <td>
        <a href="javascript:s.show('person/$a.id$.htm')">$a.name$</a>
      </td>
    </tr>
  </table>
$]authors[$
  Autor unbekannt.
$authors]$
...
```

Hier werden also sowohl das eigentliche Ziel der Verknüpfung als auch der dafür angezeigte Name dynamisch per Platzhalter definiert. Mit entsprechenden Regeln ausgestattet, können durch solche oder ähnliche Vorlagen alle Arten von *Assets* auf der Client-Seite verwendet und das Layout sehr flexibel gehandhabt werden.

- Zur Übermittlung sehr großer HTML-Formulare mit vielen Eingabefeldern, die eventuell viel Text enthalten, muß normalerweise statt der üblichen HTTP-Methode `GET` die Methode `POST` verwendet werden, da die Länge der im ersten Fall zur Übermittlung verwendeten URL beschränkt ist. Dies wird transparent von der `show()`-Funktion des Sitzungsobjekts übernommen, indem die Eigenschaften des Formulars reflektiv überprüft werden.

- Die Funktionen zur Annotation, zur Bewertung und zur Verwaltung von Sammelmappen (Aufnehmen von Einträgen und Anzeigen, ob ein *Asset* in der aktuellen Sammelmappe enthalten ist), werden per JAVASCRIPT auf dem Client realisiert. Dazu besitzt das Sitzungsobjekt beispielsweise Funktionen, die ein *Asset* in die Sammelmappen aufnehmen, sowie eine, die ein entsprechendes graphisches Symbol erzeugt. Diese Funktionen greifen dazu nicht direkt per HTTP-Anfrage auf den Server zurück, sondern benutzen temporär im Sitzungsobjekt gespeicherte Statusattribute. Erst bei der jeweils nächsten Anfrage per `show()` werden die gespeicherten Änderungen mit an den Server übertragen. Die Funktionen zur Bewertung und Annotation arbeiten auf die gleiche Weise.

- Weiterhin existieren Funktionen zur Eingabevalidierung, etwa für numerische Felder oder für Datumsfelder. Die Verwendung solcher Funktionen erspart die Überprüfung der Werte auf der Serverseite und die umständliche Fehlerrückmeldung mit der Aufforderung zur erneuten Eingabe. Natürlich muß auf der Serverseite grundsätzlich immer noch eine Validierung der Werte vorgenommen werden; dies verbessert lediglich die Benutzbarkeit.

Insgesamt gesehen erreicht man durch den Einsatz einer auf JAVASCRIPT basierenden Benutzungsoberfläche eine höhere Benutzbarkeit durch gesteigerten Komfort, verglichen mit einer reinen HTML-Lösung. Insbesondere die Möglichkeit, bei zahlreichen Interaktionen die relativ

zeitaufwendigen Client/Server-Verbindungen einzusparen und die neu entstandenen Informationen verzögert erst bei sowieso nötigen Anfragen mitzuschicken, erhöht die Leichtigkeit der Benutzung gerade bei trägen Netzwerkverbindungen erheblich.

7.2 Dokumentenverwaltung für das Projektmanagement

Nach einer Entscheidung der Freien und Hansestadt Hamburg wird derzeit die betriebswirtschaftliche Standardsoftware SAP R/3 in der Verwaltung aller Behörden zum Zwecke des Haushaltsmanagements eingeführt. In einem Einführungsprojekt dieser Größenordnung, an dem über 4000 direkt betroffene Mitarbeiter in den Behörden und diverse externe Beratungshäuser und Softwarehersteller beteiligt sind, wird die Bereitstellung aktueller Projektinformationen zu einem kritischen Faktor. Ein auf der Basis der in den vorangegangenen Kapiteln beschriebenen Erkenntnisse konzipiertes Portal – auch FHH-Broker genannt – für das projektspezifische Dokumenten- und Informationsmanagement wurde zur Lösung dieser Problematik in einem Kooperationsprojekt zwischen Technischer Universität und Finanzbehörde entwickelt und wird mittlerweile erfolgreich eingesetzt [RMS+01]. Das Ziel war es, Arbeitsunterlagen, Mitarbeitererfahrungen und projektspezifische Sachbegriffe allen beteiligten Personen bedarfs- und rollengerecht zugänglich zu machen.

Das Portalsystem unterstützt drei wesentliche Klassen von Projektressourcen, die sich alle drei einfach auf die zuvor beschriebenen Konzepten abbilden lassen:

- **Projektdomänenkenntnisse:** Dies sind projektrelevante Sachbegriffe und Themen mit ihren jeweiligen Definitionen und Zusammenhängen mit anderen Begriffen und Erfahrungen, die in stets änderbaren Begriffsnetzen mit Spezialisierungen und Querverweisen gepflegt werden. Das Begriffsvokabular wird hier insbesondere als zusätzliches Leitsystem verwendet, über das an weitere Projektressourcen gelangt werden kann.

- **Projektmitarbeitererfahrungen:** Diese umfassen ausgewählte Personenstammdaten, Information über die themenbezogene Projektmitarbeit, themenübergreifende Qualifikationen und Interessen sowie die für die rollenbasierten Zugriffsberechtigungen vergebenen Rechte und Gruppenmitgliedschaften in Arbeitsgruppen und Interessengruppen.

- **Projektarbeitsunterlagen:** Dies sind alle Arbeitsunterlagen über Projektziele, Projektmittel, Zwischenstände und Ergebnisse und weitere Ereignisse. Diese Unterlagen werden vollständig in Form digitaler, evtl. multimedialer Dokumente abgelegt und erschlossen. Hier kommt es insbesondere darauf an, sie mit modernen Analysemethoden inhaltlich tief zu erschließen und darauf aufbauend bedarfsgerecht bereitzustellen, etwa durch verschiedenartige Leitsysteme wie das oben angesprochene Begriffsnetzwerk, konventionelle Verzeichnishierarchien und mächtige Recherchefunktionen.

Die Leistungen des Portals liegen also in der Möglichkeit feingranular definierbarer Sichten auf alle Arbeitsunterlagen und eines umfangreichen Angebots von personalisierten Dokumentenmanagementfunktionen. Zudem ist durch die grundsätzlich vorhandenen Integrationsmöglichkeiten eine weitgehende Erweiterbarkeit für zukünftige Anforderungen gegeben.

Die Recherche ist entweder durch dokumentenorientierte Suche möglich, wie in Abbildung 7.2 dargestellt, durch Verzeichnisnavigation oder durch Nutzung des Begriffsnavigators, der auch dort vorhanden ist. Zudem ist es mit Hilfe der verschiedenen Leitsysteme möglich, durch den

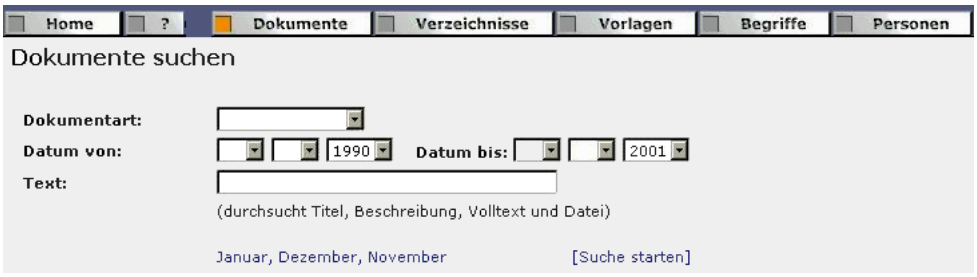

Abbildung 7.2: Recherche durch Dokumentensuche

Bestand an Dokumenten zu navigieren und Beziehungen zu anderen Objekten wie weiteren Dokumenten, Begriffen und Mitarbeitern zu erschließen. Die Navigation durch Verzeichnisse ist in Abbildung 7.3 anhand eines Ausschnittes aus einem Bildschirmabzug illustriert.

Abbildung 7.3: Navigation in Verzeichnissen

Insbesondere die begriffsgestützte Navigation durch die Hierarchie der einzelnen Begriffe und der damit verbundenen Dokumente stellt ein zentrales Merkmal des Systems dar. Dazu können jedem Begriff beliebig viele Dokumente zugeordnet werden, die ihn als inhaltlichen Schwerpunkt besitzen. Jedes Dokument kann mehreren Begriffen zugeordnet werden. Bei Ansicht eines Begriffes werden neben seiner Einbettung in das Begriffsnetz (sowohl im graphischen Begriffsnavigator als auch durch Hyperlinks) die zugeordneten Dokumente angezeigt (siehe Abbildung 7.4).

Schließlich besitzt das Portal Akquisitionsdienste, mit denen Projektmitglieder neue Dokumente in das Dokumentenverwaltungssystem integrieren können, wobei durch automatische Verschlagwortung, inhaltliche Texterschließung und das Vorschlagen von einzutragenden Beziehungen zu den Begriffen diese für andere Personen sichtbar und nutzbar werden. Um Diskussionen zwischen den Benutzern über den Inhalt und die Nutzung des Bestandes an Dokumenten und Begriffen zu ermöglichen, existieren Rückmeldungsmöglichkeiten und Diskussionsforen.

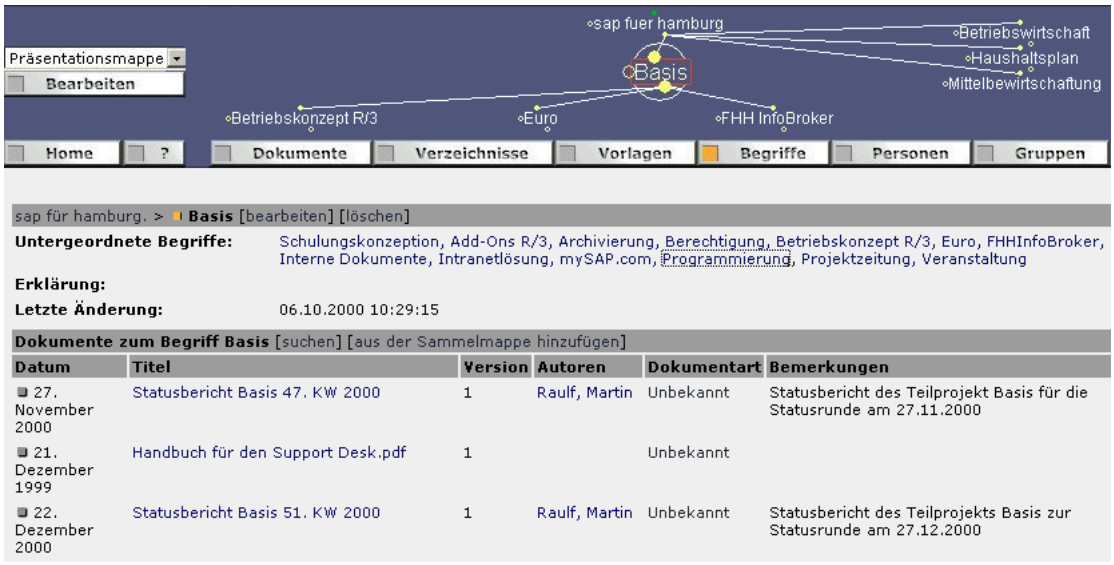

Abbildung 7.4: Begriffsnavigation und Dokumente

Mit diesem seit 1999 andauernden Projekt wurde gezeigt, daß ein Portalsystem, das auf den in den vorangegangenen Kapiteln beschriebenen Konzepten basiert, nicht nur den fachlichen Anforderungen an ein Wissensportal gerecht wird, sondern auch aufgrund der verwendeten Konzepte und der umgesetzten Systemarchitektur, wie sie im Entwurf in Kapitel 6 beschrieben werden, den Systemanforderungen bezüglich großer Einsatzszenarien gerecht wird. In dem Portalsystem des FHH-Projekts sind derzeit deutlich über 10.000 Dokumente in knapp 2.000 Verzeichnissen angesiedelt, die durch eine Begriffstaxonomie mit ca. 300 Begriffen erschlossen sind. Dem System sind über 50 registrierte Benutzer bekannt, die an der Projektplanung beteiligt sind. Es zeigt sich hiermit, daß die Architektur und die Implementierung auch solchen Anforderungen gewachsen sind, und zwar sowohl bezüglich der Quantität der verwalteten Informationen als auch der Qualität, d.h. der Reaktions- und Bearbeitungszeiten seitens des Servers.

7.3 Web-Assessment-Center

Das zweite Anwendungsbeispiel, das in diesem Kapitel beschrieben wird, ist die Entwicklung eines Informationssystems, das dazu dient, Stellenausschreibungen mit Eignungstests für ein Kompetenzmodell zu entwerfen und Bewerber diese Tests über das Internet im Rahmen des Internetauftritts einer Organisation durchführen zu lassen. Diese Entwicklung fand unter Kooperation zwischen der Universität[1] und mehreren Firmen[2] auf der Basis des Kernsystems statt, das im Rahmen dieser Arbeit prototypisch entwickelt wurde und im kommerziellen Kontext weiterentwickelt wird. Im folgenden wird dieses im Projekt *Web-Assessment-Center* genannte System auch kurz WebAC-System genannt.

Das WebAC-System ist neben der Tatsache, daß es auf der Basis des bestehenden Portals realisiert wurde, aus zwei Gründen interessant und wird deswegen hier näher vorgestellt:

1. Dieses Projekt zeigt die Anpassungsfähigkeit der entworfenen Systemarchitektur insbeson-

[1] Arbeitsbereich Softwaresysteme der Technischen Universität Hamburg-Harburg
[2] INFOASSET AG, Hamburg und OBERMANN+SCHIEL GmbH, Köln

dere auf der Ebene der Dienstschicht und der Interaktionsschicht. Der Aufwand zur Ergänzung des konzeptuellen Modells und zur Umsetzung war dabei relativ gering, verglichen mit der Analysephase der fachlichen Domäne.

2. Die hier benötigten fachlichen Konzepte der Kompetenzmodelle und Bewerberprofile passen sehr gut in ein Wissensmanagementsystem zur Beschreibung der persönlichen Qualifikationen von Mitarbeitern (*Skill management*) und können in generalisierterer Form auch nutzbringend in einem allgemeinen Wissensportal Verwendung finden. Ferner gehen die Eigenschaften der automatischen Testdurchführung und Auswertung bereits stark in die Richtung des elektronischen Lernens (*e-Learnings*), also einen Bereich, der sich derzeit starker Aufmerksamkeit sowohl aus der Richtung des Wissensmanagements als auch aus der universitären Lehre erfreut.

Das WebAC-System ist für zwei unterschiedliche Benutzergruppen konzipiert: Personalsachbearbeiter und ggf. Personalberater sowie Bewerber. Beide Nutzergruppen greifen technisch gesehen auf dasselbe System zu, haben aber unterschiedliche Sichten auf die Stellenausschreibung und die Tests. Die Personalsachbearbeiter erstellen ggf. in Zusammenarbeit mit Personalberatern die Stellenausschreibungen, definieren die zugehörigen Anforderungen anhand eignungspsychologischer Überlegungen nach [Obe92] und erstellen Tests, um die Passung der Bewerber zu den Anforderungen messen zu können. Ausschreibung, Anforderungen und Tests für eine Stelle zusammen werden hier als Kompetenzmodell bezeichnet. Das System erlaubt es, eine breite Spanne unterschiedlicher Möglichkeiten zum Test auszunutzen. Schließlich können die Testergebnisse der Bewerber, die sich den Tests unterzogen haben, inspiziert und evaluiert werden. Die Bewerber bemerken von dem eignungspsychologischen Instrumentarium nichts; sie werden lediglich mit einer Reihe von Fragen und ggf. Hinweisen sowie, falls gewünscht, auch mit Rückmeldungen konfrontiert, die aus den entworfenen Tests stammen. Der Bewerber wird dazu durch eine Folge von Seiten geleitet, auf denen jeweils eine oder mehrere Fragen zu beantworten sind, Hinweise zum Verfahren stehen oder die ihm Rückmeldung geben. Um anonyme „Testbewerbungen“ zum Kennenlernen der Fragen zu erschweren, sind zum einen verschiedene Authentifizierungsstrategien vorgesehen, zum anderen läßt sich durch zufällige Permutation und zufällige Auswahl von Fragen aus einem größeren Katalog eine äußerst hohe Zahl von nicht wiederkehrenden Testabläufen erreichen.

7.3.1 Konzeptuelles Modell und Funktionalität

Im folgenden wird zunächst das konzeptuelle Modell des WebAC-Systems und seine Funktionalität kurz beschrieben. Eine Übersicht ist mit einem Klassendiagramm in Abbildung 7.5 gegeben.

Die Ausschreibung einer konkreten Stelle besteht aus einem Kompetenzmodell verbunden mit einem Katalog von Anforderungen an die gewünschten Kompetenzen des Bewerbers. Ausschreibungen werden wie Dokumente in Verzeichnissen verwaltet. Zu jeder Anforderung gehört ein Test, der es gestattet zu messen, inwieweit ein Bewerber die Anforderung erfüllt. Da normalerweise in einem Kompetenzmodell mehrere sehr unterschiedliche fachliche Anforderungen bestehen, können sie hierarchisch gegliedert werden: Ähnlich wie Verzeichnisse und Dateien im Dateisystem eines Rechners existieren hier Anforderungsgruppen und Anforderungen, wobei die Anforderungsgruppen analog zu den Verzeichnissen als Strukturierungshilfe für die Anforderungen dienen. Die Einschränkung ist hier, daß eine Anforderungsgruppe entweder nur Anforderungen oder nur Anforderungsgruppen enthalten kann, nicht aber beides gemischt. Zusätzlich zu den Anforderungen existieren weiterhin Hinweistexte, die den Testablauf für den Bewerber steuern helfen, sowie sogenannte Feedbackstationen, die dem Bewerber an geeigneter

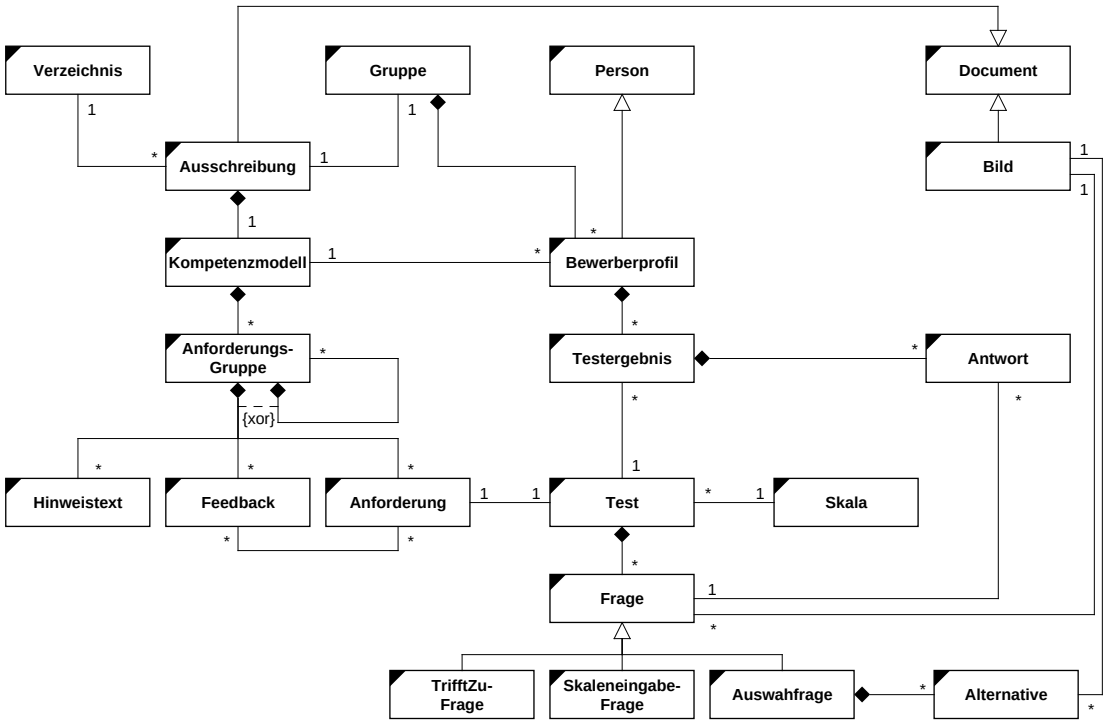

Abbildung 7.5: Konzeptuelles Modell des WebAC-Systems

Stelle Rückmeldung über seine bisherige Leistung zu geben vermögen. Die Verwendung von beidem ist optional. Jedes Kompetenzmodell muß zumindest eine Anforderungsgruppe enthalten, die ihrerseits zumindest wieder eine Anforderung enthalten muß. Auf der obersten Ebene sind nur Anforderungsgruppen möglich. Zur Verwaltung der Bewerberdaten, speziell der persönlichen Informationen wie Titel, Namen, Anschrift, Geburtstag, Telefonnummern etc. bedient sich das Kompetenzmodell der Funktionalität der Personen und Gruppen. Dazu wird jedes Modell zugleich als Gruppe aufgefaßt (indem es mit einer automatisch dafür erzeugten Gruppe verbunden ist), der die entsprechenden Bewerber als Personen über Mitgliedschaften zugeordnet sind.

Eine Anforderung beschreibt eine spezielle gewünschte Kompetenz des Bewerbers, legt fest, inwieweit ein Bewerber diese Kompetenz erfüllen muß und definiert einen Test, der diese Passung ermittelt. Eine Anforderung besteht dazu aus drei Teilen:

1. Der erste Teil beschreibt die gewünschte Kompetenz durch eine textuelle Erläuterung sowie durch eine Skala, anhand derer diese Kompetenz gemessen wird. Die Skala einer Anforderung besteht aus einer geordneten Menge von Werten, die als Texte eingetragen werden können. Ein Beispiel wären die Schulnoten „sehr gut“, „gut“ etc. Eine Skala kann eine Einheit erhalten, die sie näher beschreibt. Zu jedem einzelnen Skalenwert kann ferner eine Liste von Textbausteinen definiert werden, die in den Feedbackstationen verwendet werden können, um dem Bewerber Aufschluss über seine Leistung zu geben.

2. Der zweite Teil einer Anforderung besteht aus der Festlegung der konkreten Anforderung an die Kompetenz. Das Modell dahinter ist nicht so einfach aufgebaut, daß automatisch der höchste Skalenwert das beste Testergebnis ist; in vielen Bereichen ist es häufig auch wünschenswert, fordern zu können, daß eine bestimmte Eigenschaft bei einem Kandidaten nur bis zu einem bestimmten Grad ausgeprägt ist. Um keine reinen ja/nein-Entscheidungen

fällen zu müssen, kann durch Angabe von vier Stützstellen eine Trapezfunktion definiert werden, so daß auch Ergebnisse zwischen ja und nein (bzw. 0% und 100% Erfüllung) möglich werden. Weiterhin muß festgelegt werden, welchem Anforderungstyp („k.o.“, „muß“, „kann“, „Information“) eine Anforderung angehören soll.

3. Der dritte und umfangreichste Teil der Anforderung definiert den Test, der die Kompetenz eines Bewerbers mißt. Das Testverfahren gestaltet sich folgendermaßen: Durch eine Reihe von Fragen wird zunächst eine Punktzahl ermittelt. Diese Punktzahl wird durch eine Abbildungsvorschrift in Form einer Normierungstabelle auf einen Skalenwert abgebildet. Anhand der oben definierten Trapezfunktion wird daraus ein Ergebnis für diese Anforderung als Prozentwert ermittelt. Alle Prozentwerte aller Anforderungen desselben Typs („k.o.“, „muß“, „kann“) zusammen geben gemittelt das Ergebnis der Bewerbung. Wenn alle Werte über den jeweiligen Schwellen liegen, gilt die Bewerbung als erfolgreich. Die Ermittlung der Prozentwerte geschieht dabei ohne Berücksichtigung der Hierarchie der Anforderungsgruppen; die Ergebnisse der Tests aller Anforderungen zählen dabei gleich viel.

   Der eigentliche Test wird durch umfangreiche Einstellungen beschrieben. Dazu gehören die Anzahl der gestellten Fragen, evtl. die Angabe von Mischungsverhältnissen der Fragen verschiedener Schwierigkeitsgrade (was der Erhöhung der Zahl der möglichen Testabläufe dient), die Reihenfolge der Fragen (zufällig oder fest), Zeitbeschränkungen und natürlich die Fragen selbst.

   Es gibt drei verschiedene Typen von Fragen. Allen gemeinsam ist die Beschreibung durch einen Namen, einen Fragetext, die Schwierigkeit, ein Zeitlimit und ein optionales Bild zur Illustration, das ein *Asset* vom Typ Bild (ein Subtyp von Dokument) sein muß. Ansonsten gibt es den Typ der Trifft-Zu-Frage, die genau vier fest vorgegebene Antwortalternativen besitzt („trifft zu“, „trifft eher zu“, „trifft eher nicht zu“ und „trifft nicht zu“), den Typ der Auswahlfrage, die eine frei definierbare Menge an Antwortalternativen besitzt, von denen entweder genau eine oder eine beliebige Anzahl beim Beantworten ausgewählt werden kann (*Single choice* bzw. *Multiple choice*), und schließlich die Skaleneingabefrage, die unter Vermeidung der Anwendung der Normierungstabelle direkt die Auswahl eines Skalenwertes gestattet, was nützlich zum stereotypen Abfragen von Qualifikationen sein kann.

Die Struktur der Anforderungsgruppen legt zugleich auch den Testablauf fest, jedenfalls soweit keine zufällige Reihenfolge eingestellt wurde. Jede Anforderung führt also zur Anzeige von einigen Fragen für den Bewerber. In diese Struktur einbetten lassen sich zusätzliche Hinweistexte und Rückmeldungen, die an der jeweiligen Stelle dem Bewerber angezeigt werden, um Hinweise oder Feedback über die bisher von ihm erbrachte Leistung zu geben. Feedbackstationen stehen dazu in Beziehung zu ausgewählten Anforderungen, um die Rückmeldung gezielt steuern zu können. Mit den Textbausteinen der korrespondierenden Skalenwerte lassen sich individuelle Rückmeldungstexte generieren.

Die Auswertung der Bewerberprofile, also der einzelnen Testergebnisse, ist auf verschiedenen Ebenen möglich, die von einer Übersicht über alle Anforderungen und die jeweils erreichten Ergebnisse bis hin zu der detaillierten Analyse einzelner Antworten geht. Die Übersicht (vgl. Abbildung 7.6) beispielsweise stellt die jeweilige Passung graphisch anhand der Skalenwerte jeder Anforderung dar.

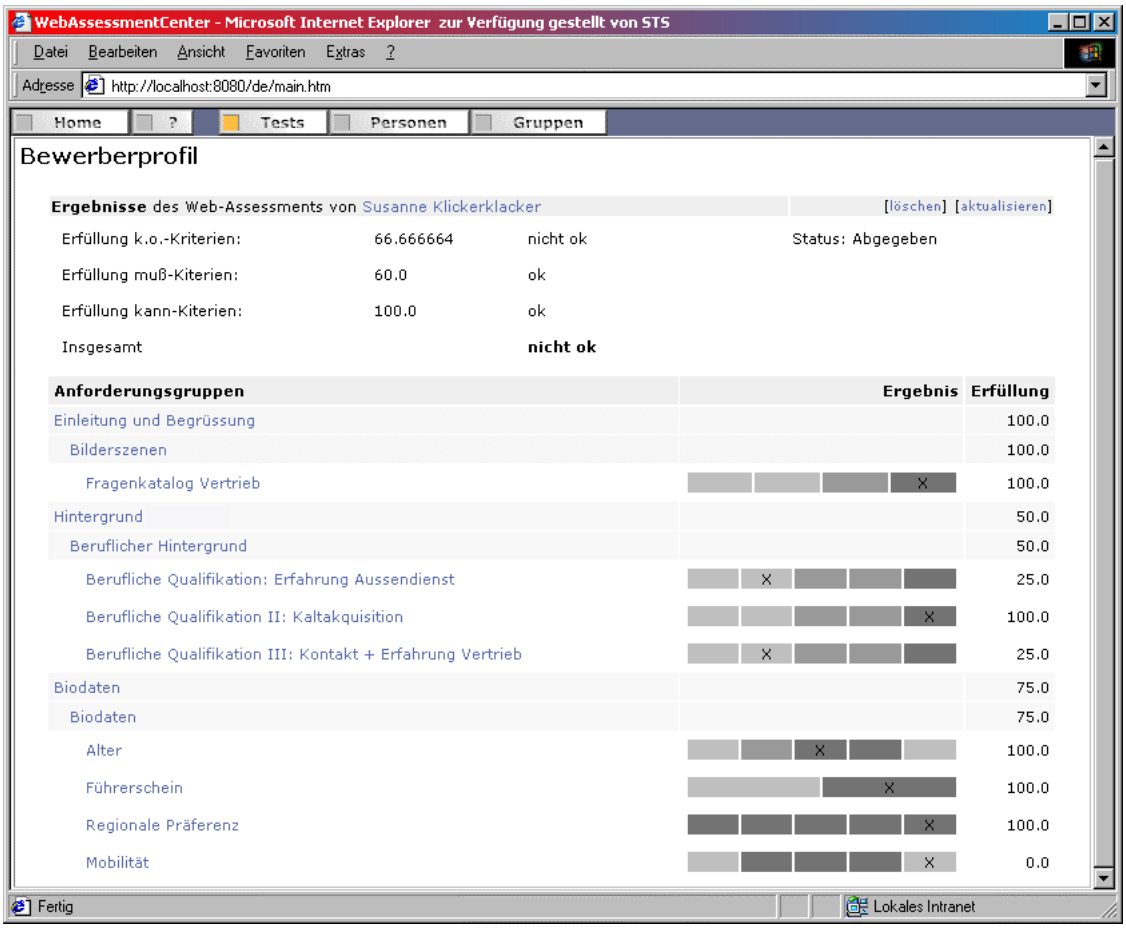

Abbildung 7.6: Auswertung der Bewerbungsergebnisse

7.3.2 Realisierung

Wie bereits in Abbildung 7.5 erkennbar ist, werden alle konzeptuellen Klassen als *Assets* modelliert. Dabei wird das relativ umfangreiche Modell eng an die bestehenden Basiskonzepte der Personen, Dokumente und Verzeichnisse angelehnt, indem

- Bewerber als Subklasse von Personen modelliert werden, die – wie bereits erwähnt – in den jeweils den Ausschreibungen zugeordneten Gruppen verwaltet werden,
- Ausschreibungen als Subklasse von Dokumenten genau wie Dokumente in der Verzeichnisstruktur des Systems aufbewahrt werden und
- die von den Fragen und den Antwortalternativen der Auswahlfragen benutzten Bilddokumente durch eine Dokumentensubklasse modelliert werden.

Die Entwurfsalternative, alle konzeptuellen Klassen (und nicht nur die Ausschreibung) des WebAC-Systems zu Subklassen des Dokuments zu machen, wurde verworfen, da die Eigenschaft der unabhängigen Anordnung in Verzeichnissen hier gerade nicht gewünscht ist, sondern die Erreichbarkeit der einzelnen Objekte durch die komplexen Beziehungen der Klassen untereinander besser gewährleistet wird. Eine Ausnahme bilden die Bilddokumente, die zwecks Wiederver-

wendung aus beliebigen Verzeichnissen stammen dürfen und deshalb als Dokumentensubklasse modelliert werden. Dennoch sind alle Klassen als *Assets* ausgelegt, so daß sie an den üblichen Mechanismen zur Annotation, Bewertung etc. teilnehmen können.

Es hat sich gezeigt, daß auch bei dem als relativ aufwendig geltenden Ansatz, alle konzeptuellen Klassen als neue *Assets* mit entsprechenden *Asset Containern* zu implementieren, der Aufwand sehr begrenzt war, da auf effiziente und weite Bereiche abdeckende Basisimplementierungen zurückgegriffen werden konnte (vgl. Abschnitt 6.4.2). Die Erstellung der Vorlagen und Regeln war im Vergleich damit etwa ebenso aufwendig. Einen etwa ebenso großen Anteil der Zeit kostete die Anforderungsermittlung und Analyse. Für den Produktivbetrieb ist die Möglichkeit, mehrere Web-Server-Instanzen betreiben zu können, sehr günstig, da damit eine strikte Trennung der Bewerbersicht auf die Tests und der Sicht der Personalsachbearbeiter auf der anderen Seite erzwungen werden kann und so auch höheren Sicherheitsanforderungen Rechnung getragen wird.

Insgesamt zeigt dieses Projekt, daß die bestehende Infrastruktur, die auf den in dieser Arbeit entwickelten Konzepten und Entwürfen basiert, mit geringem Aufwand erweiterbar ist und eine hohe Flexibilität im Umgang mit neuen fachlichen Anforderungen ermöglicht.

Kapitel 8

Schluß

Manchmal muß man zurückblicken, um vorwärts zu kommen.
– Capt. KATHRYN JANEWAY, Voyager, Folge 94

Zum Schluß dieser Arbeit werden die Ergebnisse zusammengefaßt und eine Bewertung des Erreichten durch einen Rückblick vorgenommen (Abschnitt 8.1) sowie ein darauf aufbauender Ausblick auf offen gebliebene und im Laufe der Arbeit erschienene neue Fragen gegeben (Abschnitt 8.2).

8.1 Ergebnisse

In dieser Arbeit wurde eine umfassende Analyse der Grundlagen des Wissensmanagements in Hinblick auf die Unterstützung durch Informationssysteme im Bereich der Internet-basierten Portalsoftware vorgenommen, auf der der konzeptuelle und softwaretechnische Entwurf eines integrierten Wissensportals aufbaut. Dabei wurde das verwendete Dokumentenmodell als Gegenstand von besonderer Bedeutung identifiziert, der von bestehenden Systemen bislang noch nicht ausreichend berücksichtigt wird. Mit den *Information Assets* wurde daher ein Metamodell geschaffen, das sowohl die uniforme Modellierung aller Informationsartefakte als auch die Integration bestehender heterogener Landschaften von Informationssystemen erlaubt. Das auf diesen Grundlagen realisierte System wird den formulierten Anforderungen gerecht und zeigte bereits in verschiedenen Projekten seine Leistungsfähigkeit.

Diese äußerst komprimierte Aussage über das Ergebnis der Arbeit wird im folgenden anhand der in der Einleitung formulierten Leitfragen (vgl. Abschnitt 1.2) durch einen Rückblick auf die Teilergebnisse der vorangegangenen Kapitel begründet.

Die beiden wichtigsten Aspekte der Fragestellung gliederten die Arbeit in den Teil der Analyse (Kapitel 2 bis 4) und den der Synthese (Kapitel 5 bis 6). Die Leitfragen zu beiden Aspekten, die hier der besseren Lesbarkeit halber nochmals mit aufgeführt sind (vgl. Seite 5), und die Ergebnisse dazu sind:

Für den ersten Aspekt (konzeptuelle Fundierung und Analyse):

- *Wie läßt sich der Begriff Wissen näher beschreiben und welche Ziele, Aufgaben und Prozesse sind im Wissensmanagement wichtig? Welche davon lassen sich durch Informationssysteme gut unterstützen?*

Der Frage nach einer geeigneten Charakterisierung des Wissensbegriffs wurde zunächst in Abschnitt 2.2 durch die Unterscheidung von Daten, Informationen und Wissen und in Abschnitt 2.3 durch die Diskussion verschiedener Dimensionen des Wissens nachgegangen. Als wichtigste Dimension wurde die Unterscheidung von implizitem und explizitem Wissens gefunden. Der Begriff der Information beschreibt dabei das in Informationssystemen darstellbare Wissen, das per se explizit ist.

Die weitergehende Frage nach Zielen, Aufgaben und Prozessen des Wissensmanagements wurde durch eine Untersuchung der von [PRR99] vorgeschlagenen Bausteine des Wissensmanagements in Abschnitt 2.4 beantwortet, wobei diese Sicht mit einigen anderen in Beziehung gesetzt werden konnte (Abschnitt 2.5). Insbesondere die in den zumeist zyklisch ablaufenden Wissensprozessen auftretenden Teilaufgaben der Wissensverteilung und Wissensnutzung wurden als gut durch Informationssysteme unterstützbar erkannt. Dies deckt sich mit der Einschätzung der Wissensverteilung als „Hebelfunktion“ des Wissensmanagements.

- *Welche Arten von Systemen sind anzutreffen und wie lassen sie sich grundsätzlich klassifizieren? Was leisten Informationssysteme im Bereich des Wissensmanagements bisher?*

  Informationssysteme zur Unterstützung des Wissensmanagements wurden systematisch in Kapitel 3 untersucht. Dabei wurden verschiedene Klassifikationsdimensionen der Systeme nach ihrer Orientierung auf Produkte oder Prozesse sowie nach dem Strukturierungsgrad der verwalteten Informationen unterschieden, die auf der obigen Definition von Information beruht. Eine Untersuchung bestehender Systemklassen ergab, daß die Abdeckung dieser Dimensionen sehr gering ist und spezielle Systeme dominieren. Für das Wissensmanagement – und insbesondere die Wissensverteilung und -nutzung – ist aber eine höhere Abdeckung erforderlich, womit Systeme erforderlich werden, die auf der einen Seite sowohl unstrukturierte, semistrukturierte und strukturierte Informationen einbinden und auf der anderen Seite einen weiten Bereich von Aufgaben und Prozessen im Wissenszyklus unterstützen.

- *Wie lassen sich die eng miteinander verwandten Dokumenten-, Content- und Wissensmanagement-Systeme sowie Portale generell anhand der Konzepte und Funktionalitäten klassifizieren und was leisten Systeme, die bereits im Kontext der Internet-basierten Portalsoftware für das Wissensmanagement anzusiedeln sind?*

  Die Definition und genaue Untersuchung von Dokumenten- und Content-Management-Systemen sowie von Portalen wurde in den Abschnitten 3.3.1, 3.3.2 und 3.3.3 vorgenommen. Dabei wurde zunächst eine enge Verzahnung der Content-Management- mit den Dokumenten-Management-Systemen sowohl auf technischer als auch auf konzeptueller Ebene festgestellt, wobei der als wesentlich erkannte Dokumentenbegriff allerdings vom DMS zum CMS stark weiterentwickelt ist. Der Fokus der CMS und damit der Hauptunterschied zu DMS liegt auf der Unterstützung von Publikationsprozessen und dynamischen Inhalten. Dies bedingt die Differenzierung von Dokumenten in ihre logischen Bestandteile: Struktur, Inhalt, Gestaltung sowie Funktionalität.

  Nachdem sowohl DMS als auch CMS eine Sicht auf unternehmensweite Informationen (entweder für die interne Nutzung oder für die ggf. externe Publikation) von innen besitzen, stellen Portale eine äußere Sicht auf – evtl. die gleichen – Informationen dar. Die sonst sehr uneinheitlich gebrauchte Terminologie wurde konsolidiert und es wurde eine Taxonomie von Portalbegriffen eingeführt. Als für das Wissensmanagement bedeutsamste Art wurde das Wissensportal identifiziert. Als wichtiges Ergebnis wurde die zukünftige Konvergenz unterschiedlicher Portalklassen (Informationsportale, Anwendungsportale, Wissensportale) sowie das Zusammenwachsen interner Funktionalität (Intranet, DMS, Anwendungen) mit der sich entwickelnden externen Funktionalität (Internet, *e-Commerce*,

CRM, SCM etc.) propagiert. Diese Integration von externen und internen Systemen sowie DMS und CMS in Gestalt eines Unternehmens- oder Wissensportals stellt unter Einbeziehung des restlichen organisatorischen Umfeldes das Szenario zukünftiger Wissensmanagement-Systeme dar (vgl. Abbildung 3.7).

Nach dieser Klärung wurden drei konkrete kommerzielle Systeme, die im Wissensportalbereich anzusiedeln sind, eingehend untersucht und ihre Leistungen im Vergleich mit dem zuvor formulierten Szenario bewertet. Die Kernfunktionalität ist bei allen betrachteten Systemen relativ einheitlich und umfaßt alle wesentlichen DMS-Funktionen, die wichtigsten Funktionen zur Prozessunterstützung (meist für einfache Publikationsprozesse) und zum Teil stark ausgeprägte Erschließungs- und Klassifikationsfähigkeiten. Die ebenfalls betrachteten Modelle führen zur nächsten Leitfrage.

- *Welche Modellierungskonzepte werden häufig verwendet und welche Konzepte bedürfen weiterer Untersuchung bzw. wurden noch nicht hinreichend beachtet oder fehlen?*

  Bereits bei der Untersuchung der Systemklassen von DMS und CMS wurde das verwendete Dokumentenmodell – im Sinne eines Metamodells – als wesentlicher Faktor zum Verständnis und zur Beschreibung der Mächtigkeit der Systeme wahrgenommen. Die verwendeten Modelle wurden ebenfalls bei der Untersuchung konkreter Portalsysteme analysiert. Hier ergab sich ein zwiespältiges Bild; die Modelle reichen von nicht formalisierten, nahezu unstrukturierten Dateimetaphern bis hin zu elaborierten, objektorientiert entworfenen konzeptuellen Modellen.

  Die zentrale Erkenntnis dieser Untersuchung ist, daß für den Aufgabenbereich der Portale für das Wissensmanagement mit der oben genannten Kernanforderung der Integration von Informationsbeständen aus verschiedensten Informationssystemen ein problemadäquates Dokumentenmodell fehlt, das die uniforme Repräsentation und Nutzung von Informationsartefakten unterschiedlichster Art (unstrukturiert, semistrukturiert, strukturiert) aus unterschiedlichsten Systemen zur tiefen Erschließung von explizitem und implizitem Wissen ermöglicht.

- *Welche fachlichen Anforderungen an die Funktionalität und welche Systemanforderungen lassen sich für Portalsoftware für Wissensmanagement-Systeme stellen?*

  Neben dem Fehlen eines problemadäquaten Modells wurde durch die Betrachtung der Funktionalität der Systemklassen der DMS, CMS und Portale sowie ausgewählter konkreter Systeme in Abschnitt 3.5 deutlich, daß die Abdeckung der fachlichen Anforderungen gerade im Bereich der Wissensportale noch relativ gering ist und mit einem integrierten Ansatz zur Beschreibung der fachlichen Anforderungen und der Nutzung des eben genannten integrierten Modells deutlich umfassender gestaltet werden könnte.

  Dieser Ansatz wurde in Kapitel 4 umfassend ausgebaut, indem ein feingranular strukturierter Katalog von fachlichen Anforderungen an die Funktionalität und an die Benutzungsoberfläche aufgestellt wurde. Er wurde durch wichtige Anforderungen an das Systemmanagement ergänzt, um dem Integrationsgedanken auf dieser Ebene Rechnung zu tragen. Die Anforderungen untergliedern sich in die Bereiche Dokumente, Personen, Verzeichnisse, Begriffe, Personalisierung, Wissenssuche und Wissenszustellung, Wissensentdeckung, Gemeinschaften, Prozesse und Workflow, Benutzungsoberfläche und Systemanforderungen. Dabei war festzustellen, daß sehr viele Anforderungen eng miteinander verknüpft sind, und es zeigte sich, daß die Fülle der Anforderungen erst mit Hilfe eines uniformen Dokumentenmodells adäquat umgesetzt werden kann, das die Nutzbarmachung und tiefe Erschließung aller Informationsartefakte erlaubt. Außerdem wurde hier bereits deutlich, daß für alle Aufgaben ein konsistenzerhaltendes und bidirektionales Linkmanagement zwingend nötig wird, das dann integraler Bestandteil des Dokumentenmodells sein muß.

Für den zweiten Aspekt (Modelle und softwaretechnischer Entwurf):

- *Wie läßt sich ein problemadäquates Dokumentenmodell ausformulieren, das auf den Einsichten des ersten Aspekts beruht? Welche Bezüge zu bestehenden Modellen lassen sich dabei herstellen?*

  Ein problemadäquates Dokumentenmodell für die Aufgaben eines integrierenden Wissensportals wurde in Kapitel 5 entwickelt. Dazu wurden zunächst die von den klassischen Software- und Informationssystemen verwendeten Datenmodelle analysiert, da gerade Systeme mit solchen Modellen integriert werden müssen. Zu den relevanten Modellen zählen das relationale Modell, das objektorientierte Modell und neuere dokumentenorientierte Modelle. Nach einer eingehenden Bewertung der Modellierungskonzepte wurde als Synthese aller Modelle das Modell der *Information Assets* in Abschnitt 5.1.4 entworfen. Es zeichnet sich durch die flache Aggregation von Attributen aus, die es dem relationalen Modell sehr ähnlich macht, übernimmt aber zugleich die Vorteile der Vererbung von objektorientierten Modellen und die Dynamik und Reflexivität von dokumentenorientierten Modellen. Dieses eigentlich als Metamodell zu bezeichnende Modell stellt das Bindeglied zwischen dem späteren konzeptuellen Modell nebst objektorientiertem Entwurf und Implementierung auf der einen Seite und auf der anderen Seite den verschiedenen zu integrierenden Systemen und Modellen dar und federt so die Modell- und Medienbrüche wirkungsvoll ab. Erst damit wird die uniforme Behandlung aller bestehenden Informationsartefakte ermöglicht.

- *Wie läßt sich ein Prozeßmodell für ein Wissensportal unter besonderer Berücksichtigung bestehender Ansätze zur Modellierung kooperativer Informationssysteme formulieren?*

  Neben dem Dokumentenmodell war als Grundlage für den Entwurf ein abstraktes Prozeßmodell ebenso wichtig. Dieses wurde unter Bezugnahme auf das aus dem Umfeld der kooperativen Informationssysteme stammende Modell der *Business Conversations* in Abschnitt 5.2.2 formuliert. Die wesentliche Abstraktion ist die einer Kunde-Dienstleisterbeziehung in Form einer evtl. durch eine formale Spezifikation ermöglichten Konversation. Die medien- und aktorenunabhängige Modellierung der Konversation nimmt dabei eine zentrale Position ein, die zu der Kapselung der Anwendungslogik in rollenbasierten Regeln führt. Dieses Modell bestimmte den späteren Entwurf der entsprechenden Schichten maßgeblich.

- *Welche Entwurfsentscheidungen sind zu treffen und wie werden sie begründet? Welche Aufgaben des Systems sollten dabei durch bestehende Produkte (Komponenten) erfüllt werden und welche Funktionalität (Komponenten) sollten oder müssen neu entworfen und implementiert werden (make vs. buy)?*

  In Abschnitt 6.1 wurden die wesentlichen Entwurfsentscheidungen getroffen. Als wichtigste ist zu nennen, das gesamte Portalsystem vollständig neu objektorientiert zu entwerfen und nicht durch Zusammenstellung bestehender Softwarekomponenten aufzubauen, um eine möglichst bruchlose Umsetzung der Konzepte des Dokumenten- und Prozeßmodells zu erreichen. Neben der objektorientierten Modellierung und der Notation des Entwurfs mit UML wurde als Implementierungssprache JAVA gewählt. Wie in Kapitel 7 dargelegt wurde, konnten damit konkrete Systeme realisiert werden, die allen Anforderungen entsprechen.

- *Welche System-Architekturen werden von Portalsystemen für das Wissensmanagement häufig eingesetzt und wie sieht eine gute Architektur für ein Portalsystem unter Berücksichtigung des Wissensstandes über Client/Server-Systeme und Mehrschichtenarchitekturen aus? Wie erhält man größtmögliche Freiheit für Skalierbarkeit, neue Anforderungen, Wartbarkeit und Erweiterbarkeit?*

Der Frage nach üblichen Architekturen von Wissensportalen wurde bereits in Kapitel 3 nachgegangen. Naturgemäß kommen hier wie bei den meisten Internet-Informationssystemen mehrschichtige Client/Server-Architekturen zum Einsatz, die i.d.R. eine hohe Skalierbarkeit durch die Verteilung ihrer Komponenten erlauben. Diese Architektur wurde auch für das zu entwerfende System maßgeblich eingesetzt. Sie unterteilt das System in eine Präsentationsschicht, die durch die entfernten Web-Browser sowie andere Endgeräte gebildet wird, und die Serverschichten, die sich aus der Kommunikationsschicht, der Interaktionsschicht, der Dienstschicht, der Speicherungsschicht sowie den von den zu integrierenden bzw. zu nutzenden Informations- und Persistenzsystemen auf der untersten Schicht zusammensetzen. Diese Schichtung gestattet gleichermaßen die Skalierung durch Verteilung wie die nahtlose, effiziente Zusammenarbeit aller Komponenten.

- *Wie sieht der softwaretechnische Entwurf aller Schichten und Komponenten auf Grundlage der Antworten auf die obigen Fragen aus? Welche Abstraktionen und Entwurfsmuster sind zu verwenden?*

  Der Entwurf jeder Schicht wurde in Kapitel 6 detailliert beschreiben, wobei der Entwurf durch zahlreiche UML-Diagramme illustriert wurde. Als zentrale Eigenschaften einzelner Schichten können angesehen werden:

  - Umsetzung des medien- und aktorenunabhängigen Konzepts des Prozeßmodells in Server-Modulen und Regeln in der Kommunikationsschicht sowie teilweise in der Interaktionsschicht. Das Konzept der Geräteklassen erlaubt überdies die ökonomische Bedienung einer großen Anzahl von Endgerätetypen mit unterschiedlichen Eigenschaften. Die Wiederverwendung von Regeln und Vorlagen konnte damit in erheblichem Umfang erreicht werden.
  - Eine effiziente, gut skalierbare und zuverlässige Server-Modul-Architektur, die den simultanen Betrieb von mehreren insbesondere bezüglich der Rechte frei konfigurierbaren HTTP-Server-Modulen, e-Mail-Servern, FTP-Servern etc. ermöglicht, die dennoch uniform über das medien- und aktorenunabhängige Modell Zugang zu der generischen, durch Regeln repräsentierten Anwendungs-Funktionalität haben.
  - Strikte Trennung von Inhalt, Gestaltung und Funktionalität durch Vorlagen, Platzhalter und Regeln. Insbesondere das Konzept für Bedingungen und für Listenabstraktionen stellt einen deutlichen Fortschritt gegenüber anderen Mechanismen (JSP, ASP, PHP etc.) dar und wurde durch eine sowohl entwurfstechnisch als auch programmiersprachlich elegante Lösung umgesetzt, die in den konkreten Projekten bereits eine hohe Produktivität zur Folge hatte.
  - Umsetzung des mit objektorientierten Mitteln beschriebenen konzeptuellen Modells durch eine reichhaltige Sammlung von Schnittstellendefinitionen und ihre Implementierung auf der Ebene der Dienstschicht. Dies stellt sowohl eine objektorientierte Schnittstelle für die Interaktionsschicht dar als auch eine Umsetzung des uniformen und reflexiven Dokumentenmodells der *Information Assets*. Für alle Basiskonzepte (Personen, Dokumente, Verzeichnisse, Begriffe) sowie eine große Zahl weiterer *Assets* existieren in der Praxis erprobte und leistungsfähige Implementierungen.
  - Eine generische und effiziente Basisimplementierung, derer sich alle Implementierungen der konzeptuellen Entitäten bzw. *Assets* bedienen können, um eine nahtlose Anbindung an die Persistenzmechanismen der darunterliegenden Speicherungsschicht zu erhalten. Diese erleichtert das Erstellen der Implementierung von neuen *Assets* durch die Übernahme sämtlicher Routineaufgaben nachhaltig. Das in Abschnitt 7.3 geschilderte Projekt belegte dies nachdrücklich.
  - Die konsistenzsichernden Automatismen der Dienste für bidirektionale Verknüpfungen, die zugleich die Verwendung von verschiedene Informationssysteme übergreifende Verknüpfungen gestattet. Diese Kernfunktion ermöglicht erst die Realisierung

zahlreicher essentieller Klassifikations- und Erschließungsaufgaben. Die Art des Entwurfes und der Realisierung macht diese Funktionalität zudem ungewöhnlich effizient, was die Erfahrungen der verschiedenen Projekte belegen.

- Ein effektives Konzept zur Sicherstellung der Konsistenz bei nebenläufigen Zugriffen auf die *Information Assets*, das erst auf der Ebene der Dienstschicht greift und zusätzliche logische Sperren bereitstellt. Dadurch wird der Integrationsgedanke weiter unterstützt.
- Eine erprobte Anbindung eines generischen Klassifikations-Rahmenwerks [Büc01], das sich seinerseits in hohem Maße der Funktionalität zur sicheren bidirektionalen Verknüpfung beliebiger *Assets* bedient.
- Ein Modell für die Speicherungsschicht, das aus abstrakt beschriebenen Managern, Containern und Inhaltsobjekten gemäß eines explizit konstruierten, dynamischen Inhaltstyps besteht. Die für die wichtigsten, z.T. von Drittherstellern stammenden Informationssysteme (relationale Datenbanken, Dateisysteme, hauptspeicherbasierte persistente Datenbanken etc.) vorhandenen Implementierungen von konkreten Managern, Containern und Inhaltsobjekten ermöglichen die Integration bestehender Informationsquellen und die Nutzung verschiedenster Systeme zur Informationsablage. Das abstrakte Inhaltsmodell erlaubt die effiziente, bruchlose und transparente Abbildung des Modells *Information Assets* auf die jeweiligen Persistenzsysteme. Ein entscheidender Vorteil ist die Kombinierbarkeit und Austauschbarkeit der verschiedenen Implementierungen in wechselnden Einsatzszenarien. Erst dadurch wird der Integrations- und Skalierungsgedanke auf der technischen Ebene des Systemmanagements wirkungsvoll umgesetzt.

Insgesamt gesehen wurde durch die Bruchlosigkeit des Entwurfs eine enge und effizienzbewahrende Kopplung aller Schichten erreicht – beginnend bei den Strukturen der Speicherungsschicht über die generischen *Asset*-Implementierungen bis hin zu den Regeln, Vorlagen und Server-Modulen. Diese zentrale Errungenschaft des entstandenen Systems rechtfertigt den vollständigen Neuentwurf.

Die Erfahrungsberichte und Einsatzbeispiele des entstandenen Systems in verschiedenen Projekten verdeutlichen zudem nicht nur die Erreichung der technisch wünschenswerten Eigenschaften wie Stabilität, Effizienz und Sicherheit, sondern dokumentieren genauso die Richtigkeit der aufgestellten fachlichen Anforderungen, die natürlich gemessen an dem umfassenden Anforderungskatalog fallbezogene Schwerpunkte und Teilmengen besitzen.

Folgende quantitative Aussagen über das Ende 2001 bestehende System lassen sich abschließend treffen: Es ist vollständig in JAVA unter Nutzung des *Java Development Kit* (JDK) 1.2.2 entwickelt worden, wodurch es ebenso unter dem JDK 1.3 und der kommenden Version 1.4 lauffähig ist. Derzeit existieren Anbindungen der Speicherungsschicht an die relationalen Datenbanken ORACLE 8I, MYSQL und eine in der Entwicklung befindliche Anbindung an DB2 sowie an eine ebenfalls in JAVA realisierte persistente hauptspeicherbasierte Datenbank. Ferner existiert eine (bereits in Abschnitt 6.3.3 beschriebene) Anbindung an Dateisysteme und das kommerzielle CMS COREMEDIA.

Die Implementierung des Systems fand seit 1999 hauptsächlich am Arbeitsbereich Softwaresysteme der Technischen Universität Hamburg-Harburg sowie für bestimmte Projekte im kommerziellen Kontext statt (vgl. Abschnitt 7.3) und wurde durch mehrere Mitarbeiter und Studierende unterstützt (vgl. [Leh01, Büc01, Die01, Bar01]). Das implementierte System ist relativ kompakt und besteht inklusive der Realisierungen für die in Kapitel 7 genannten beiden Projekte und zwei weiteren, hier nicht beschriebenen aus derzeit (Ende 2001) etwas weniger als 900 JAVA-Klassen, die sich wie folgt auf die Schichten verteilen:

- Kommunikationsschicht: 41
- Interaktionsschicht[1]: 372
- Dienstschicht: 121 Schnittstellen und 162 Implementierungsklassen
- Speicherungsschicht: 83
- Sonstiges (Hilfsklassen): 101

Das von [Büc01] entwickelte Klassifikationssystem ist in dieser Statistik noch nicht enthalten. Es besitzt einen Umfang von derzeit etwa 200 JAVA-Klassen.

Im normalen Betrieb bietet das System durchweg sehr gute Antwortzeiten; der typische Abruf einer dynamisch generierten Seite (also einer Vorlage mit Platzhaltern, die von Regeln ersetzt werden und die dazu auf die Dienste der Dienstschicht zugreifen, die wiederum die Speicherungsschicht bemüht und so einige Datenbankabfragen auslöst) dauert zwischen 200 ms und 1000 ms und liegt im Mittel unter 500 ms. Dies ist ein für ein System mit dynamischer Seitengenerierung äußerst guter Wert (siehe etwa [Cor00]) und bietet noch erhebliche Leistungsreserven. Das Serversystem selbst ist sehr kompakt und benötigt – sofern die Klassifikationsumgebung nicht eingebunden ist – mit mindestens 32 MB vergleichsweise wenig Hauptspeicher (ohne die Datenbanken und weitere integrierte Systeme).

8.2 Ausblick

I am sorry that I have had to leave so many problems unsolved.
I always have to make this apology, but the world really is rather puzzling and I cannot help it.
– BERTRAND RUSSELL. From The Philosophy of Logical Atomism, Lecture V

Abschließend werden in einem Ausblick noch einige Themen angesprochen, die offen bleiben mußten oder sich als neu und nicht mehr behandelbar erwiesen haben. Außerdem wird eine Prognose der zukünftigen Entwicklung im Umfeld der Wissensportale gegeben.

Das bereits in [Büc01] insbesondere bezüglich der Software-Architektur ausführlich betrachtete Thema der automatischen Klassifikation zur inhaltlichen Erschließung wurde in Abgrenzung zu jener Arbeit hier nicht besonders behandelt. Als Kernkomponente eines Wissensportals (vgl. Abschnitt 4.3.7) ist diese Funktionalität aber so wichtig, daß eine noch wesentlich weitergehende Untersuchung der konkreten Algorithmen und ihrer Konfiguration sinnvoll erscheint. Hier sind auch in hohem Maße empirische Untersuchungen über die Güte und Brauchbarkeit der automatischen Ergebnisse nötig.

Ein ebenfalls hier nur ansatzweise behandeltes Thema ist das der Prozesse und des Workflows. Erste Ergebnisse von [Leh01] sind bereits vorhanden; aber gerade zur Unterstützung der in Abschnitt 4.3.9 genannten Anforderungen ist eine noch wesentlich breitere Untersuchung sowohl der Modelle als auch der praktischen Lösungsmöglichkeiten erforderlich. Zu diesen Themen sind Ergebnisse aus weiteren laufenden Dissertationsvorhaben zu erwarten, die auf den Prozess-Modellen der *Business Conversations* [Hup00] sowie der in [Zie98] definierten Workflow-Metaphern aufbauen. Erste Ansätze sind in [ZHMS01] beschrieben.

[1] nur Regeln, ohne die inneren Klassen der Substitutionen

An die in Abschnitt 7.3 beschriebene Umgebung für Kompetenzmodelle und Tests knüpft die Funktionalität von *e-Learning*-Systemen nahtlos an. Diesen neuen Bereich an die in dem existierenden System vorhandenen Fähigkeiten zur Personenbeschreibung mittels Kompetenzklassifikation, Selbsteinschätzungen, Bewertungen und Empfehlungen sowie der guten Kollaborationsmöglichkeiten, Dokumentenverwaltung etc. anzugliedern stellt eine vielversprechende Erweiterung für ein – thematisch ja recht nahe gelagertes – Wissensportal dar. Ein Portalsystem zur Erschließung der im universitären Umfeld benutzen Lehrmaterialien ist bereits im Aufbau befindlich (vgl. Abbildung 7.1) und könnte ebenfalls von solcher Funktionalität etwa zur Unterstützung von Übungsgruppen und Tutorien profitieren.

Die in Abschnitt 3.3.3 genannte Konvergenz von Portalklassen schloß auch die sogenannten Anwendungsportale zur Integration bestehender Anwendungen innerhalb der Benutzungsoberfläche des Portals mit ein. In dieser Arbeit wurde dieser eher der Oberfläche zuzurechnende Teil nicht weiter untersucht; die Möglichkeiten der Kopplung wären aber noch weitere Untersuchungen wert. Ein systematischer Vergleich mit anderen Portalsystemen, die Anwendungsintegration versprechen, wäre auch angesichts der Neuartigkeit des Themas interessant. Hier ist insbesondere das MYSAP.COM-Portal zu nennen [SAP01].

Auf der technischen Seite wären zukünftige Ziele, die bestehenden und in Kapitel 7 geschilderten – teilweise prototypischen – Systeme auf den vollen Stand der hier entwickelten Entwürfe zu bringen. Eine weitere interessante Untersuchung wäre die Durchführung expliziter Lasttests, um die bisherigen Aussagen über Antwortzeiten etc. systematisch zu messen. Hier wäre auch ein umfassender Vergleich mit anderen Systemen aufschlußreich.

Abschließend soll noch kurz die mögliche zukünftige Entwicklung des Bereichs der Wissens- und Unternehmensportale beleuchtet werden. Die bereits beschriebene Konvergenz der derzeit noch getrennt angebotenen Arten von Unternehmensportalen (vgl. Abschnitt 3.3.3) ist bereits in ersten Produkten sichtbar, etwa dem MYSAP.COM-Portal. Dieser Trend wird sich noch verstärken und auch die bereits sich einander annähernden Dokumentenmanagement- und Content-Management-Systeme mit einbeziehen und ihre Integration in (interne) Unternehmensportale bewirken. Insgesamt werden zuvor strikt getrennte Systeme für Funktionen nach außen (Internet, Systeme für Kunden und Partner) und restriktiv genutzte interne Systeme (Intranet, DMS, CMS, Wissensmanagement-Werkzeuge, ERP) wesentlich stärker integriert werden. Auf der anderen Seite werden explizite Wissensmanagementsysteme weiterhin dem Trend zur Nutzung von Portalen als Zugangssystem unterliegen, so daß auch hier eine starke Annäherung stattfindet. Um die technischen Herausforderungen solcher groß angelegter Integrationsszenarien zu bewältigen, wird die Systemklasse der Anwendungsserver (*Application Server*) noch eine wesentlich höhere Bedeutung erlangen. Hier stehen dann Gesichtspunkte wie Interoperabilität und Standards zum Datenaustausch im Mittelpunkt. In diesem Bereich werden dann XML-basierte Standards und Formate dominieren.

Literaturverzeichnis

[ABH+98] ABECKER, Andreas; BERNARDI, Ansgar; HINKELMANN, Knut; KÜHN, Otto; SINTEK, Michael: Towards a Technology for Organizational Memories. In: *IEEE Intelligent Systems & Their Applications* 13 (1998), Mai/Juni, Nr. 3, S. 40–48

[ADI+00] ACKERMANN, Michael; DIMMELER, Daniel; ITEN, Pascal; MEISTER, Daniel; WEHNER, Theo; KUMBRUCK, Christel (Hrsg.); DICK, Michael (Hrsg.): Wissensmanagement in der Praxis – Umfrageergebnisse und Trends / Technische Universität Hamburg-Harburg, Arbeitsbereich Arbeitswissenschaft 1-08/1. 2000. – Harburger Beiträge zur Psychologie und Soziologie der Arbeit, Nr. 21. – ISSN 0944–565X

[ADK98] ABECKER, A.; DECKER, S.; KÜHN, O.: Aktuelles Schlagwort: Organizational Memory. In: *Informatik Spektrum* 21 (1998), August, Nr. 4, S. 213–214

[AHI00] Architecture of the Hyperwave Information Server / Hyperwave AG. 2000. – White Paper

[Akc00] AKCICEK, Celal: *Studienarbeit: Objekttransformationen bei einem Paradigmenwechsel: Konzepte und Werkzeuge zur Abbildung von Programmobjekten auf Datenbankrelationen*, Universität Hamburg, TU Hamburg-Harburg, Studienarbeit, Oktober 2000

[AKM95] ANDREWS, Keith; KAPPE, Frank; MAURER, Hermann: Serving Information to the Web with Hyper-G. In: *Proceedings of the Third International World Wide Web Conference WWW'95, Darmstadt, Germany* Bd. 27, Elsevier Science, 1995, S. 919–926

[AMB01] ABECKER, Andreas; MAUS, Heiko; BERNARDI, Ansgar: Software-Unterstützung für das Geschäftsprozessorientierte Wissensmanagement. In: MÜLLER, Heinz-Jürgen (Hrsg.); ABECKER, Andreas (Hrsg.); HINKELMANN, Knut (Hrsg.); MAUS, Heiko (Hrsg.): *Proceedings des Workshops Geschäftsprozessorientiertes Wissensmanagement im Rahmen der 1. Konferenz Professionelles Wissensmanagement – Erfahrungen und Visionen (WM 2001), Baden-Baden, 14.-16. März 2001* Bd. 37, 2001

[Ame00] AMELINGMEYER, Jenny: *Wissensmanagement. Analyse und Gestaltung der Wissensbasis von Unternehmen*. Wiesbaden: Deutscher Universitätsverlag, 2000. – ISBN 3–8244–7098–5

[Apa01] *The Apache Software Foundation*. http://www.apache.org. 2001

[App00] factory3: The Next-Generation Enterprise Portal Suite. Offenbach: appsolut software gmbh, April 2000. – White Paper

[AS99] ANTONI, C.H. (Hrsg.); SOMMERLATTE, T. (Hrsg.): *Spezialreport Wissensmanagement: Wie deutsche Firmen ihr Wissen profitabel machen*. 2. Auflage. Düsseldorf: Symposion Publishing GmbH, 1999. – ISBN 3–933814–02–2

[ASP01] *Active Server Pages*. http://msdn.microsoft.com/asp/. 2001

[Att99] ATTIA, Karim: *Persistenz in offenen, verteilten Anwendungssystemen. Network Computing und die Bedeutung von Datenbanksystemen im Rahmen offener, verteilter Informationssysteme*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, April 1999

[Aus62] AUSTIN, J.: How to do things with words / Oxford University Press. 1962. – Forschungsbericht

[Bal96] BALZERT, Helmut: *Lehrbuch der Software-Technik*. 1. Auflage. Spektrum Akademischer Verlag, 1996. – ISBN 3–8274–00420–2

[Bal01] BALZERT, Helmut: *Lehrbuch der Software-Technik*. 2. Auflage. Spektrum Akademischer Verlag, 2001 (Lehrbücher der Informatik). – ISBN 3–8274–0480–2

[Bar01] BARTEL, Rudolf: *Endgeräteunabhängige Realisierung von interaktiven Client/Server-Anwendungen am Beispiel eines integrierten Web- und WAP-Portaldienstes*, Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg, Diplomarbeit, April 2001

[Bau01] BAUER, Herbert: *Unternehmensportale: Geschäftsmodelle, Design, Technologien*. 1. Auflage. Bonn: Galileo Press, 2001. – ISBN 3–89842–133–3

[BBM01] BULLINGER, Hans-Jörg; BUCHER, Michael; MÜLLER, Martin: Knowledge meets system. Wissensbasierte Informationssysteme. Stuttgart: Fraunhofer IRB-Verlag, 2001. – Forschungsbericht. CD-ROM. – ISBN 3–8167–5901–7

[BEG97] BUCK-EMDEN, R.; GALIMOW, J.: *The Client/Server Technology of the SAP R/3 System*. Addison-Wesley Publishing Company, 1997

[BG98] BENN, Wolfgang; GRINGER, Ingo: Zugriff auf Datenbanken über das World Wide Web. In: *Informatik-Spektrum* 21 (1998), Februar, Nr. 1, S. 1–8

[BKM92] BULLINGER, Hans-Jörg; KURZ, Eberhard; MAYER, Renate: Zukunftsorientiertes Dokumenten-Mangement in Böro und Fertigung: Aufgaben, Nutzen und Trends. In: BULLINGER, H.-J. (Hrsg.): *Dokumenten-Mangagement. IAO-Forum 6. Oktober 1992*. Berlin, Heidelberg, New York, London: Springer-Verlag, Oktober 1992 (Forschung und Praxis Band T 34). – ISBN 3–540–56146–3, S. 10–35

[BKOS01] BRY, François; KRAUS, Michael; OLTEANU, Dan; SCHAFFERT, Sebastian: Semistrukturierte Daten. In: *Informatik Spektrum* 24 (2001), August, Nr. 4, S. 230–233

[BL94] BERNDT, Oliver; LEGER, Lothar: *Dokumenten-Management-Systeme. Nutzen, Organisation, Technik*. Neuwied, Kriftel, Berlin: Luchterhand, 1994. – ISBN 3–472–01889–5

[BLHL01] BERNERS-LEE, Tim; HENDLER, James; LASSILA, Ora: Mein Computer versteht mich. In: *Spektrum der Wissenschaft* (2001), August, Nr. 8, S. 42–49

[BLW01] BERINGER, Jörg; LESSMANN, Carsten; WALOSZEK, Gerd: Generic Portal Pages – What Do Most Portals Need? / SAP AG, Usability Engineering Center. SAP Design Guild, 2001. – Forschungsbericht

[BM93] BULLINGER, Hans-Jörg; MAYER, Renate: Integriertes Dokumenten-Mangement: Geschäftsprozeß-Automatisierung und Informationswiedergewinnung. In: BULLINGER, H.-J. (Hrsg.): *Dokumenten-Mangagement: Workflow-Automation und Information Retrieval. IAO-Forum 28. April 1993*. Berlin, Heidelberg, New York, London: Springer-Verlag, April 1993 (Forschung und Praxis Band T 37). – ISBN 3–540–56780–1, S. 10–31

[BMS01] BESTGEN, Jochen; MEIER, Thorsten; SCHMIDT, Carsten: *IT-Konzepte für das Wissensmanagement*. Norderstedt: Books on Demand GmbH, 2001. – ISBN 3–8311–1626–1

[Boo94] BOOCH, Grady: *Object-Oriented Design with Applications*. 2. Auflage. Benjamin/Cummings Publishing Company, 1994

[Boo96] BOOCH, Grady: *Object Solutions: Managing the Object-Oriented Project*. Addison-Wesley Publishing Company, 1996

[Bor00] BOROWSKY, Rainer; SCHEER, August-Wilhelm (Hrsg.): Wissensgemeinschaften: Konzeption und betriebliche Umsetzung eines Knowledge Management-Instruments. Saarbrücken: Institut für Wirtschaftsinformatik (IWi), Universität des Saarlandes, August 2000. – Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 163. – ISSN 1438–5678

[BÖV00] BACH, Volker (Hrsg.); ÖSTERLE, Hubert (Hrsg.); VOGLER, Petra (Hrsg.): *Business Knowledge Management in der Praxis. Prozessorientierte Lösungen zwischen Knowledge Portal und Kompetenzmanagement*. Springer-Verlag, 2000. – ISBN 3–540–67497–7

[BP98a] BORGHOFF, Uwe M. (Hrsg.); PARESCHI, Remo (Hrsg.): *Information Technology for Knowledge Management*. Springer-Verlag, 1998. – ISBN 3–540–63764–8

[BP98b] BRIN, S.; PAGE, L.: The anatomy of a large-scale hypertextual Web search engine. In: ASHMAN, H. (Hrsg.); THISTLEWAITE, P. (Hrsg.): *Proceedings of 7th Int. World Wide Web Conference, Brisbane, Australia* Bd. 30, 1998, S. 107–117

[BRJ99] BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar: *The Unified Modeling Language User Guide*. Addison-Wesley Publishing Company, 1999

[Bro89] *dtv-Brockhaus Lexikon in 20 Bänden*. München: Deutscher Taschenbuch Verlag, 1989

[BS99] BROY, Manfred; SCHMIDT, Joachim W.: Informatik: Grundlagenwissenschaft oder Ingenieurdisziplin? In: *Informatik-Spektrum* 22 (1999), Juni, Nr. 3, S. 206–209

[Büc99] BÜCHNER, Thomas: *Konzeption und Implementation von Fuzzy-Prädikaten in digitalen Bibliotheken*, Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg, Studienarbeit, November 1999

[Büc01] BÜCHNER, Thomas: *Entwurf und Realisierung eines Java-Frameworks zur inhaltlichen Erschließung von Dokumenten*, Technische Universität Hamburg-Harburg, Arbeitsbereich Softwaresysteme, Diplomarbeit, 2001

[Bun97] BUNEMAN, Peter: Semistructured data. In: *Proceedings of the 16th ACM symposium on Principles of database systems (PODS'97)*, 1997, S. 117–121

[Bus45] BUSH, Vannevar: As We May Think. In: *The Atlantic Monthly* 167 (1945), Juli, Nr. 1, S. 101–108

[BVÖ99] BACH, Volker (Hrsg.); VOGLER, Petra (Hrsg.); ÖSTERLE, Hubert (Hrsg.): *Business Knowledge Management. Praxiserfahrungen mit Intranet-basierten Lösungen*. Springer-Verlag, 1999. – ISBN 3–540–65246–9

[BWP97] BULLINGER, Hans-Jörg; WÖRNER, Kai; PRIETO, Juan: Wissensmanagement heute: Daten, Fakten, Trends / Fraunhofer Institut Arbeitswirtschaft und Organisation (IAO), Stuttgart. 1997. – Forschungsbericht

[BWP98] BULLINGER, Hans-Jörg; WÖRNER, Kai; PRIETO, Juan: Wissensmanagement – Modelle und Strategien für die Praxis. In: BÜRGEL, Hans D. (Hrsg.): *Wissensmanagement: Schritte zum intelligenten Unternehmen*. Springer-Verlag, 1998, S. 21–39

[BZTZ00] BÜCHNER, Heino; ZSCHAU, Oliver; TRAUB, Dennis; ZAHRADKA, Rik: *Web Content Management. Websites professionell betreiben*. Bonn: Galileo Press, 2000. – ISBN 3–934358–86–1

[Car99] CARLSEN, Andreas: *Ein generisches Online-Verkaufssystem: Anforderungsanalyse, Objektorientierter Entwurf, Realisierung mit Lotus Notes und SAP R/3*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Juni 1999

[Cat94] CATTELL, R.G.G. (Hrsg.): *Object Data Management: Object oriented and extended relational database systems*. 2. Auflage. Addison-Wesley Publishing Company, 1994

[Cat97] CATTELL, R.G.G.: *The Object Database Standard: ODMG 2.0*. Morgan Kaufman, 1997

[Cat00] CATTELL, R.G.G.: *The Object Database Standard: ODMG 3.0*. San Francisco, Kalifornien: Morgan Kaufman, 2000

[Cla99] CLARK, James: XSL Transformations Specification (XSLT). WWW Consortium (W3C), November 1999. – Recommendation. http://www.w3.org/TR/xslt

[CMS99] CARD, Stuart K.; MACKINLAY, Jock D.; SHNEIDERMAN, Ben: *Readings in Information Visualization – Using Vision to Think*. Morgan Kaufman, 1999. – ISBN 1–55860–533–9

[Cod70] CODD, E.F.: A relational model of data for large shared data banks. In: *Communications of the ACM* 13 (1970), Dezember, Nr. 6, S. 387–387

[Cor00] CoreMedia Publisher / CoreMedia AG. 2000. – White Paper

[Cor01] COREMEDIA AG. *CoreMedia Home Page*. http://www.coremedia.de/. 2001

[Dan99] DANDL, Jörg: Dokumenten-Management-Systeme: Eine Einführung. Mainz: Lehrstuhl für allgemeine BWL und Wirtschaftsinformatik, Johannes Gutenberg Universität, 1999. – Arbeitspapiere WI, Nr. 6/99

[Dat95] DATE, C.J.: *An Introduction to Database Systems*. 6th edition. Addison-Wesley Publishing Company, 1995 (The Systems Programming Series)

[DB92] DOURISH, Paul; BELLOTTI, Victoria: Awareness and Coordination in Shared Workspaces. In: TURNER, J. (Hrsg.); KRAUT, R. (Hrsg.): *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW '92), Toronto, Canada*, ACM Press, November 1992, S. 107–114

[DDJ+97] DE MICHELIS, Giorgio; DUBOIS, Eric; JARKE, Matthias; MATTHES, Florian; MYLOPOULOS, John; PAPAZOGLOU, Mike; POHL, Klaus; SCHMIDT, Joachim; WOO, Carson; YU, Eric: Cooperative Information Systems: A Manifesto. In: PAPAZOGLOU, Mike P. (Hrsg.); SCHLAGETER, Gunther (Hrsg.): *Cooperative Information System: Trends and Directions*. Academic Press, 1997

[DDJ+98] DE MICHELIS, Giorgio; DUBOIS, Eric; JARKE, Matthias; MATTHES, Florian; MYLOPOULOS, John; SCHMIDT, Joachim W.; WOO, Carson; YU, Eric: A Three-Faceted View of Information Systems. In: *Communications of the ACM* 41 (1998), Dezember, Nr. 12, S. 64–70

[Dea99] DEACH, Stephen: Extensible Stylesheet Language (XSL) Specification. WWW Consortium (W3C), April 1999. – Working Draft. http://www.w3.org/TR/WD-xsl

[Del00a] Business Portals: Applications & Architecture. Delphi Group, 2000. – Research Report

[Del00b] Corporate Portals: Portal Design Primer: 68 Questions for Portal Planners. Delphi Group, September 2000. – White Paper: Research Note

[Det00] DETLOR, Brian: The corporate portal as information infrastructure: towards a framework for portal design. In: *International Journal of Information Management* 20 (2000), April, Nr. 2, S. 91–101

[DH95] DALITZ, Wolfgang; HEYER, Gernot: *Hyper-G: Das Internet-Informationssystem der 2. Generation*. Heidelberg: dpunkt.verlag, 1995. – ISBN 3–920993–14–4

[DH96] DALITZ, Wolfgang; HEYER, Gernot: *HyperWave: The new Generation Internet Informations System based on Hyper-G Technology*. Heidelberg: dpunkt.verlag, 1996. – ISBN 3–920993–26–8

[DH98] DEMPSEY, L.; HEERY, R.: Metadata: A current review of practice and issues. In: *Journal of Documentation* 54 (1998), Nr. 2, S. 145–172

[DH99] DICK, Michael; HAINKE, Steffen; KUMBRUCK, Christel (Hrsg.); DICK, Michael (Hrsg.): Das ist doch das Einzige, was ich habe an Kapital. Mitarbeitereinschätzungen über Wissensmanagement / Technische Universität Hamburg-Harburg, Arbeitsbereich Arbeitswissenschaft 1-08/1. 1999. – Harburger Beiträge zur Psychologie und Soziologie der Arbeit, Nr. 16. – ISSN 0944–565X

[Die01] DIESTELHORST, Lars: *Recommendation Engines*, Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg, Studienarbeit, Juli 2001

[DIN78] *DIN Taschenbuch 25: Informationsverarbeitung 1, DIN 44300*, 1978

[Dit00] DITTMAR, Carsten: Vom Data- zum Knowledge-Warehouse – Wissen sichtbar machen. In: *Computerwoche extra* (2000), Nr. 4, S. 14–17

[Dob00] DOBRATZ, Susanne: Dissertationsportale im Internet. In: *nfd Information – Wissenschaft und Praxis* 51 (2000), Nr. 6

[DP98] DAVENPORT, Thomas H.; PRUSAK, Laurence: *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, 1998

[DP99] DAVENPORT, Thomas H.; PRUSAK, Laurence: *Wenn Ihr Unternehmen wüsste, was es alles weiß. Das Praxisbuch zum Wissensmanagement*. Verlag Moderne Industrie, Landsberg/Lech, 1999

[Dru98] DRUCKER, P. F.: Knowledge Management. In: *Harvard Business Review* (1998), April

[Dud88] *Duden Informatik*. Mannheim, Wien, Zürich: Dudenverlag, 1988. – ISBN 3–411–02421–6

[Dud89] *Deutsches Universalwörterbuch A–Z*. Mannheim ; Wien ; Zürich: Dudenverlag, 1989. – ISBN 3–411–02176–4

[Due01] DUECK, Gunter: Kopfgold (oder: Knowledge Management). In: *Informatik Spektrum* 24 (2001), Dezember, Nr. 6, S. 387–392

[Ede96] EDELMANN, Walter: *Lernpsychologie*. 5. vollst. überarb. Auflage. Weinheim, Basel: Beltz Verlag, 1996

[EHB+00] EUSTACE, Clark; HOLTHAM, Clive W.; BIANCHI, Patrizio; COHEN, Laurance J.; EDVINSSON, Leif; ENQVIST, Reinhold; FIDLER, Simon; LEV, Baruch; RAMIN, Kurt P.; VOLLMANN, Thomas E.; ZAMBON, Stefano: The Intangible Economy: Impact and Policy Issues / European Commission Directorate General for Enterprise. 2000. – Report of the European High Level Expert Group on the Intangible Economy

[EN89] ELMASRI, S.; NAVATHE, S.: *Fundamentals of Database Systems*. Benjamin/Cummings Publishing Company, 1989

[End99] ENDRES, Albert: Die Informatik als Ingenieurwissenschaft. Noch ein Beitrag zu einer nicht endenden Diskussion. In: *Informatik-Spektrum* 22 (1999), Dezember, Nr. 6, S. 439–443

[Eng90] ENGELBART, Douglas: Knowledge-Domain Interoperability and an Open Hyperdocument System. In: *Proceedings of ACM CSCW'90 Conference on Computer Supported Cooperative Work, Los Angeles*, ACM Press, Oktober 1990, S. 143–156

[EOO94] EBERLEH, Edmund (Hrsg.); OBERQUELLE, Horst (Hrsg.); OPPERMANN, Reinhard (Hrsg.): *Einführung in die Software-Ergonomie: Gestaltung graphisch-interaktiver Systeme: Prinzipien, Werkzeuge, Lösungen*. 2. Auflage. Berlin: de Gruyter, 1994. – ISBN 311013814X

[Ern98] ERNST, Matthias: *Typüberprüfung in einer polymorphen objektorientierten Programmiersprache. Analyse, Design und Implementierung eines Typprüfers für Tycoon-2*, Fachbereich Informatik, Universität Hamburg, Studienarbeit, Juni 1998

[ES00] ERNST, Matthias; SCHNEIDER, Daniel: *Konzepte und Implementierungen moderner virtueller Maschinen*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Dezember 2000

[Far99] FARSI, Reza: Aktuelles Schlagwort: XML. In: *Informatik-Spektrum* 22 (1999), Dezember, Nr. 6, S. 436–438

[Fau00] FAULSTICH, Rainer: *Internet Portals for Electronic Commerce*, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, Universität Karlsruhe, Diplomarbeit, Oktober 2000

[FGHW88] FLORES, F.; GRAVES, M.; HARTFIELD, B.; WINOGRAD, T.: Computer Systems and the Design of Organizational Interaction. In: *ACM Transactions on Office Information Systems* 6 (1988), Nr. 2, S. 153–172

[FGM+99] FIELDING, R.; GETTYS, J.; MOGUL, J.; FRYSTYK, H.; MASINTER, L.; LEACH, P.; BERNERS-LEE, T.: Hypertext Transfer Protocol – HTTP/1.1 / W3C. 1999. – RFC 2616

[Fir00] FIRESTONE, Joseph M.: Enterprise Knowledge Portals: What They Are and What They Do. In: *Knowledge and Innovation: Journal of the KMCI* 1 (2000), Oktober, Nr. 1, S. 85–108

[FPU98] FOCHLER, K.; PERC, P.; UNGERMANN, J.: *Lotus Domino 4.6: Internet- und Intranetlösungen mit dem Lotus Domino Server*. Addison-Wesley Publishing Company, 1998

[Fra92] FRAKES, William B.: Stemming algorithms. In: FRAKES, William B. (Hrsg.); BAEZA-YATES, Ricardo (Hrsg.): *Information Retrieval: Data structures and algorithms*. Prentice Hall, 1992, S. 131–160

[FS99] FOWLER, Martin; SCOTT, Kendall: *UML Distilled: A Brief Guide to the Standard Object Modelling Language*. 2. Auflage. Addison-Wesley Publishing Company, 1999. – ISBN 0–201–65783–X

[Fuh96] FUHR, N.: Models for Integrated Information Retrieval and Database Systems. In: *Data Engineering Bulletin* 19 (1996), März, Nr. 1, S. 3–13

[Gaß99] GASSEN, Helga: Wissensmanagement – Grundlagen und IT-Instrumentarium. Mainz: Lehrstuhl für allgemeine BWL und Wirtschaftsinformatik, Johannes Gutenberg Universität, 1999. – Arbeitspapiere WI, Nr. 6/99

[Gen99] GENTSCH, Peter: *Wissen managen mit innovativer Informationstechnologie: Strategien – Werkzeuge – Praxisbeispiele*. Wiesbaden: Gabler, 1999. – ISBN 3–409–19016–3

[GFS98] GOESMANN, Thomas; FÖCKER, Egbert; STRIEMER, Rüdiger: Wissensmangement zur Unterstützung der Gestaltung und Durchführung von Geschäftsprozessen / Fraunhofer Institut für Software- und Systemtechnik (ISST). Dortmund, 1998. – ISST Bericht Nr. 48

[GHJV95] GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSADES, J.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Publishing Company, 1995. – ISBN 0–201–63361–2

[GHS91] GREWENDORF, Günther; HAMM, Fritz; STERNEFELD, Wolfgang: *Sprachliches Wissen: Eine Einführung in moderne Theorien der grammatischen Beschreibung*. 5. Auflage. Frankfurt/Main: Suhrkamp, 1991

[GJS96] GOSLING, J.; JOY, B.; STEELE, G.: *The Java Language Specification*. Addison-Wesley Publishing Company, 1996 (The Java Series)

[Glu97] GLUCHOWSKI, Peter: Das aktuelle Schlagwort: Data Warehouse. In: *Informatik-Spektrum* 20 (1997), Februar, Nr. 1, S. 48–49

[GM95] GAWECKI, A.; MATTHES, F.: TooL: A Persistent Language Integrating Subtyping, Matching and Type Quantification / FIDE Project Coordinator, Department of Computing Sciences, University of Glasgow. 1995 (FIDE/95/135). – FIDE Technical Report Series

[Gol99] Internet Portals In Europe. London: Goldman Sachs Investment Research, März 1999. – Research Paper

[GR01] GARCIA, Roberto; RIBAS, Christine: Multi-Access Portals: Information Whenever, Wherever, However You Want It. Madrid: c-quential, Arthur D. Little, Januar 2001. – Forschungsbericht

[Gri98] GRIFFEL, Frank: *Componentware: Konzepte und Techniken eines Softwareparadigmas*. Heidelberg: dpunkt.verlag, 1998

[Gro00a] GRONAU, Norbert (Hrsg.): *Technologien des Wissensmanagements. Veranstaltungsdokumentation*. Universität Oldenburg, Fachbereich Informatik, Abteilung Wirtschaftsinformatik, Juni 2000

[Gro00b] GRONAU, Norbert: Trends im Wissensmanagement. In: [Gro00a], S. 1–18

[Gro01] GRONAU, Norbert (Hrsg.): *Wissensmanagement: Systeme – Anwendungen – Technologien*. Aachen: Shaker Verlag, 2001 (Berichte aus der Wirtschaftsinformatik). – ISBN 3–8265–9024–4

[GW98] GAWECKI, Andreas; WIENBERG, Axel: Report on the Tycoon-2 Programming Language. Version 1.0 (Draft) / Higher-Order GmbH, Hamburg, and Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg. 1998. – Forschungsbericht

[GWF+99] GOLAND, Y.; WHITEHEAD, E.; FAIZI, A.; CARTER, S.; JENSEN, D.: HTTP Extensions for Distributed Authoring – WEBDAV. 1999. – RFC 2518

[Hab99] HABERMANN, Frank; SCHEER, August-Wilhelm (Hrsg.): Organisational-Memory-Systeme für das Management von Geschäftsprozesswissen. Saarbrücken: Institut für Wirtschaftsinformatik (IWi), Universität des Saarlandes, Dezember 1999. – Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 154. – ISSN 1438–5678

[Hau00] HAUER, Manfred: Automatische Indexierung. In: [Sch00b], S. 203–212. – ISBN 3–925474–41–2

[Hei00] HEID, Ulrike: Ein Konzept für Wissensmanagement: Ein Vergleich von Wissensmanagement und Informationsmanagement. In: *nfd Information – Wissenschaft und Praxis* 51 (2000), Nr. 7

[Heu92] HEUER, Andreas: *Objektorientierte Datenbanken: Konzepte, Modelle, Systeme*. Addison-Wesley Publishing Company, 1992

[HIP00] Hyperwave Information Portal / Hyperwave AG. 2000. – White Paper, Document Number HW-MWP-HIP-003

[HMW98] HUPE, Patrick; MATTHES, Florian; WEGNER, Holm: Ein bruchloser Übergang von der Prozeßmodellierung zu kooperativen Software-Architekturen / Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg. 1998. – Forschungsbericht

[Höh01] HÖHNE, Michael: Message in a Portal. In: *groupware magazin* (2001), August, Nr. 8, S. 86–89

[Hup98] HUPE, Patrick: *Ein Typsystem zur Analyse dialogorientierter Workflows in kooperativen Informationssystemen*, Fachbereich Informatik, Universität Hamburg, Studienarbeit, November 1998

[Hup00] HUPE, Patrick: *Eine daten- und prozeßorientierte Architektur zur Integration kooperierender Informationssysteme*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Februar 2000

[HV98] HEISIG, Peter; VORBECK, Jens: Benchmarking Wissensmanagement – Best-Practices in Deutschland und Europa / Fraunhofer Institut für Produktionsanlagen und Konstruktionstechnik (IPK), Bereich Planungstechnik, Berlin. 1998. – Forschungsbericht

[Hyp01a] *Hyperwave Information Management AG*. http://www.hyperwave.de/. 2001

[Hyp01b] HYPERWAVE AG: Hyperwave eLearning Suite: Interaktives Training auf der Basis von Wissensmanagement / Hyperwave AG. 2001. – Broschüre

[IW99a] Intelligente Systeme als Motor des Markts. In: *Information Week* (1999), September, Nr. 21, S. 10

[IW99b] Wissen aus der virtuellen Hängemappe. In: *Information Week* (1999), September, Nr. 20, S. 48

[IW99c] Wissen ja – Management nein. In: *Information Week* (1999), Oktober, Nr. 22, S. 18

[IW00a] Content-Management: Das Ende der Beschaulichkeit. In: *Information Week* (2000), April, Nr. 11, S. 52

[IW00b] Content-Management: Kundenpflege über das Internet. In: *Information Week* (2000), August, Nr. 20, S. 32

[IW00c] DMS-Markt: Eine geschlossene Gesellschaft. In: *Information Week* (2000), August, Nr. 20, S. 28

[IW00d] Rückenwind von SAP & Co. In: *Information Week* (2000), August, Nr. 20, S. 22

[IW00e] Schneller als jedes Auge. In: *Information Week* (2000), April, Nr. 10, S. 63

[IW00f] Serie: Dokumentenmanagement. In: *Information Week* (2000), Februar, Nr. 4–6

[IW00g] Wissensmanagement: Bringt der Rabe Glück? In: *Information Week* (2000), September, Nr. 21, S. 20

[IW01a] Portale werden mobil. In: *Information Week* (2001), Nr. 8, S. 9

[IW01b] Das Tor zum Unternehmen. In: *Information Week* (2001), März, Nr. 7, S. 58 ff.

[IW01c] Wissensmanagement lockt die Anwender. In: *Information Week* (2001), April, Nr. 9, S. 26

[JBÖ00] JANSEN, Christoph M.; BACH, Volker; ÖSTERLE, Hubert: Knowledge Portals: Using the Internet to Enable Business Transformation. In: *Proceedings of the 10th Annual Internet Society Conference (INET 2000), 18-21 July 2000, Yokohama, Japan*, 2000

[JBR99] JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James: *The Unified Software Development Process*. Addison-Wesley Publishing Company, 1999

[JBS97] JABLONSKI (Hrsg.); BÖHM (Hrsg.); SCHULZE (Hrsg.): *Workflow-Management: Entwicklung und Anwendung von Systemen*. 1. Auflage. Heidelberg: dpunkt.verlag, 1997

[JCJÖ92] JACOBSON, I.; CHRISTERSON, M.; JONSON, P.; ÖVERGAARD, G.: *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley Publishing Company, 1992

[Joh97] JOHANNISSON, Nico: *Eine Umgebung für mobile Agenten: Agentenbasierte verteilte Datenbanken am Beispiel der Kopplung autonomer Internet Web Site Profiler*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, April 1997

[JS96] JavaScript Language Specification, Version 1.1. Netscape Inc., 1996. – Forschungsbericht

[JV87] JESSEN, Eike; VALK, Rüdiger: *Rechensysteme: Grundlagen der Modellbildung*. Berlin; Heidelberg; New York: Springer-Verlag, 1987 (Studienreihe Informatik). – ISBN 3–540–16383–2

[KA98] KÜHN, Otto; ABECKER, Andreas: Corporate Memories for Knowledge Management in Industrial Practice: Prospects and Challenges. In: [BP98a], S. 183–206. – ISBN 3–540–63764–8

[Kah98] KAHLBRANDT, Bernd: *Software-Engineering, Objektorientierte Software-Entwicklung mit der Unified Modeling Language*. Springer-Verlag, 1998

[KAM95] KAPPE, Frank; ANDREWS, Keith; MAURER, Hermann: The Hyper-G network information system. In: *Journal of Universal Computer Science* 1 (1995), Nr. 4, S. 206–220

[Kam01] KAMP, Gerd: Multichannel-Publishing. In: *Objekt Spektrum* (2001), Nr. 5

[Kap91] KAPPE, Frank: *Aspects of a Modern Multi-Media Information System*, Institute for Foundations of Information Processing and Computer Supported New Media (IICM), Graz University of Technology, Dissertation, Juni 1991

[Kap94] KAPPE, Frank: Hyper-G text format (HTF) / IICM, Technische Universität Graz. 1994. – Forschungsbericht

[Kap95] KAPPE, Frank: A Scalable Architecture for Maintaining Referential Integrity in Distributed Information Systems. In: *Journal of Universal Computer Science* 1 (1995), Februar, Nr. 2, S. 84–104

[Kap99a] KAPPE, Frank: Hyperwave Information Server / Hyperwave AG. 1999. – Technical White Paper

[Kap99b] KAPPE, Frank: Managing Knowledge with Hyperwave Information Server / Hyperwave AG. 1999. – White Paper

[Kap01] KAPS, Gabriele; NOHR, Holger (Hrsg.): Erfolgsmessung im Wissensmanagement unter Anwendung von Balanced Scorecards / Fachhochschule Stuttgart. 2001. – Arbeitspapiere Wissensmanagement, Nr. 2/2001. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[KE97] KEMPER, A.; EICKLER, A.: *Datenbanksysteme – Eine Einführung*. München, Wien: Oldenbourg Verlag, 1997

[Kes00] KESSLER, Thomas H.: Content Management. Content: Notwendigkeit für E-Business / WestLB Panmure. 2000. – Analyse

[KL86] KANTOR, Brian; LAPSLEY, Phil: Network News Transfer Protocol: A Proposed Standard for the Stream-Based Transmission of News. 1986. – RFC 977

[KLT00] KOENEMANN, Jürgen; LINDNER, Hans-Günther; THOMAS, Christoph: Unternehmensportale: Von Suchmaschinen zum Wissensmanagement. In: *nfd Information – Wissenschaft und Praxis* 51 (2000), September, Nr. 6, S. 325–334

[KMP01] KEMP, J.L.C.; MOERMAN, P.A.; PRIETO, J.: On the Nature of Knowledge-intensive Organisations: Strategy and Organisation in the New Economy. In: THOBEN (Hrsg.); WEBER (Hrsg.); PAWAR (Hrsg.): *Proceedings of 7th International Conference on Concurrent Enterprising, Bremen, Juni 2001*, 2001, S. 251–260

[KN96] KAPLAN; NORTON: *The Balanced Score Card*. HBS Press, Boston, 1996

[KN01] KAPS, Gabriele; NOHR, Holger: Erfolgsmessung im Wissensmanagement mit Balanced Scorecards. In: *nfd Information – Wissenschaft und Praxis* 52 (2001), Nr. 2–3, S. 89–97, 151–158

[KO96] KIRN, Stefan (Hrsg.); O'HARE, Gregory (Hrsg.): *Cooparative Knowledge Processing. The Key Technology for Intelligent Organizations*. Springer-Verlag, 1996 (Computer Supported Cooperative Work). – ISBN 3–540–19951–9

[KP94] KAPPE, Frank; PANI, Gerald: Hyper-G Client/Server Protocol (HP-CSP), Version 7.05 / IICM, Technische Universität Graz. 1994. – Forschungsbericht

[Kre99] KREUZ, Detlef: *Formale Semantik von Konnektoren*, Technische Universität Hamburg-Harburg, Dissertation, Mai 1999

[Kru97] KRUSE, Wolfgang: *Historienmanagement für kooperative Aktivitäten*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Dezember 1997

[Ksi99] KSIEZYK, Rafal: Trying not to get lost with a topic map. In: *Proc. of XML Europe 99*, 1999

[Lag87] LAGEMANN, Klaus: *Rechnerstrukturen*. Springer-Verlag, 1987. – ISBN 3–540–17618–7

[LD01] LACHER, Martin S.; DECKER, Stefan: On the Integration of Topic Map data and RDF data. In: *Proceedings of the 1st International Semantic Web Working Symposium (SWWS '01), Stanford University, Stanford, CA, July 29–Aug 1, 2001*, 2001, S. 331–344

[Leh01] LEHEL, Vanda: *Entwurf und Realisierung eines Basisdienstes zur Unterstützung arbeitsteiliger Geschäftsvorgänge in einem Unternehmensportal*, Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg, Studienarbeit, Juli 2001

[Lex91] *Die große Bertelsmann Lexikothek*. Gütersloh: Bertelsmann Lexikothek Verlag, 1991

[LL99] LATZA, Jens; LÜHR, Rüdiger: *Eine generische und objektorientierte Schnittstelle von TYCOON zum System SAP R/3*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Februar 1999

[Loe98] LOESER, Henrik: Techniken für Web-basierte Datenbankanwendungen: Anforderungen, Ansätze, Architekturen. In: *Informatik Forsch. Entw.* 13 (1998), S. 196–216

[Lot98] LOTUS DEVELOPMENT INTERNATIONS CORPORATION. *Lotus | Domino Server 4.6.1*. 1998

[Lot01a] Lotus Knowledge Discovery System: Administrator's Guide / Lotus Development Corporation. 2001. – Handbuch

[Lot01b] Lotus K-station Overview / Lotus Development Corporation. 2001. – White Paper

[Lot01c] Lotus K-station. Reviewer's Guide / Lotus Development Corporation. 2001. – White Paper

[Lot01d] Lotus Discovery Server. Reviewer's Guide / Lotus Development Corporation. 2001. – White Paper

[Lot01e] Building Enterprise Taxonomies with Lotus Discovery Server / Lotus Development Corporation. 2001. – White Paper

[Lot01f] Lotus Discovery Server: Taking Advantage of the Collective Experience in Your Organization / Lotus Development Corporation. 2001. – White Paper

[Lot01g] Locating Organizational Expertise with the Lotus Discovery Server / Lotus Development Corporation. 2001. – White Paper

[LS87] LOCKEMANN, P.C. (Hrsg.); SCHMIDT, J.W. (Hrsg.): *Datenbank-Handbuch*. Springer-Verlag, 1987 (Informatik-Handbücher). – ISBN 3–540–10741–X

[Luf99] LUFTER, Jens: Objektrelationale Datenbanksysteme. In: *Informatik-Spektrum* 22 (1999), August, Nr. 4, S. 288–290

[Lut97] LUTZ, Sebastian: *Eine polymorph typisierte Schnittstelle der Sprache Tycoon zum System SAP R/3*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Juni 1997

[MAB+99] MORRISA, Henry; ACLY, Ed; BYRON, Dennis; GARONE, Steve; MCDONOUGH, Brian; MURRAY, Gerry; SWEENEY, Jacqueline; VILLARS, Richard: Portal Mania: Who Will Lead the Way to Convergence? International Data Corporation, Juni 1999. – Bulletin, Document Nr. 19325

[Mat93] MATTHES, Florian: *Persistente Objektsysteme: Integrierte Datenbankentwicklung und Programmerstellung*. Springer-Verlag, 1993. – ISBN 3–540–56581–7

[Mat96] MATHISKE, Bernd: *Mobilität in persistenten Objektsystemen*, Fachbereich Informatik, Universität Hamburg, Diss., Oktober 1996

[Mat97] MATTHES, Florian: Mobile Processes in Cooperative Information Systems. In: *Proceedings of STJA'97 (Smalltalk und Java in Industrie und Ausbildung)*. Erfurt, Germany: TU Ilmenau, Institut für Biomedizinische Technik und Informatik, Ilmenau, Germany, September 1997, S. 23–28

[Mat98] MATTHES, F.: Business Conversations: A High-Level System Model for Agent Coordination. In: CLUET, Sophie (Hrsg.); HULL, Rick (Hrsg.): *Database Programming Languages: Proceeding of the 6th International Workshop, DBPL-6, Estes Park, Colorado, USA, August 1997* Bd. 1369, Springer-Verlag, 1998. – ISBN 3–540–64823–2, S. 355–372

[Mat00] MATTHES, F.: Higher-Order Persistent Polymorphic Programming in Tycoon. In: ATKINSON, Malcom P. (Hrsg.); WELLAND, Ray (Hrsg.): *Fully Integrated Data Environments*. Springer-Verlag, 2000 (ESPRIT Basic Research Series). – ISBN 3–540–65772–X, S. 13–59

[Mau96] MAURER, Hermann (Hrsg.): *Hyper-G – now Hyperwave: The next Generation Web Solution*. Addison-Wesley Publishing Company, 1996. – ISBN 0–201–40346–3

[Met00] Der Markt für Portale, Marktplätze und Mobile Commerce in Deutschland. META Group Deutschland, 2000. – Studie

[Met01a] Unternehmensprofil Hyperwave AG. META Group Deutschland, 2001. – Teilstudie, erstellt im Rahmen von [Met00]

[Met01b] Unternehmensprofil Microsoft Deutschland GmbH. META Group Deutschland, 2001. – Teilstudie, erstellt im Rahmen von [Met00]

[Mey88] MEYER, Bertrand: *Object-oriented Software Construction*. Prentice Hall, 1988 (International Series in Computer Science)

[Mic01] MICROSOFT CORPORATION. *Microsoft Sharepoint Portal Server Home Page*. http://www.microsoft.com/sharepoint/default.asp. 2001

[MMZM73] MEADOWS, Dennis; MEADOWS, Donella; ZAHN, Erich; MILLING, Peter: *Die Grenzen des Wachstums. Bericht des Club of Rome zur Lage der Menschheit*. Reinbek bei Hamburg: Rowohlt Verlag, 1973

[MNSS99] MATTHES, Florian; NIEDERÉE, Claudia; SCHMIDT, Joachim W.; STEFFENS, Ulrike: Das Internet als Wissensmarkt. Möglichkeiten und Grenzen. In: KERSTEN, Wolfgang (Hrsg.); KUMBRUCK, Christel (Hrsg.): *Wissenmarkt Internet – Zwischen betrieblichem Wissensmanagement und virtueller Universität*. Technische Universität Hamburg-Harburg, Juli 1999 (Harburger Beiträge zur Psychologie und Soziologie der Arbeit, Sonderband 1). – ISSN 0944–565X, S. 10–25

[Möl00] MÖLLER, Susanne: *Modelle für das Hochschulcontrolling – Unterstützung durch Informations- und Kommunikationssysteme*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Oktober 2000

[Mor00] MORELLI, Francesco: *Klassifikation von Newsfeed*, Technische Universität Hamburg-Harburg, Arbeitsbereich Softwaresysteme, Diplomarbeit, 2000

[Mos00] MOSCHNER, Klaus: *Inhaltsanalyse von Internet-Portals im WWW – Entwicklung und Anwendung eines Kategoriensystems*, Universität Göttingen, Diplomarbeit, Oktober 2000

[MPR00] MANBER, U.; PATEL, A.; ROBISON, J.: Experience with Personalization on Yahoo! In: *Communications of the ACM. Special Issue on Personalization* 43 (2000), August, Nr. 8, S. 35–39

[MS99] MATTHES, Florian; STEFFENS, Ulrike: PIA – A Generic Model and System for Interactive Product and Service Catalogs. In: ABITEBOUL, Serge (Hrsg.); VERCOUSTRE, Anne-Marie (Hrsg.): *Research and Advanced Technology for Digital Libraries, Proceedings of the 3rd European Conference, ECDL'99, Paris, France* Bd. 1696, Springer-Verlag, September 1999. – ISBN 3–540–66558–7, S. 403–422

[MV98] MASERMANN, Ute; VOSSEN, Gottfried: Suchmaschinen und Anfragen im World Wide Web. In: *Informatik-Spektrum* 21 (1998), Februar, Nr. 1, S. 9–15

[MW00] MATTHES, Florian; WEGNER, Holm: Software-Engineering / Technische Universität Hamburg-Harburg, Arbeitsbereich Softwaresysteme. 2000. – Vortragsfolien zur Vorlesung. http://www.sts.tu-harburg.de/teaching/ss-00/SoftEng/

[MWH99] MATTHES, Florian; WEGNER, Holm; HUPE, Patrick: A Process-Oriented Approach to Software Component Definition. In: JARKE, M. (Hrsg.); OBERWEIS, A. (Hrsg.): *Advanced Information Systems Engineering. Proceedings of the 11th International Conference, CAiSE'99, Heidelberg, Germany, June 14-18, 1999*, Springer-Verlag, Juni 1999 (Lecture Notes in Computer Science 1626). – ISBN 3–540–66157–3, S. 26–40

[MyS01] MYSQL AB. *MySQL*. http://www.mysql.com/. 2001

[MZ98] MATTHES, Florian; ZIEMER, Stephan: Understanding SAP R/3:A Tutorial for Computer Scientists / Arbeitsbereich Softwaresysteme, Technische Universität Hamburg-Harburg. 1998. – Forschungsbericht

[Neu01] NEUMANN, Jennifer. *Media Asset Management*. http://www.canto.de/. 2001

[Noh99] NOHR, Holger: Inhaltsanalyse. In: *nfd Information – Wissenschaft und Praxis* 50 (1999), Nr. 2, S. 69–78

[Noh00a] NOHR, Holger; NOHR, Holger (Hrsg.): Automatische Dokumentindexierung – Eine Basistechnologie für das Wissensmanagement / Fachhochschule Stuttgart. 2000. – Arbeitspapiere Wissensmanagement, Nr. 2/2000. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[Noh00b] NOHR, Holger; NOHR, Holger (Hrsg.): Content Management – Die Einführung von Content-Management-Systemen / Fachhochschule Stuttgart. 2000. – Arbeitspapiere Wissensmanagement, Nr. 11/2000. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[Noh00c] NOHR, Holger; NOHR, Holger (Hrsg.): Wissensmanagement in Stuttgarter Unternehmen – Ergebnisse einer Umfrage / Fachhochschule Stuttgart. 2000. – Arbeitspapiere Wissensmanagement, Nr. 10/2000. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[Noh01] NOHR, Holger; NOHR, Holger (Hrsg.): Wissen wird zum Fokus betrieblichen Managements. Eine Kurzübersicht zum Wissensmanagement / Fachhochschule Stuttgart. 2001. – Arbeitspapiere Wissensmanagement, Nr. 4/2001. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[Nor98] NORTH, Klaus: *Wissensorientierte Unternehmensführung. Wertschöpfung durch Wissen*. Wiesbaden: Gabler, 1998

[NRP00] NORTH, Klaus; ROMHARDT, Kai; PROBST, Gilbert: Wissensgemeinschaften: Keimzellen lebendigen Wissensmanagements. In: *io-Management* 69 (2000), Juni, Nr. 7/8, S. 52–62

[NT95] NONAKA, Ikujiro; TAKEUCHI, Hirotaka: *The Knowledge-Creating Company*. Oxford University Press, 1995. – ISBN 0–19–509269–4

[Obe92] OBERMANN, Christof: *Assessment Center – Entwicklung, Durchführung, Trends*. Wiesbaden: Gabler, 1992. – ISBN 3–409–13856–0

[Obj98] OBJECT MANAGEMENT GROUP: The Common Object Request Broker: Architecture and Specification. 1998. – Document 98.2.1, Rev. 2.2.

[ODM97] ODMA COALITION: Open Document Management API, Version 2.0. 1997. – Specification

[Ora01] ORACLE CORPORATION. *Oracle Corporation Home Page*. http://www.oracle.com/. 2001

[Ovu00] Enterprise Portals: New Strategies for Information Delivery. Ovum, 2000. – Studie

[Par00] PARK, Chan-Soo: *A Study on the Web Portal Industry*, School of Public Policy and Management, KDI, Master's Thesis, 2000

[Pep99] PEPPER, Steve: Euler, Topic Maps, and Revolution. In: *Proc. of XML Europe 99*, 1999

[Pfa95] PFAFF, Thomas: *Dokumentenmanagement – das papierlose Büro? Konzepte, Technologien, Tips*. Berlin ; Offenbach: VDE-Verlag, 1995. – ISBN 3–8007–2045–0

[PHP01] *PHP Hypertext Preprocessor*. http://www.php.net/. 2001

[PN00] PANTELIC, Martina; NOHR, Holger; NOHR, Holger (Hrsg.): Data Warehousing / Fachhochschule Stuttgart. 2000. – Arbeitspapiere Wissensmanagement, Nr. 9/2000. – ISSN 1616–5349 (Internet), 1616–5330 (Print)

[Pol58] POLANYI, Michael: *Personal Knowledge: Towards a Post-Critical Philosphy*. University of Chicago Press, Chicago, 1958

[Pol62] POLANYI, Michael: *The Tacit Dimension*. Routledge & Paul, London, 1962

[Pou97] POUR, Nastaran V.: *Entwicklung von Client/Server-Anwendungen mit 4GL-Technologie: Tycoon, NATURAL und ABAP/4 im Vergleich*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, September 1997

[Pre97] PREE, Wolfgang: *Komponentenbasierte Softwareentwicklung mit Frameworks*. Heidelberg: dpunkt.verlag, 1997. – ISBN 3–920993–68–4

[PRR99] PROBST, Gilbert; RAUB, Steffen; ROMHARDT, Kai: *Wissen managen. Wie Unternehmen ihre wichtigste Ressource optimal nutzen*. 3. Auflage. Wiesbaden: Gabler-Verlag, 1999. – ISBN 3–409–39317–X

[Rat99a] RATH, Hans H.: Mit Topic Maps intelligente Informationsnetze aufbauen. In: *iX* iX (1999), Dezember, S. 149 ff.

[Rat99b] RATH, Hans H.: Technical Issues on Topic Maps. In: *Proc. of MetaStructures 99*, 1999

[RBP+91] RUMBAUGH, J.; BLAHA, M.; PREMERLANI, W.; EDDY, F.; LORENSEN, W.: *Object-Oriented Modeling and Design*. Prentice Hall, 1991

[Ric97] RICHTSMEIER, Ingo: *Kommunizierende Informationssysteme am Beispiel autonomer* Internet WebSiteProfiler*: Vergleich objekt- und agentenbasierter Ansätze*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Januar 1997

[Rip98] RIPP, Volker: *Verbesserung der Lokalität und Wiederverwendbarkeit von Geschäftsprozeßspezifikationen: Probleme und Lösungsansätze am Beispiel kundenorientierter Hotelgeschäftsprozesse*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, März 1998

[RJB99] RUMBAUGH, James; JACOBSON, Ivar; BOOCH, Grady: *The Unified Language Reference Manual*. Addison-Wesley Publishing Company, 1999

[RMS00] REIMANN, Peter; MÜLLER, Katja; STARKLOFF, Philipp: Kognitiv kompatibel? Wissensmanagement: Brückenschlag zwischen Technik und Psyche. In: *c't* (2000), Nr. 4, S. 274–281

[RMS+01] RAULF, Martin; MÜLLER, Rainer; STEFFENS, Ulrike; MATTHES, Florian; SCHEUNERT, Klaus J.; SCHMIDT, Joachim W.: Begriffsorientierte Dokumentenverwaltung für das internetgestützte Projektmanagement – Der FHH InfoBroker für das Projekt "sap für hamburg" -. In: *Tagungsband 4. GI-Fachgruppentagung "Management und Controlling von IT-Projekten", Glashütten (Taunus)*, dpunkt.verlag, März 2001

[RP99] RATH, Hans H.; PEPPER, Steve: Topic Maps: Introduction and Allegro. In: *Proc. of Markup Techologies 99*, 1999

[RR01] ROTHFUSS, Gunther; RIED, Christian: *Content Management mit XML*. Berlin ; Heidelberg ; New York: Springer-Verlag, 2001 (Xpert.press). – ISBN 3–540–66594–3

[RS99] RESCORLA, E.; SCHIFFMAN, A.: The Secure HyperText Transfer Protocol. 1999. – RFC 2660

[RSG01] ROSSI, Gustavo; SCHWABE, Daniel; GUIMARÃES, Robson: Designing Personalized Web Applications. In: *Proceedings of Tenth International World Wide Web Conference, May 1-5, Hong Kong*, 2001, S. 275–284

[SA99] SANDER, Jörg E.; ACKERMANN, Michael: Wissensmangement richtig einführen. In: *Diebold Management Report* (1999), Nr. 7, S. 19–23

[SAD99] STUDER, Rudi; ABECKER, Andreas; DECKER, Stefan: Informatik-Methoden für das Wissensmanagement. In: *Festschrift zum 60. Geburtstag von Prof. Dr. Wolffried Stucky*. Stuttgart: Teubner Verlag, Juli 1999

[SAP98] SAP AG. *Business Application Programming Interfaces (BAPI)*. http://www.sap-ag.de/bfw/interf/bapis/bapi.htm. 1998

[SAP00] Introducing Knowledge Management with mySAP.com. Walldorf: SAP AG, 2000. – White Paper

[SAP01] SAP AG. *mySAP.com*. http://www.mysap.com/solutions/. 2001

[Sch96] SCHÜPPEL, Jürgen: *Wissensmanagement*. Wiesbaden: Deutscher Universitäts-Verlag, 1996. – ISBN 3–8244–6304–0

[Sch97a] SCHNEIDER, Jochen: *Transactions: Entwurf eines wissenschaftlichen Publikationssystems*, Fachbereich 3, Universität Bremen, Diplomarbeit, 1997

[Sch97b] SCHULMEISTER, Rolf: *Grundlagen hypermedialer Lernsysteme. Theorie – Didaktik – Design*. 2. Auflage. München: Oldenbourg, 1997. – ISBN 3486244191

[Sch99a] SCHMIDT, Carsten: *Konzeption eines Groupware-basierten Enterprise Knowledge Portals*, Lehr- und Forschungseinheit Wirtschaftsinformatik 2, Universität-Gesamthochschule Paderborn, Diplomarbeit, 1999

[Sch99b] SCHULTZE, Ulrike: Investigating the Contradictions in Knowledge Management. In: LARSEN, T.J. (Hrsg.); LEVINE, L. (Hrsg.); DEGROSS, J.I. (Hrsg.): *Proceedings of IFIP Working groups 8.2 and 8.6 joint Working Conference on Information Systems: Current Issues and Future Changes, Helsinki, Finland, Dezember 1998*. Laxenburg: IFIP, 1999. – ISBN 3–901–882–02–2, S. 155–174

[Sch00a] SCHMIDT, Michael Peter: *Knowledge Communities: Mit virtuellen Wissensmärkten das Wissen in Unternehmen effektiv nutzen*. Addison-Wesley Publishing Company, 2000. – ISBN 3827316758

[Sch00b] SCHMIDT, R. (Hrsg.): Deutsche Gesellschaft für Informationswissenschaft und Informationspraxis (DGI), Mai 2000 . – ISBN 3–925474–41–2

[Sch00c] SCHMITZER, Benno: *Beiträge zur Verwendung der Framework-Technologie bei der Entwicklung und Einführung von Systemen der betrieblichen Informationsverarbeitung*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Dissertation, 2000

[Sch01] SCHULMEISTER, Rolf: *Virtuelle Universität – Virtuelles Lernen*. München: Oldenbourg, 2001. – ISBN 3486257420

[Sea69] SEARLE, J.: Speech Acts / Cambridge University Press. 1969. – Forschungsbericht

[Seg00] SEGAL, Dan: Wissensmanagement-Tools – Taxonomien, Marktüberblick und zukünftige Entwicklungen. In: [Gro00a], S. 21–30

[Sha48] SHANNON, C.: A Mathematical Theory of Communication. In: *Bell Systems Journal* 27 (1948), S. 397–423, 623–656

[Sha01] Introducing Microsoft SharePoint Portal Server 2001 / Microsoft Corporation. 2001. – White Paper

[Sku98] SKUSA, Michael: *Integration relationaler Datenbanken in ein polymorphes, persistentes Objektsystem*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, September 1998

[SM98] SCHMIDT, Joachim W.; MATTHES, Florian: *Datenbankhandbuch Kapitel 1: Datenbankmodelle und Datenbanksprachen*. 1998. – (unveröffentlichtes Manuskript)

[SOF00] SCHWICKERT, Axel C.; OSTHEIMER, Bernhard; FRANKE, Thomas S.: eUniversity – Web-Site-Generierung und Content Management für Hochschuleinrichtungen. Mainz: Lehrstuhl für allgemeine BWL und Wirtschaftsinformatik, Johannes Gutenberg Universität, 2000. – Arbeitspapiere WI, Nr. 9/2000

[Sof01] SOFTWARE AG. *Tamino*. http://www.softwareag.com/tamino/default.htm. 2001

[SRL+90] STONEBRAKER, M.; ROWE, L.A.; LINDSAY, B.; GRAY, J.; CAREY, M.; BRODIE, M.; BERNSTEIN, P.: Third-Generation Data Base System Manifesto. In: *ACM SIGMOD Record* 19 (1990), September, Nr. 3, S. 31–44

[SS99] SCHUMACHER, Mark; SCHWICKERT, Axel C.: Web-Portale – Stand und Entwicklungstendenzen. Mainz: Lehrstuhl für allgemeine BWL und Wirtschaftsinformatik, Johannes Gutenberg Universität, 1999. – Arbeitspapiere WI, Nr. 4/99

[SSSW01] SCHMIDT, Joachim W.; SEHRING, Hans-Werner; SKUSA, Michael; WIENBERG, Axel: Subject-Oriented Work: Lessons Learned from an Interdisciplinary Content Management Project. In: *Proceedings of the Fifth East-European Conference on Advances in Databases and Information Systems*, Springer, 2001 (Lecture Notes in Computer Science)

[Sta01] STAMER, Sören: Next Generation Content Management & Delivery in der Content Economy. CoreMedia AG, Mai 2001. – Folienvortrag auf der Internet World 2001

[Sto96] STONEBRAKER, M.: *Object-Relational DBMSs – The Next Great Wave*. Morgan Kaufman, 1996

[Stö99] STÖVESAND, Jan: *Intranetunterstützung für problemlösende Arbeitsprozesse im Team*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, August 1999

[Sun01a] SUN MICROSYSTEMS. *Enterprise Java Beans (EJB) technology*. http://java.sun.com/products/ejb/. 2001

[Sun01b] SUN MICROSYSTEMS. *Java Remote Method Invocation (RMI)*. http://java.sun.com/products/jdk/rmi/index.html. 2001

[Sun01c] SUN MICROSYSTEMS. *Java Server Pages (JSP)*. http://java.sun.com/products/jsp/. 2001

[Sun01d] SUN MICROSYSTEMS. *JDBC Data Access API*. http://java.sun.com/products/jdbc/. 2001

[Sve97a] SVEIBY, Karl-Erik: The Intangible Assets Monitor. In: *Journal of Human Resource Costing & Accounting* 2 (1997), Nr. 1

[Sve97b] SVEIBY, Karl-Erik: *The New Organizational Wealth*. Berrett-Koehler, San Francisco, 1997. – ISBN 1–57675–014–0

[Sve97c] SVEIBY, Karl-Erik. *Tacit Knowledge*. http://www.sveiby.com.au/Polanyi.html. Dezember 1997

[Sve98] SVEIBY, Karl-Erik. *Measuring Intangibles and Intellectual Capital – An Emerging First Standard*. http://www.sveiby.com.au/EmergingStandard.html. August 1998

[SW00a] SCHUSTER, Erwin; WILHELM, Stephan: Aktuelles Schlagwort: Content Management. In: *Informatik Spektrum* 23 (2000), Dezember, Nr. 6, S. 373–375

[SW00b] SCHUSTER, Erwin; WILHELM, Stephan; BULLINGER, Hans-Jörg (Hrsg.): Content Management Systeme: Auswahlstrategien, Architekturen und Produkte / Fraunhofer Institut Arbeitswirtschaft und Organisation (IAO), Stuttgart. 2000. – Forschungsbericht. – ISBN 3–7754–0169–5

[Tam99] Tamino – The Information Server for Electronic Business / Software AG. 1999. – White Paper

[TB99] THIESSE, Frédéric; BACH, Volker: Tools und Architekturen für Business Knowledge Management. In: BACH, Volker (Hrsg.); VOGLER, Petra (Hrsg.); ÖSTERLE, Hubert (Hrsg.): *Business Knowledge Management. Praxiserfahrungen mit Intranet-basierten Lösungen*. Springer-Verlag, 1999. – ISBN 3–540–65246–9, Kapitel 3, S. 85–116

[Tec00] TECHNISCHE UNIVERSITÄT HAMBURG-HARBURG, ARBEITSBEREICH SOFTWARESYSTEME. *Business Conversations Project Home Page*. http://www.sts.tu-harburg.com/projects/BC/. 2000

[The00] THE UNICODE CONSORTIUM (Hrsg.): *The Unicode Standard, Version 3.0*. Reading, Massachusettes: Addison-Wesley Publishing Company, 2000. – ISBN 0–201–61633–5

[The01] THEBRAIN TECHNOLOGIES CORP. *TheBrain Home Page*. http://www.thebrain.com/. 2001

[Thu00] THUROW, Karsten: *Ein generisches Notifikations-Framework*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Dezember 2000

[Tka99a] TKACH, Daniel: Knowledge Mapping. IBM Institute for Knowledge Management, 1999. – Advances In Knowledge Management

[Tka99b] TKACH, Daniel: Knowledge Portals. IBM Institute for Knowledge Management, 1999. – Advances In Knowledge Management

[Tol99] TOLKSDORF, Robert: XML und darauf basierende Standards: Die neuen Auszeichnungssprachen des Web. In: *Informatik Spektrum* 22 (1999), Dezember, Nr. 6, S. 407–421

[Töp97] TÖPFER, A. (Hrsg.): *Benchmarking: Der Weg zu Best Practice*. Springer-Verlag, 1997

[Top01] *Topic Maps*. http://www.topicmaps.org/. 2001

[Tur98] TURAU, Volker: Aktuelles Schlagwort: Web-Roboter. In: *Informatik-Spektrum* 21 (1998), Juni, Nr. 3, S. 159–160

[Tur99] TURAU, Volker: Techniken zur Realisierung Web-basierter Anwendungen. In: *Informatik-Spektrum* 22 (1999), Februar, Nr. 1, S. 3–12

[Ver00a] Verity Search. The Advantages of Advanced Information Retrieval / Verity Inc. 2000. – Verity White Paper

[Ver00b] VERMEIJ, Richard: The Portal Game. Palo Alto: Arthur D. Little, E-Business Center, August 2000. – Forschungsbericht

[Ver01] VERITY INC. *Verity Home Page*. http://www.verity.com/. 2001

[Vog00] VOGELSANG, Karlheinz: Personalinformations-Systeme und ihr Beitrag zum Wissensmanagement. In: *wissensmanagement* (2000), März, Nr. 3, S. 13–16

[Wag99] WAGNER, Hubert: *Wissensmanagement in Unternehmensberatungen*, Lehrstuhl für Wirtschaftsinformatik, Universität Saarbrücken, Diplomarbeit, 1999

[Wah98] WAHLEN, Jens: *Entwurf einer objektorientierten Sprache mit statischer Typisierung unter Beachtung kommerzieller Anforderungen*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Juni 1998

[Wal01a] WALOSZEK, Gerd: Ingredients of Portals – An Overview / SAP AG, Usability Engineering Center. SAP Design Guild, 2001. – Forschungsbericht

[Wal01b] WALOSZEK, Gerd: Is a Portal a Portal? / SAP AG, Usability Engineering Center. SAP Design Guild, 2001. – Forschungsbericht

[Wal01c] WALOSZEK, Gerd: Portal Usability – Is There Such A Thing? / SAP AG, Usability Engineering Center. SAP Design Guild, 2001. – Forschungsbericht

[Web01a] *WebDAV Resources*. http://www.webdav.org/. 2001

[Web01b] WEBER, Volker: People, Places & Things. Lotus Discovery Server fürs Wissensmanagement. In: *c't* (2001), September, Nr. 20, S. 142–144

[Weg98] WEGNER, Holm: *Objektorientierter Entwurf und Realisierung eines Agentensystems für kooperative Internet-Informationssysteme*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Mai 1998

[Weg99] WEGGEMAN, Mathieu: *Wissensmanagement – Der richtige Umgang mit der wichtigsten Ressource des Unternehmens*. Bonn: MITP Verlag, 1999. – ISBN 3–8266–0508–X

[Wei98] WEIKARD, Marc: *Entwurf und Implementierung einer portablen multiprozessorfähigen virtuellen Maschine für eine persistente, objektorientierte Programmiersprache*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, Juli 1998

[WF86] WINOGRAD, T. (Hrsg.); FLORES, C.F. (Hrsg.): *Understanding computers and cognition – a new foundation for design*. Norwood, New Jersey: Ablex Publishing Coorporation, 1986

[WHM00] WEGNER, Holm; HUPE, Patrick; MATTHES, Florian: A Process-Oriented and Content-Based Perspective on Software Components. In: *Information Systems* 25 (2000), Nr. 2, S. 135–156

[Wie97] WIENBERG, Axel: *Bootstrap einer persistenten objektorientierten Programmierumgebung*, Fachbereich Informatik, Universität Hamburg, Studienarbeit, August 1997

[Wil99] WILDE, Erik: *World Wide Web: Technische Grundlagen*. Springer-Verlag, 1999. – ISBN 3–540–64700–7

[Win87] WINOGRAD, T.A.: A Language/Action Perspective on the Design of Cooperative Work / Stanford University. 1987 (No. STAN-CS-87-1158). – Forschungsbericht

[Win88] WINOGRAD, T.: Where the action is. In: *Byte* (1988), Dezember, S. 256–258

[Win01] WINKLER, Ramona: Portals – The All-In-One Web Supersites: Features, Functions, Definitions, Taxonomy / SAP AG, Product Design Center. SAP Design Guild, 2001. – Forschungsbericht

[Wir86] WIRTH, Niklaus: *Algorithmen und Datenstrukturen mit Modula-2*. 4. Auflage. Stuttgart: Teubner Verlag, 1986. – ISBN 3–519–02260–5

[Wit63] WITTGENSTEIN, Ludwig: *Tractatus logico-philosophicus. Logisch-philosophische Abhandlung*. Frankfurt/Main: Suhrkamp, 1963

[Wit00] WITTENBURG, André: *Konzepte und Techniken zur Adaption von betrieblicher Standardsoftware*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, März 2000

[WML00] WAP-WML Language Specification. Wireless Application Protocol Forum, Februar 2000. – Forschungsbericht

[WS98] WOODS, Eric; SHEINA, Madan: *Knowledge Management: Application, Markets and Technologies*. Ovum, Oktober 1998

[WWW98a] WWW CONSORTIUM (W3C). *Extensible Markup Language (XML) Specification 1.0*. http://www.w3.org/TR/REC-xml. 1998

[WWW98b] WWW CONSORTIUM (W3C). *The Information and Content Exchange (ICE) Protocol*. http://www.w3.org/TR/NOTE-ice. 1998

[WWW99] WWW CONSORTIUM (W3C). *XML Schema Part 1: Structures and Datatypes*. http://www.w3.org/TR/xmlschema-1/, http://www.w3.org/TR/xmlschema-2/. 1999

[Zac00] ZACK, Michael H.: Researching Organizational Systems using Social Network Analysis. In: *Proceedings of the 33rd Hawaii International Conference on System Sciences, Maui, Hawaii (IEEE 2000)*, 2000

[Zad65] ZADEH, Lotfi A.: Fuzzy sets. In: *Information and Control* 8 (1965), S. 338–353

[Zad98] ZADE, Heiko: *Visualisierungsunterstützung für kooperative Aktivitäten im Internet*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, März 1998

[ZHMS01] ZIEMER, Stephan; HUPE, Patrick; MATTHES, Florian; SCHMIDT, Joachim W.: *Enterprise Service Maps: Bringing the Public Transport Map Metaphor to Work*. 2001. – Eingereicht zur CAiSE'02

[Zie97] ZIEMER, Stephan: *SAP R/3: An Overview of its Concepts and Languages*, Fachbereich Informatik, Universität Hamburg, Studienarbeit, September 1997

[Zie98] ZIEMER, Stephan: *Eine transportorientierte Workflow-Sprache: Definition, Anwendung und Bewertung*, Fachbereich Informatik, Universität Hamburg, Diplomarbeit, November 1998

[Zöl00] ZÖLLER, Bernhard: Web definiert DMS neu. In: *Information Week* (2000), September, Nr. 21, S. 16

[Zül98] ZÜLLIGHOVEN, Heinz: *Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz*. 1. Auflage. Heidelberg: dpunkt.verlag, 1998. – ISBN 3–932588–05–3

Lebenslauf

Holm Wegner

* 2.12.1969	Hamburg
1976-1980	Grundschule Ahrensburger Weg
1980-1989	Walddörfer Gymnasium in Volksdorf
1989	Abitur
1989-1990	Wehrdienst in Flensburg
1990-1998	Studium der Informatik an der Universität Hamburg
1993	Vordiplom
1998	Diplom
1998-2001	Wissenschaftlicher Mitarbeiter am Arbeitsbereich Softwaresysteme der Technischen Universität Hamburg-Harburg
2002-...	*Professional* im Entwicklungsbereich der CSC Ploenzke AG, Hamburg

Erklärung

Hiermit erkläre ich, daß ich die vorliegende Dissertation selbständig durchgeführt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Hamburg, den 1. Januar 2002

(Holm Wegner)